Entity Summarisation for Entity-centric Publish/Subscribe Systems

Niki Pavlopoulou

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

Advisor: **Prof. Edward Curry**
External Examiner: **Prof. Payam Barnaghi**
Internal Examiner: **Dr. Michael Schukat**

Data Science Institute
The Insight Centre for Data Analytics
National University of Ireland, Galway
November 2021

Declaration of Authorship

I, Niki Pavlopoulou, declare that this thesis titled "Entity Summarisation for Entity-centric Publish/Subscribe Systems" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: ____

Date: ____

Abstract

The Internet of Things (IoT) has contributed to physical devices generating entity-centric data (e.g. smart buildings). To bridge the gap between the devices' data and the users' interests, Publish/Subscribe systems (Pub/Sub) are suitable middleware to deal with dynamic large-scale IoT applications due to their decoupling traits. However, the IoT contains more challenges than dynamism related to data and users. Specifically, data can be voluminous and heterogeneous due to integration or enrichment as well as redundant or semantically similar due to the sensors' spatial proximity. Existing approaches tackle semantic interoperability through ontologies and taxonomies resulting in rigidness, non-scalability, and domain-dependency. At the same time, users can either create representationally-coupled queries that could be complex (e.g. SPARQL), independent of their data knowledge and expertise, or simple queries that lead to redundant information, which can overwhelm them. Existing approaches either use complex queries or create high-level data abstractions that are either not usable or complex for dynamic environments and suffer from representational coupling.

This thesis addresses these problems and analyses two research questions involving the formulation of a new Pub/Sub scheme; the Entity-centric Publish/Subscribe Summarisation System that involves user-friendly and contextually-aware subscriptions as well as extractive and abstractive summarisation approaches for the publications. Its goal is to address usability, user expressibility, data expressiveness, user and data effectiveness, and system efficiency. Three approaches are proposed; PubSum, IoTSAX, and PoSSUM. PubSum is a dynamic diverse entity summarisation of heterogeneous Linked Data streams through windowing policies, embedding-based DBSCAN clustering, and geometric-based top-k ranking. IoTSAX is a dynamic abstractive summarisation of heterogeneous numerical entity graph streams through enhanced Symbolic Aggregate approximation (SAX) and approximate rule-based reasoning. PoSSUM is an extractive and abstractive diverse summarisation of heterogeneous numerical and Linked Data streams through novel partly-incremental conceptual clustering based on embedding models and variance as well as contextual-based top-k ranking. As an example, doctors are not experts in query languages and are unaware of the content and representations of patient data in a system. The proposed system will require a simple patient-centric subscription that will create a summary as a notification. This summary will be abstractive by interpreting the shape of real-time health sensor readings and providing a high-level inference as well as extractive by including the most important and conceptually/contextually diverse information coming from external sources (e.g. personal information).

The proposed system has been extensively evaluated by synthetic and real-world data from the domains of Healthcare and Smart Cities achieving comparable results in correctness and system performance. Specifically, PubSum, involving DBpedia data, achieves up to 92% reduction of forwarded messages, 69.3% duplication reduction, and 0.95 redundancy-aware F-score compared to traditional Pub/Sub, but at the expense of 4 times more latency, while achieving 6 times less latency and 3 times less memory compared to the state-of-the-art diverse entity summarisation with throughput ranging from 833 to 1,005 events/second. IoTSAX, involving real-world heterogeneous data related to Healthcare and Smart Cities, achieves up to 0.87 reasoning F-score, 98% reduction of forwarded messages, and outperforms the original SAX in approximation error (2 to 3 times less) and compression space-saving percentage when data redundancy occurs (from 71.75% to 94.99%) while maintaining similar or better latency and throughput. The latency is 2 to 3 times more compared to traditional Pub/Sub and the throughput ranges from 13.231 to 97.393 events/second. PoSSUM, involving real-world heterogeneous data, discovers up to 80% data diversity desire by users and achieves the best summary quality for more than half of the entities as well as the best conceptual clustering F-score from 0.69 to 0.83 compared to traditional Pub/Sub and the state-of-the-art diverse entity summarisation. Also, up to 0.95 redundancy-aware F-score and 99% message reduction compared to traditional Pub/Sub. Finally, it has less clustering processing time, scoring and memory consumption, and comparable latency and throughput.

To Mum, Dad, Grandma, Sofia, and Brendan

Acknowledgments

My biggest gratitude goes to my supervisor Prof. Edward Curry for teaching me what it is to be a true scientist. His continuous support, patience, and kindness have helped me bringing this PhD to fruition. No words can truly convey the amount of assistance and support he provided me in terms of work, advice, guidance, and personal development. I will forever be grateful to him for giving me the opportunity and the resources to start and complete my PhD journey and gain invaluable experience.

I would like to thank my GRC committee members Dr. Matthias Nickles, Dr. Ojo Adegboyega, Dr. Colm O'Riordan, Prof. Dietrich Rebholz-Schuhmann, and Dr. Souleiman Hasan for their guidance and feedback during different stages of my PhD as well as my board of examiners Prof. Payam Barnaghi and Dr. Michael Schukat. My thanks also go to Dr. Amin Anjomshoaa for reviewing my papers and giving me constructive feedback, Dr. Souleiman Hasan and Dr. Umair ul Hassan for giving me much needed guidance for the early stages of my research, and my colleagues Asra Aslam, Piyush Yadav, and Tarek Zaarour for their support, discussions, and feedback. Thanks also go to all the other colleagues and friends that from a friendly chat in a corridor to going out for a walk by the lake contributed to a positive aspect of my PhD journey.

I would like to gratefully acknowledge the funders Big Data Value ecosystem (BDVe) and Science Foundation of Ireland (SFI) for supporting my PhD. This work has emanated from a research grant from the European Union's Horizon 2020 research programme BDVe grant No 732630 and in part from SFI under Grant Number SFI/12/RC/2289_P2, co-funded by the European Regional Development Fund.

My warmest thanks go to my parents Evgenia and Constantinos for without them I would not be where I stand today, my sister Sofia, who is my role model and my constant rock, and my partner Brendan, who is my number one supporter and constantly encourages me to reach my dreams. This PhD is also dedicated to my grandmother Sofia, who was like a second mother to me.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Emerging Technological Trends

"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.", stated Mark Weiser [2]. With this in mind, the future is painted to be promising as a lot of attention is drawn to technologies that could seamlessly connect people, devices, and entities/things on a large scale to create a unified data landscape that could lead to intelligent operations [3–5]. As all technology trends, a data landscape of this scale may pose some ethical dilemmas. Aldous Huxley in the novel "Brave New World" [6] described a future world that will rely too much on technology and will lead to a dystopia with people being bereft of intuition, thought, and emotions. Nevertheless, the noble goal of technology is to facilitate people's lives in a great manner. Terminologies like the "Internet of Things" (IoT) have flooded the news and have raised important issues and challenges both in academia and industry as well as non-experts interested in technology trends. IoT is aspiring to be one of these technologies that will "disappear" in the future.

1.1.1 Internet of Things

IoT is a technology that refers to the inter-connectivity, over the Internet, of physical devices embedded with sensors and actuators [7] that through the generation of data and its analysis in real-time, valuable insight and important actions take place. Its goal is to turn the physical and virtual world into an information one by providing a network infrastructure with interoperable communication protocols and software [8]. An IoT environment consists of physical devices such as smart cars, smartphones, smart watches, and computers that are all connected to create a unified data landscape. The IoT's applications could be vast ranging from Healthcare (Healthcare IoT or Internet of Medical Things) to Smart Cities [9–11]. For example, smart devices that monitor a person's health (heart beat etc.) could contribute to real-time remote patient monitoring, while a digital twin of a wind turbine could monitor its performance and environmental aspects (energy output, temperature etc.). The overall goal of IoT is to create a network of information that could facilitate end-users ranging from an average citizen to a governing body to an industrial organisation [12]. It is expected to be one of the main future technologies since the advent of the 5G wireless network to scale IoT coverage will lead to a surge in connected devices to the magnitude of billions by 2025 [13].

1.1.2 Event-based Communication Paradigms

IoT has several architecture models [7, 14, 15] with four main layers: Sensing, Network, Middleware, and Application, illustrated in Fig. 1.1. This thesis' focus is the Middleware layer that stores the data received from the Network layer, processes, and analyses it (e.g. event-based systems). It shares the data or its analysis between applications and devices, and it provides event notifications.

There is an abundance of middleware (systems) that can process "flows of information" as termed by Cugola and Margara [16]. Nevertheless, not all systems are suitable for IoT environments and there is no standardisation among them in terms of architecture, data models, rule languages, and processing methodologies. For IoT, point-to-point and synchronous communication among things would hinder the performance of such a dynamic environment. Suitable interaction schemes need to be defined for dynamic large scale applications.

The most prominent interaction scheme is Publish/Subscribe systems (Pub/Sub) [1], where three parties are involved; publishers, subscribers, and the event service. Publishers generate data (events or publications) while subscribers register their interest in an event or pattern of events (like queries). All publications and subscriptions are sent to the event service, which is responsible for detecting matches when an event fully satisfies all the requirements of a subscription. If a match occurs, then, the events are sent as notifications to the corresponding

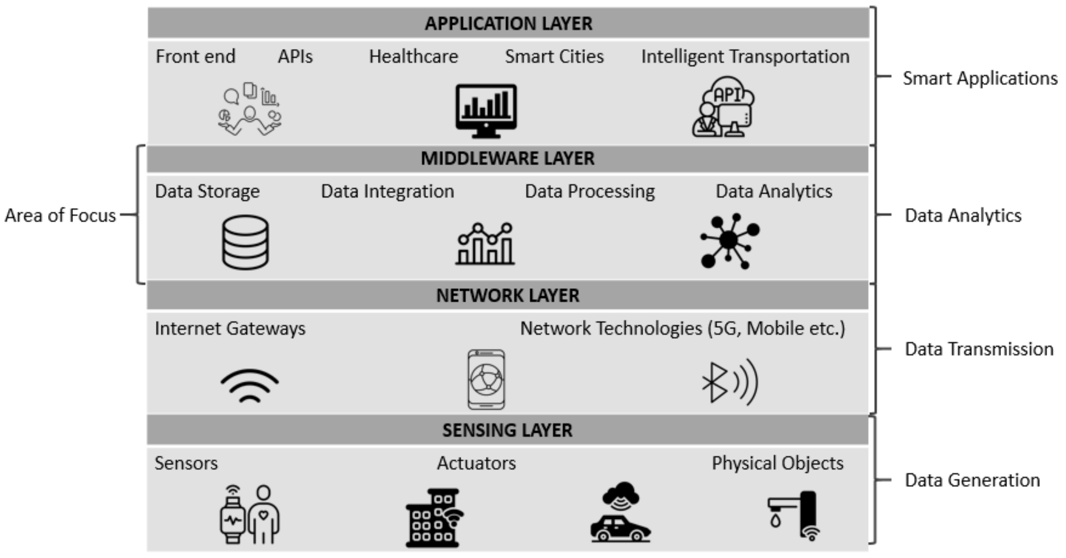


Figure 1.1: Architecture of IoT.

subscribers. A simple Pub/Sub is depicted in Fig. 1.2.

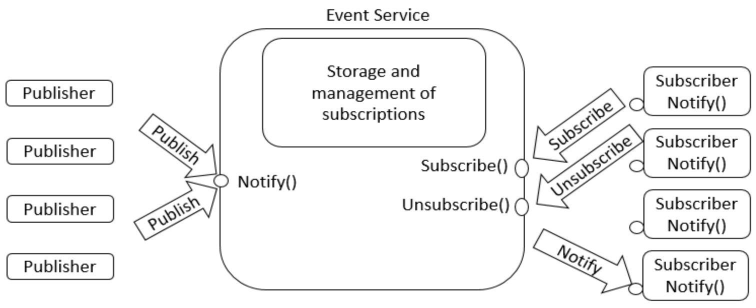


Figure 1.2: A simple Pub/Sub system. Adapted by Eugster et al. [1]

1.2 Motivation and Problem Overview

An architecture as complex as IoT is expected to contain a high volume of devices (sources) and end-users generating data and performing queries based on the data and specific needs, respectively. The sources are related to multiple entities, that is real-world or abstract things, which through integration and sharing result in diversity in a conceptual and contextual level. This creates a vast information range available to the end-users that may prove challenging for data sharing and communication among systems and users. Furthermore, the sources could belong to different manufacturers and follow their own standards/protocols; therefore, they contain different data representations in the form of schemata and semantics [12, 13]. This creates a representational heterogeneity on a semantic and schematic level, where schemata refer to the syntactic structure of the data (e.g. class hierarchies or relations in a graph), whereas semantics refers to the lack of specific words used within the data and the possibility of the use of synonyms and related words instead. At the same time, the end-users are interested in the analysis or the interpretation of data deriving from the sources for specific applications by performing queries. The interpretation and specific query creation derive from humans' subjective nature and may be affected by personal experiences or biases [17, 18]. Also, queries could be expressed in different ways to satisfy the needs of users ranging from simple to complex ones, assuming the user has the necessary background knowledge. For example, technically acquired end-users (e.g. developers) are experts in query languages so they can provide targeted to their needs complex queries. On the other hand, day-to-day end-users (e.g. doctors) are not involved in the design of the system and they have no a-priori information regarding the system's architecture, data representations, the content provided by the available sensors, and query languages to create targeted queries. System performance is also important in terms of scalability, real-time communication and analysis, and resource constraints. Since the large volume of devices and end-users is dynamic, it can increase at any moment generating continuous (unbounded) data with high velocity [8]. Also, architectures should dynamically interact with devices/things and perform data analytics in real-time [12, 15, 19]. Finally, the devices are low in energy (e.g. battery-powered devices), memory, and processing power (e.g. nodes with limited capabilities) [3, 12, 19].

It is evident, then, that there are challenges in the data, user, and system side of IoT environments (Fig. 1.3). The goal is to achieve semantic interoperability and to facilitate the end-users in an effective and efficient manner. The non-expert users (e.g. doctors) should be abstracted by the schematic and semantic complexity, and should provide the simplest possible queries through an app or interface like querying on the Web, but not at the expense of the satisfaction of their needs [20]. There is a demand, then, for data sharing, processing, communication among things of different systems, and a need for usability and query simplicity. At the same time, this should occur in a scalable [15], low latency, high throughput [13], and lightweight way. This is not a trivial task and one way to tackle this is to create a context-aware Pub/Sub system that could efficiently and effectively map user queries (subscriptions) to generated heterogeneous data (publications). The Pub/Sub system should also be employed with Artificial Intelligence techniques to create an intelligent IoT environment characterised by extensibility, self-configuration, self-optimisation, scalability, and interoperability among heterogeneous devices with low processing power and reasoning capabilities [15]. The focus of this thesis is to propose a Pub/Sub system with the aforementioned capabilities with an emphasis on end-users that have no a-priori knowledge and they require an unbiased and contextually-aware information by providing minimal configurations to express their needs. Expert users could also use this type of Pub/Sub system; nevertheless, for more targeted information by providing complex queries, other existing types of Pub/Sub could be used. Challenges involving data security and user privacy [21] are out of scope.

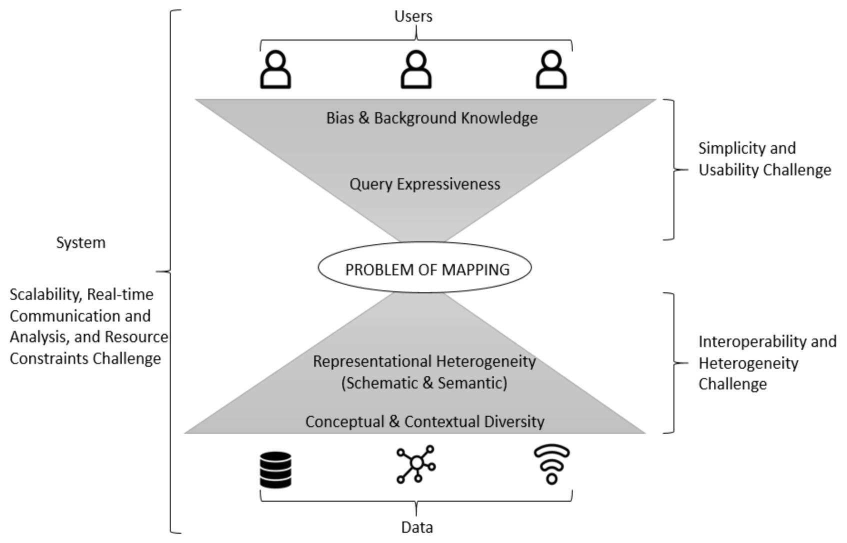

Users
Bias & Background Knowledge
Query Expressiveness
Simplicity and Usability Challenge
System
Scalability, Real-time Communication and Analysis, and Resource Constraints Challenge
PROBLEM OF MAPPING
Representational Heterogeneity (Schematic & Semantic)
Conceptual & Contextual Diversity
Interoperability and Heterogeneity Challenge
Data

Figure 1.3: IoT challenges addressed in this thesis.

1.2.1 Different Levels of Query Expressiveness

Users involved in IoT have different levels of technical expertise and data understanding. A data landscape of this complexity with voluminous devices generating continuous and ever-changing data of high velocity could prove overwhelming for non-expert users. The burden falls to the users to be aware of query languages, schemata, and semantics used by the data, and the content as well as the context of the data in order to satisfy their specific needs [20, 22].

This defies the purpose of IoT as its aspiration is to facilitate people's lives and not demand complex tasks from their side.

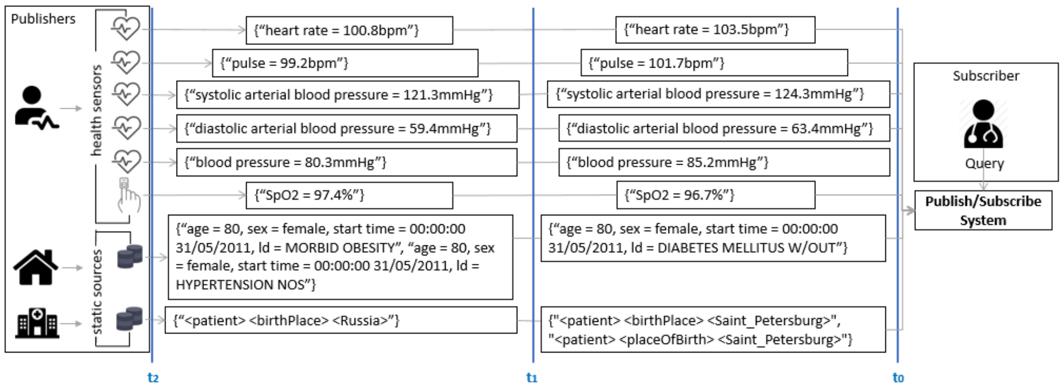
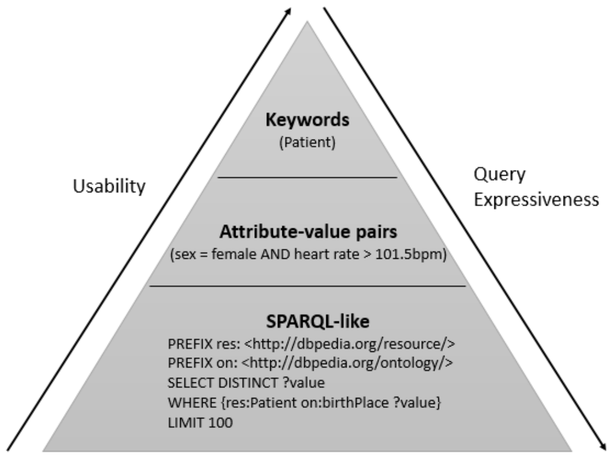

Publishers
health sensors
static sources
{"heart rate = 100.8bpm"}
{"pulse = 99.2bpm"}
{"systolic arterial blood pressure = 121.3mmHg"}
{"diastolic arterial blood pressure = 59.4mmHg"}
{"blood pressure = 80.3mmHg"}
{"SpO2 = 97.4%"}
{"age = 80, sex = female, start time = 00:00:00 31/05/2011, Id = MORBID OBESITY", "age = 80, sex = female, start time = 00:00:00 31/05/2011, Id = HYPERTENSION NOS"}
{"<patient> <birthPlace> <Russia>"}
{"heart rate = 103.5bpm"}
{"pulse = 101.7bpm"}
{"systolic arterial blood pressure = 124.3mmHg"}
{"diastolic arterial blood pressure = 63.4mmHg"}
{"blood pressure = 85.2mmHg"}
{"SpO2 = 96.7%"}
{"age = 80, sex = female, start time = 00:00:00 31/05/2011, Id = DIABETES MELLITUS W/OUT"}
{"<patient> <birthPlace> <Saint_Petersburg>", "<patient> <placeOfBirth> <Saint_Petersburg>"}
Subscriber
Query
Publish/Subscribe System
t2
t1
t0

Figure 1.4: A Pub/Sub system example.

For example, in Fig. 1.4 a doctor (subscriber) receives a plethora of medical information (publications) regarding a patient (entity). This information derives from several body sensors measuring real-time physiological conditions, medical history, and living conditions of the patient. Although there are different kinds of queries (subscriptions) that the doctor can provide, there is a trade-off between usability and query expressiveness [22, 23] as higher query complexity results in more expressiveness, but lower usability (Fig. 1.5). Keyword-based queries or topics in Pub/Sub are the simplest form of query. These queries are easily defined and clearly understood by the non-expert users. Due to their simplicity and shortness, the returned results may contain abstract, redundant, or non-relevant information [24]. The users are usually required to go through the information and gather more targeted to their needs results. Attribute-value pairs queries consist of tuples containing individual or logical combinations (and, or etc.) of constraints through comparison operators (=, <, ≤, >, ≥) on attribute-value pairs. The constraints should follow the data schema, and results are returned only if they completely satisfy all constraints. The users are required not only to be aware a-priori of the data schema and available content, but also of the semantics of the data, otherwise, no information will be returned (representational coupling). For an ever-evolving data landscape, the users will also need to be updated on the changes. SPARQL-like queries consist of graph-based queries with specific nodes and their relation (edges) among them in a graph. Results are returned if they completely satisfy the query [25]. The users need to know the data schema, semantics (representational coupling), and content a-priori to perform the query, and to be experts in complex query languages [26]. Also, graphs are complex structures containing numerous nodes and edges; therefore, not only do queries need to be strict enough to return relevant results, but when the graph is updated, users may have more options for new queries or irrelevant results from their old queries. A comparison of the different query or subscription kinds is given in Table 1.1.

The lack of a commonly accepted query language and the fact that none of the current kinds of queries can achieve high usability, while maintaining high query expressiveness, is problematic. This problem could be formulated as finding a Pub/Sub system that can support representational decoupling (removing representational dependencies) in user queries, while expressing their information needs, which is one of the problems addressed in this thesis. The focus is on facilitating non-expert end-users, although expert users could also avail of this system.



```
PREFIX res: <http://dbpedia.org/resource/>
PREFIX on: <http://dbpedia.org/ontology/>
SELECT DISTINCT ?value
WHERE {res:Patient on:birthPlace ?value}
LIMIT 100
```

Figure 1.5: Trade-off between usability and query expressiveness.

Table 1.1: Overall comparison of different kinds of queries

Query Type	**Schema & Semantic Agnostic**	**Relevant Results**	**Query Language Expertise**	**Exact Queries**
Keywords	✓			
Attribute-value pairs		✓	✓(medium)	✓(constraints)
SPARQL-like		✓	✓(high)	✓(triples)

1.2.2 Representational Heterogeneity of Entity-centric Data

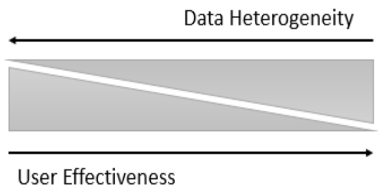
Natural language is a dynamic tool that may be used in generating, querying, and presenting entity data to interested users. Fig. 1.4 depicts the level of flexibility in regards to the generated patient data and the possible query choices of the doctor that lead to the following challenges:

Structural Heterogeneity: Different entity sources translate to different data structures/schemata. There is a plethora of data formats that is used ranging from text, attribute-value pairs, graphs etc. All these formats may also have different data models to represent them. Fig. 1.4 depicts a mixture of attribute-value pairs and triples (i.e. graphs).

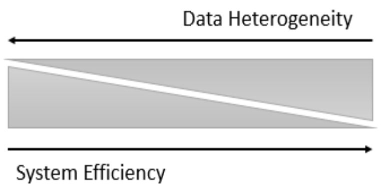
Semantic Heterogeneity: Different entity sources also translate to different data semantics. Semantics could have different levels of diversity depending on the word(s) used. Synonyms (same/related word), hyponyms (sub-word), hypernyms (super-word) and antonyms (opposing-word) [27] create a big collection of semantic relations of words that could be used by different sources. In Fig. 1.4 the words birthPlace and placeOfBirth are synonyms.

Conceptual and Contextual Diversity: Entities are complex things that could themselves refer to a wide range of different concepts and contexts. For example, entity information might be semantically closer in one context compared to another [28] (e.g. all blood pressure types in Fig. 1.4). Often this entity data may be integrated with other data [29] or enriched by external sources to provide richer and complementary contextual information, and situation awareness (e.g. the patient's medical history and living conditions in Fig. 1.4). The data could also apply to different domains making this diversity even more evident. Polysemy (multiple senses of a word) as well as ambiguity could create an environment of different meanings and possible interpretations.

It seems that the high volume of sources and the continuous, dynamic and high-velocity nature of IoT data will only make matters worse in creating a massive, heterogeneous data landscape that suffers from representational coupling [19]. These challenges affect the user and system side as a well (Fig. 1.6). "If everyone always agreed on what to call things, the user's word would be the designer's word would be the system's word...", stated George Furnas [30]. Users when left unguided will tend to use a great variety of words and structures to express queries to refer to the same thing. These words and structures may not match the words and structures used by the data in the IoT systems



(a) The more heterogeneous the data is, the more difficult it is for the users to create appropriate queries, and the more they are overloaded by massive and redundant information.



(b) The more heterogeneous and voluminous the data is, the more data abundance might exist, and the more inefficient the system becomes in terms of memory, power, and network interfaces.

Figure 1.6: Trade-off between data heterogeneity, user effectiveness, and system efficiency.

(schematic/structural and semantic coupling). Also, users come with different biases and knowledge backgrounds. "Facts is precisely what there is not, only interpretations", stated Friedrich Nietzsche [31]; therefore, it is unclear whether users completely understand their data needs or the different domains used in a plethora of systems in IoT [18]. The data heterogeneity will also create a load of information that may be unnecessary or redundant to the user. According to George Miller, people can only receive, process, and remember a limited amount of information at a time, otherwise, they get overloaded [32]. The creation of complex technical queries for more targeted information may prove too challenging for users, as aforementioned.

1.2.3 System Performance

As aforementioned, IoT architectures are characterised by large, dynamic, and high-velocity streams, devices with resource constraints, and real-time data analytics requirements [19]. The volume, heterogeneity, and abundance of data, described above, make any possible data analytics cumbersome in terms of memory, power, and network interfaces (Fig. 1.6b) through significant propagation, storage overheads of unnecessary data within a network, and slower processing time [33, 34]. This affects the system performance and hinders the systems from having the processing power to perform more important processes in higher layers or extract up-to-date information in real-time from large-scale dynamic data streams.

The challenges involving scalability, real-time communication, and resource constraints contributed to pushing the data analytics on the edge (Edge/Fog Computing) without the need to move data to centralised locations (e.g Cloud)

[3] as this will negatively impact the latency, processing time, and network overhead [35]. The integration with Event-based Systems [36] makes this solution more robust; however, suitable solutions are still needed that can also structure, annotate, share, and understand IoT data [37] in a quick, versatile, and resource-efficient manner [38].

1.2.4 Publish/Subscribe Limitations

Overall, Pub/Sub are ideal communication schemes for IoT since they abstract the users from the underlying data analytics and are scalable and distributed [39]. They are decoupled in space (publishers and subscribers do not need to know each other), time (publishers and subscribers do not need to be active at the same time), and synchronisation (publishers and subscribers are not blocked from performing other concurrent activities). These traits can ease the system challenges regarding scalability, real-time communication, and analysis requirements as well as resource constraints.

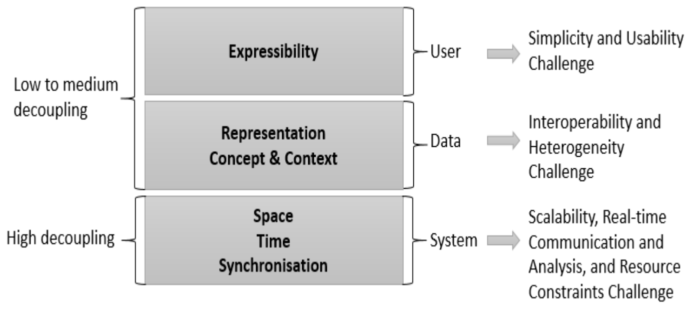

Low to medium decoupling
High decoupling
Expressibility
Representation Concept & Context
Space Time Synchronisation
User
Data
System
Simplicity and Usability Challenge
Interoperability and Heterogeneity Challenge
Scalability, Real-time Communication and Analysis, and Resource Constraints Challenge

Figure 1.7: Decoupling dimensions of Pub/Sub.

However, all of the aforementioned problems still apply. Fig. 1.7 illustrates the decoupling dimensions of Pub/Sub systems. The systems present low to medium decoupling when user expressibility is involved, that is constructing a subscription independent of representation, bias, data understanding, or technical expertise. This creates a limitation in overcoming the user challenges regarding simplicity and usability. The same decoupling level applies to data representations including conceptual and contextual diversity. This creates a limitation in overcoming the data challenges regarding interoperability and heterogeneity. The latter affects the system challenges, as well, since information redundancy may occur; therefore, even though Pub/Sub systems present high decoupling in system challenges, improvements need to take place.

1.3 Core Requirements and Research Questions

The core requirements and research questions of this thesis' work span among the perspectives of the user, data, and system.

1.3.1 Requirements

The requirements are the following:

- **R1: Usability**: refers to a system with high usability, which can be used easily by all users independent of representational coupling, query language expertise, system knowledge, bias, and background knowledge.
- **R2: User Expressibility**: refers to the ability of users to understand their data needs at a satisfying level and express them by simple subscriptions of high usability with minimal configuration settings.
- **R3: Data Expressiveness**: refers to a domain-agnostic system that can tackle interoperability and heterogeneity by providing rich notifications that contain conceptual and contextual diversity or high-level abstractions.
- **R4: User Effectiveness**: refers to a system that provides notifications of high quality according to the users' needs.
- **R5: Data Effectiveness**: refers to a system that provides redundancy-aware and expressive notifications of high quality according to the wide range of concepts and contexts.
- **R6: Efficiency**: refers to a system that is efficient in terms of memory, processing time, throughput, and scalability.

1.3.2 Research Questions

These requirements are formulated into the following research questions:

RQ1: Can a Pub/Sub be created that offers usability *(R1)* while addressing users' expressibility *(R2)* effectively *(R4)* and efficiently *(R6)*?

- **RQ1.1**: can a simple abstract representationally-decoupled subscription be defined that relies on expert and non-expert users alike *(R1)*?

- **RQ1.2**: can a subscription be defined that its notification will cover a range of different human interpretations independent of its complexity and with minimal configuration settings *(R2)*?

- **RQ1.3**: can a subscription be defined that its notification will not overwhelm the users with data overload *(R2)*?

- **RQ1.4**: can the satisfaction of users based on the received notifications be evaluated according to: (1) how well they address the users' different needs and interpretations *(R4)*?, (2) how much they reduce the information overload to the users *(R4)* and to the system *(R6)*?

RQ2: Can a Pub/Sub be created that offers expressiveness of heterogeneous data *(R3)* effectively *(R4, R5)* and efficiently *(R6)*?

- **RQ2.1**: can a methodology be defined for integrating data from multiple sources *(R3)*?

- **RQ2.2**: can an appropriate publication structure of integrated data be defined that is also understandable to the users *(R3)*?

- **RQ2.3**: can a methodology be defined for semantically abstracting integrated data and providing rich notifications with conceptual and contextual diversity or high-level abstractions independent of domain *(R3)*?

- **RQ2.4**: can the semantically-abstracted rich notifications be evaluated according to: (1) how well they cover the wide range of different concepts and contexts *(R5)*?, (2) how well they reduce information redundancy without sacrificing important information *(R5)*?, 3) how much they boost the system's performance *(R6)*, 4) how many dependencies are needed (e.g. domain experts, external ontologies, memory-heavy models etc.) *(R4, R5, R6)*?

1.4 Existing Approaches

The literature related to this work can be spread in a multitude of fields, analysed below (more details in Chapters 2, 4, 5, and 6):

Event-based Systems: These systems include Pub/Sub, Complex Event Processing (CEP) and stream processing systems [1, 16] that formulate their own architectures, data, and query models. Pub/Sub contain the best trade-off between system performance capabilities and simplicity. However, no existing scheme can completely overcome the aforementioned challenges. CEP are the most expressible systems, but they are complex with low usability and limited processing capabilities. Stream processing systems have the most advanced processing capabilities, but they have low usability and lack the system performance capabilities of the other systems.

Summarisation and Approximation: A trade-off between system performance and data or user quality can be achieved by data summarisation, approximation [40], or aggregation that reduce the data size, but maintain essential information or patterns of the original data [29]. These solutions are acceptable as long as the error is low [41]. Approaches involve *Time Series Approximation* [42–45], *Graph and Stream Approximation* [46–49], and *Entity Summarisation* [50–52]. The latter emphasises on entity-centric data and creates sophisticated summaries with conceptual and contextual diversity based on user importance, diversity, relevance, and popularity [53]. These approaches may be supervised, domain-dependent, static, and diverse in terms of effectiveness and system efficiency. Some are also bound by ontologies, thesauri, and taxonomies that are limited in terms of complexity and inefficiency, representational coupling, complex SPARQL-like queries, and domain dependency.

Sophisticated Pub/Sub Systems: Traditional Pub/Sub may have focused on single events; nevertheless, several approaches have tried to extend them to satisfy more expressive subscriptions and data information since then. Approaches involve *Diversity in Pub/Sub* [54–56] that produce diverse notifications to tackle redundant information, *Approximate Semantic Matching in Pub/Sub* [39, 57, 58] that try to resolve the rigidness of subscription models by proposing approximate subscriptions and matchers, and *Semantic Engines in Pub/Sub* [59, 60] that integrate data from heterogeneous sources or semantically enrich data with information deriving from external sources. These approaches have a range of advantages and disadvantages related to data expressiveness, usability, dependency on ontologies, thesauri and taxonomies, and they do not apply to entity-based publications that contain rich conceptual and contextual information.

Conceptual and Contextual Awareness: The attention has been drawn to the Semantic Web (Web of Things) in order to create interoperable and context-aware IoT solutions [3] that provide rich information to the users in a conceptual or contextual level [37, 61]. It is understood that if the data is transformed into a machine-interpretable one, then, problems of data management and sharing will be resolved. Nevertheless, semantic data is often depicted in rich and complex graphs, and described by models like Resource Description Framework (RDF) that themselves pose many challenges. Approaches involve *Semantic Rule Engines and Languages* [62–64] that propose systems or languages that contain rules for semantic reasoning and interoperability, and *Machine Learning and Approximation* [42, 65, 66] that use advanced data analytics in order to infer high-level abstractions of data. These approaches have a range of advantages and disadvantages related to dependency on ontologies, domain-specific and representationally-coupled reasoning rules, complexity, and efficiency.

1.5 Overview of the Proposed Approach

In order to address the challenges, requirements and research questions analysed above, this thesis proposes an entity-centric Pub/Sub summarisation system. "To name a thing is easy: the difficulty is to discern it before its appearance", stated Pierre-Joseph Proudhon [67]. In order to turn this proposal from an abstractive notion to a tangible system, several components were implemented, which are illustrated in Fig. 1.8.

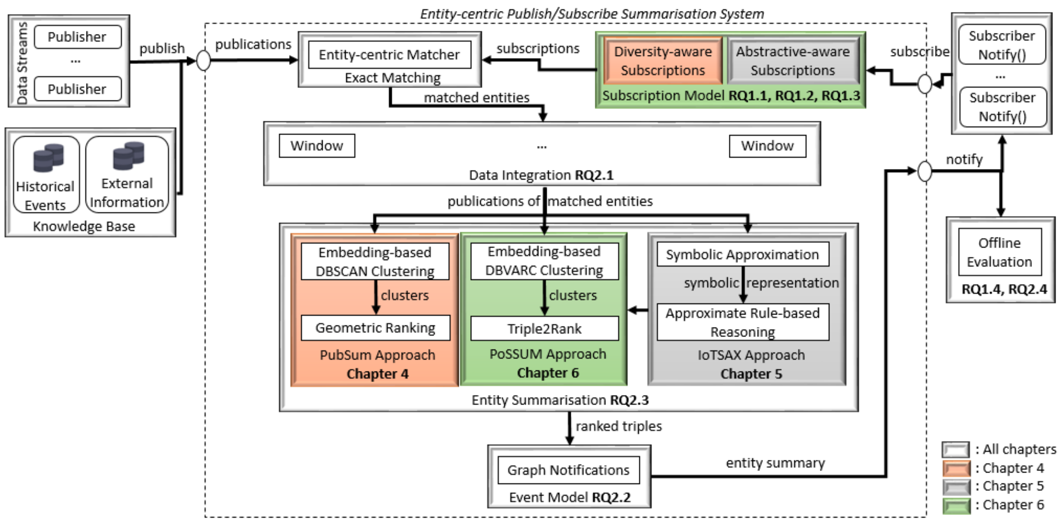


Figure 1.8: The proposed entity-centric Pub/Sub summarisation system.

The proposed system is split into the following main components (more details in Chapter 3):

Exact Matching: The entities included in both subscriptions and publications form a match.

Data Integration and Event Model: The events deriving from sensor streams and external knowledge bases containing historical or additional entity information are efficiently integrated in entity-centric windows. In this way, they are not observed in isolation and are analysed in batch or incrementally. The integrated events and the notifications sent to the users after the analysis form conceptually rich graphs, which are understandable to the subscribers.

Subscription Model: Users should create subscriptions of high usability independent of representational coupling, query language expertise, system knowledge, bias, and background knowledge. Also, the subscription model should be flexible enough to express the users' different data needs/interpretations via simplicity and minimal configuration settings as well as the users' lack of desire for information overload. The proposed diversity-aware and abstractive-aware subscriptions expect the user only to be aware of the entity name they are interested in and to provide minimal parameters related to windows, summary type, and level of filtering. In this way, subscribers are presented with entity summaries that cover a wide range of concepts catering for different user interpretations and without being overwhelmed, while providing highly usable and representationally-decoupled subscriptions.

Entity Summarisation: The volume, velocity, and heterogeneity of entity-centric data result in information overload that contains multiple semantics and redundancy. Extractive and abstractive approaches are proposed for the summarisation of entities, that is domain-agnostic and interoperable approaches that semantically abstract the integrated data and provide rich notifications with conceptual and contextual diversity or high-level abstractions. This improves interoperability and notification expressiveness as well as system performance for further network propagations. Three approaches are proposed: 1) dynamic diverse entity summarisation of Linked Data with the use of a range of windowing policies, embedding-based density clustering, and a geometric-based top-k ranking, 2) dynamic abstractive entity summarisation for numerical data that is based on symbolic approximation and approximate rule-based reasoning, and 3) dynamic abstractive and extractive diverse entity summarisation for enriched data that involves both numerical and Linked Data by proposing a novel partly-incremental and parameter-free conceptual clustering based on embedding models and variance, and a contextual-based top-k ranking. More details of the corresponding approaches are given in Chapters 4, 5, and 6.

Offline Evaluation: The entity summaries (notifications) are evaluated based on their user and data effectiveness, and system efficiency. Multiple metrics are formulated or proposed related to quality, redundancy-awareness, expressiveness, size, processing time, memory, throughput, and scalability.

An example of the proposed entity-centric Pub/Sub summarisation system is illustrated in Fig. 1.9.

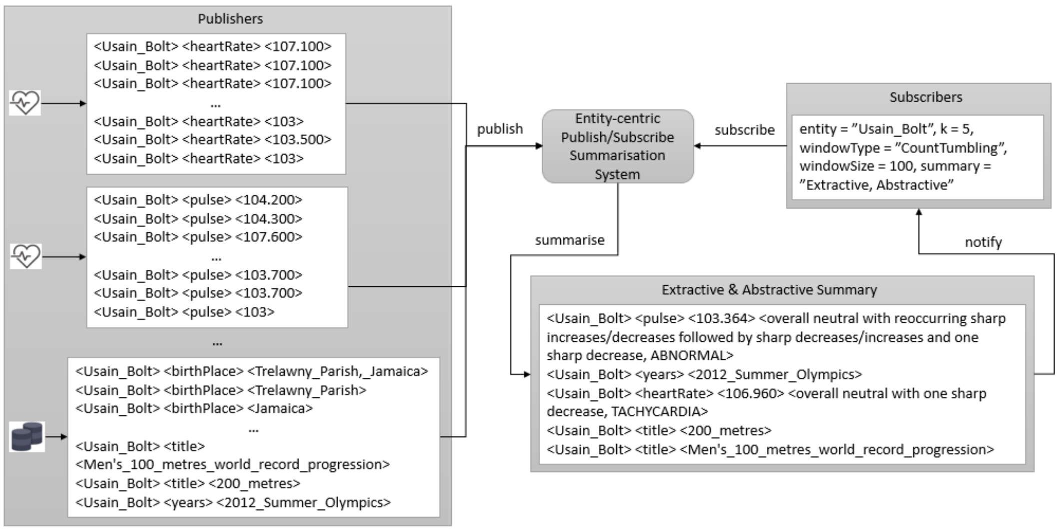

Publishers
<Usain_Bolt> <heartRate> <107.100>
<Usain_Bolt> <heartRate> <107.100>
<Usain_Bolt> <heartRate> <107.100>
...
<Usain_Bolt> <heartRate> <103>
<Usain_Bolt> <heartRate> <103.500>
<Usain_Bolt> <heartRate> <103>
<Usain_Bolt> <pulse> <104.200>
<Usain_Bolt> <pulse> <104.300>
<Usain_Bolt> <pulse> <107.600>
...
<Usain_Bolt> <pulse> <103.700>
<Usain_Bolt> <pulse> <103.700>
<Usain_Bolt> <pulse> <103>
...
<Usain_Bolt> <birthPlace> <Trelawny_Parish,_Jamaica>
<Usain_Bolt> <birthPlace> <Trelawny_Parish>
<Usain_Bolt> <birthPlace> <Jamaica>
...
<Usain_Bolt> <title> <Men's_100_metres_world_record_progression>
<Usain_Bolt> <title> <200_metres>
<Usain_Bolt> <years> <2012_Summer_Olympics>
publish
Entity-centric Publish/Subscribe Summarisation System
subscribe
Subscribers
entity = "Usain_Bolt", k = 5, windowType = "CountTumbling", windowSize = 100, summary = "Extractive, Abstractive"
summarise
notify
Extractive & Abstractive Summary
<Usain_Bolt> <pulse> <103.364> <overall neutral with reoccurring sharp increases/decreases followed by sharp decreases/increases and one sharp decrease, ABNORMAL>
<Usain_Bolt> <years> <2012_Summer_Olympics>
<Usain_Bolt> <heartRate> <106.960> <overall neutral with one sharp decrease, TACHYCARDIA>
<Usain_Bolt> <title> <200_metres>
<Usain_Bolt> <title> <Men's_100_metres_world_record_progression>

Figure 1.9: An example of the proposed entity-centric Pub/Sub summarisation system.

1.6 Research Methodology

The research, conducted in this thesis, followed the research methodology described below:

- Conduct literature review in Even-based systems, IoT, knowledge graphs, diversity, summarisation, approximation, and ranking and understand the works' successes, limitations, and assumptions.
- Formulation of gaps and limitations of the literature review into challenges, requirements, and research questions for a usable and expressive entity-centric Pub/Sub summarisation system.

- Focused literature review and gap analysis based on requirements and research questions.
- Conceptualisation and formalisation of the framework, components, and functions involved in the usable and expressive entity-centric Pub/Sub summarisation system.
- Evaluation design based on the following:
  - Construction of an evaluation dataset based on real-world sensor and Linked Data related to the domains of Healthcare and Smart Cities. The datasets' characteristics involve heterogeneity in data types, semantics, concepts, and contexts. Suitable pre-processing occurred regarding data structures, missing values, and noise.
  - Construction of two ground truths (relevance and conceptual) based on semantically extending original datasets with the use of thesauri and ontologies, and observing the contextual coherence, direct word links, and sub-categories in taxonomies.
  - Identification and formulation of evaluation metrics regarding user effectiveness. Metrics involved: summaries' agreement and diversity consensus among judges, and quality of summaries based on judges.
  - Identification and formulation of evaluation metrics regarding data effectiveness. Metrics involved: F-score, redundancy-aware F-score, conceptual clustering F-score, and approximation error.
  - Identification and formulation of evaluation metrics regarding efficiency. Metrics involved: window processing time, end-to-end latency, size reduction in messages, compression space saving, memory, and throughput.
  - Identification of baselines and adaptation to dynamic online systems.
- Implementation of the proposed framework, components and functions involved in the usable and expressive entity-centric Pub/Sub summarisation system.
- Evaluation of the system, analysis of the results, and conclusions.

1.7 Contributions

The contributions of this thesis are the following:

- Identification of a set of challenges, requirements, and research questions for IoT environments and event-based systems with the support of the literature.
- A new Pub/Sub scheme; an entity-centric Pub/Sub system that provides entity summaries for numerical and Linked Data. Its goal is to address usability, user expressibility, data expressiveness, user and data effectiveness, and system efficiency.
- A user-friendly subscription model for receiving abstractive and/or diverse extractive entity summaries.
- A novel evaluation methodology including: 1) a real-world heterogeneous dataset related to Healthcare and Smart Cities, 2) adaptation and formulation of ground truths, 3) baselines and adaptation to dynamic online systems, 4) adaptation and formulation of evaluation metrics, and 5) extensive evaluation by examining the trade-off among user effectiveness, data effectiveness, and system efficiency by using a range of windowing policies.
- Introduce PubSum, a dynamic diverse summarisation methodology for heterogeneous Linked Data streams that is based on an embedding-based density clustering, and a geometric-based top-k ranking. The results include: 1) promising user effectiveness, 2) up to 0.95 and 69.3% redundancy-awareness and duplication reduction, respectively, 3) 6 times less latency and 3 times less memory, 4) up to 92% message reduction, and 5) throughput from 833 to 1,005 events/second. This work is published in [68–70].
- Introduce IoTSAX, a dynamic abstractive summarisation methodology for heterogeneous numerical entity graph streams that is based on an enhanced dynamic symbolic approximation and approximate rule-based reasoning. The results include: 1) 2 to 3 times less approximation error and up to 0.87 reasoning F-score, 2) 2 to 3 times slower latency, 3) throughput from 13.231 to 97.393 events/second, 4) up to 98% message reduction, and 5) compression space-saving from 71.75% to 94.99%. This work is published in [71] (early access).
- Introduce PoSSUM, a dynamic extractive and abstractive diverse summarisation methodology for heterogeneous numerical and Linked Data entity graph streams, alike, that is based on an embedding-based and density-based conceptual clustering, and a contextual-based top-k ranking. The results include:

1) up to 80% data diversity user desire and best summary quality for more than half of the entities, 2) best conceptual clustering F-score from 0.69 to 0.83 and up to 0.95 redundancy-awareness, 3) up to 99% message reduction, 4) less clustering processing time, scoring, and memory consumption, and 5) comparable latency and throughput. This work is published in [72] (in press).

1.8 Thesis Outline

The rest of this thesis is organised as follows:

- *Chapter 2 - Background and Problem Analysis*: This chapter motivates the problem for applications in the domains of Healthcare and Smart Cities. It provides an analysis of the challenges and limitations of existing systems with regards to the requirements and research questions. The necessary background is also included regarding Pub/Sub, CEP, and data stream processing systems and their traits, ontologies, linguistic resources, and deep neural network embedding models, entity summarisation and data approximation.

- *Chapter 3 - Entity-centric Publish/Subscribe Summarisation System*: This chapter presents a high-level view of the proposed Pub/Sub system regarding approaches, components, and functions and how they map to challenges, requirements, and research questions.

- *Chapter 4 - Dynamic Diverse Entity Summarisation of Linked Data*: This chapter presents the PubSum approach along with experiments and results. Background and related work are also provided.

- *Chapter 5 - Dynamic Abstractive Summarisation of Numerical Data*: This chapter presents the IoTSAX approach along with experiments and results. Background and related work are also provided.

- *Chapter 6 - Dynamic Entity Summarisation of Enriched Data*: This chapter presents the PoSSUM approach along with experiments and results. Background and related work are also provided.

- *Chapter 7 - Conclusions and Future Work*: This chapter concludes the thesis and discusses contributions, limitations, and future work.

1.9 Associated Publications

Different aspects of this work were disseminated in different publications.

Journal publications:

- Pavlopoulou, N. and Curry, E., 2021. PoSSUM: An Entity-centric Publish/Subscribe System for Diverse Summarisation in Internet of Things. ACM Trans. Internet Technol. (in press)
- Pavlopoulou, N. and Curry, E., 2021. IoTSAX: A Dynamic Abstractive Entity Summarisation Approach with Approximation and Embedding-based Reasoning Rules in Publish/Subscribe Systems. IEEE Internet of Things Journal. (early access)
- Pavlopoulou, N. and Curry, E., 2020. Dynamic diverse summarisation in heterogeneous graph streams: a comparison between thesaurus/ontology-based and embeddings-based approaches. Int. J. Graph Comput, 1(1), pp.23-39

Conference/Workshop publications:

- Pavlopoulou, N. and Curry, E., 2019, September. Using embeddings for dynamic diverse summarisation in heterogeneous graph streams. In 2019 First International Conference on Graph Computing (GC) (pp. 5-12). IEEE.
- Pavlopoulou, N. and Curry, E., 2019. Towards a window-based diverse entity summarisation engine in publish/subscribe systems. In Proceedings of EYRE 19: 2nd International Workshop on Entity Retrieval. ACM.

1.10 Overall Map

The overall map among requirements, research questions, contributions, chapters, and publications is presented in Table 1.2 and Table 1.3.

Table 1.2: Overall map among requirements, research questions, contributions, chapters, and publications

Requirements	Research Questions	Contributions		Chapters	Publications
		Component	Solution		
R1 - Usability	RQ1.1	Subscription Model	Representationally-decoupled subscriptions with minimal parameters and a-priori knowledge only of the entity name	3	All
R2 - User Expressibility	RQ1.2, RQ1.3	Subscription Model	Expressive diversity-aware and abstractive-aware subscriptions with desired windowing policy and summary size	3	All
R3 - Data Expressiveness	RQ2.1	Data Integration	Integration of data in windows	4, 5, 6	All
	RQ2.2	Event Model	Entity graphs that can represent conceptually rich information	3	All
	RQ2.3	Entity Summarisation	PubSum, a dynamic diverse summarisation methodology for heterogeneous Linked Data streams that is based on an embedding-based density clustering and a geometric-based top-k ranking	4	EYRE2019, GC2019, IJGC2020
			IoTSAX, a dynamic abstractive summarisation methodology for heterogeneous numerical entity graph streams that is based on an enhanced dynamic symbolic approximation and an approximate rule-based reasoning	5	IOT2021
			PoSSUM, a dynamic extractive and abstractive diverse summarisation methodology for heterogeneous numerical and Linked Data entity graph streams, alike, that is based on an embedding-based and density-based conceptual clustering, and a contextual-based top-k ranking	6	TOIT2021
R4 - User Effectiveness	RQ1.4	Offline Evaluation	A real-world heterogeneous dataset related to Healthcare and Smart Cities	5, 6	IOT2021, TOIT2021
			• adaptation of ground truth • adaptation and formulation of evaluation metrics	4, 6	EYRE2019, GC2019, IJGC2020, TOIT2021
			Baselines and adaptation to dynamic online systems	4, 5, 6	All
			PubSum: • promising user effectiveness PoSSUM: • up to 80% data diversity user desire and best summary quality for more than half of the entities	4, 6	EYRE2019, GC2019, IJGC2020, TOIT2021

Table 1.3: Overall map among requirements, research questions, contributions, chapters, and publications (table continued)

Requirements	Research Questions	Contributions		Chapters	Publications
		Component	Solution		
R5 - Data Effectiveness	RQ2.4	Offline Evaluation	• a real-world heterogeneous dataset related to Healthcare and Smart Cities • formulation of ground truths	5, 6	IOT2021, TOIT2021
			• adaptation and formulation of evaluation metrics • baselines and adaptation to dynamic online systems PubSum: • up to 0.95 and 69.3% redundancy-awareness and duplication reduction, respectively IoTSAX: • 2 to 3 times less approximation error and up to 0.87 reasoning F-score PoSSUM: • best conceptual clustering F-score from 0.69 to 0.83 and up to 0.95 redundancy-awareness	4, 5, 6	All
R6 - Efficiency	RQ1.4, RQ2.4	Pub/Sub, Data Integration, Entity Summarisation	A new Pub/Sub scheme; an entity-centric Pub/Sub system that integrates data in windows and provides entity summaries for numerical and Linked Data	3	All
		Offline Evaluation	A real-world heterogeneous dataset related to Healthcare and Smart Cities	5, 6	IOT2021, TOIT2021
			• adaptation of evaluation metrics • baselines and adaptation to dynamic online systems PubSum: • 6 times less latency and 3 times less memory • up to 92% message reduction • throughput from 833 to 1,005 events/second IoTSAX: • 2 to 3 times slower latency • throughput from 13.231 to 97.393 events/second • up to 98% message reduction • compression space-saving from 71.75% to 94.99% PoSSUM: • up to 99% message reduction • less clustering processing time, scoring, and memory consumption • comparable latency and throughput	4, 5, 6	All

Chapter 2

Background and Problem Analysis

2.1 Introduction

Internet of Things (IoT) is a technological trend that is aspiring to be one of the leading ones in the future in order to benefit several applications, including Healthcare and Smart Cities. Nevertheless, a complex environment like IoT is bound to pose a number of challenges that are linked to data, users, and system performance.

In IoT, interoperability [73] is a significant issue. Applications like Healthcare and Smart Cities are envisioned to be deployed massively collecting data from a large number of devices belonging to different manufacturers or service providers. These providers follow different data representation standards, that is data formats or schemata and semantics. This creates a high level of heterogeneity that results in representational coupling that hinders seamless connectivity, and redundant information related to duplication or similar concepts and contexts. These issues, along with the high data velocity, burden the users and the system performance with the propagation of unnecessary data in resource-constrained environments that can overwhelm the users.

As the world becomes more complex, the end-user needs become more challenging. Often users are interested in turning raw data to information to valuable knowledge and to eventually insight/wisdom [15, 37]. They are interested in real-world entities (e.g. things, places, people etc.) and their high-level states [74]. For example, a doctor should check a patient's medical history, apart from real-time health readings, to diagnose an illness as a deviation in a single health indicator is not enough [75, 76]. This necessitates the sensors' data integration as well as enrichment with external sources to provide conceptually and contextually richer information. Enriched data along with the volume, velocity, and heterogeneity of IoT data may lead to information abundance that will overwhelm the users. Meaningful content or high-level abstractions should be extracted that cover a wide range of information chunks and provide perceptions understood by users [29, 37]. Nevertheless, end-users should be abstracted from this procedure and IoT systems should provide automatic ways of the aforementioned data needs for the end-users' convenience.

Another issue is the end-users' ability to express their needs through queries based on their understanding and expertise. Day-to-day end-users, which have no knowledge in query languages and were not involved in the design of the system so they have no information regarding the system's architecture, data representations or the content provided by the available sensors, should be abstracted from the underlying complexity of IoT environments and provide simple representationally-decoupled queries of high expressiveness without sacrificing usability [20]. For example, SPARQL-like queries are highly expressive requests, but also complex for non-expert users and representationally-coupled (semantic and schematic mismatch) [26]. Keywords, on the other hand, are simple and representationally-decoupled queries without needing a-priori knowledge of data content, but they may return unnecessary information as the user's intent is not clear or targeted enough due to the lack of filters (low query expressiveness). Data heterogeneity will only make matters worse when users need to provide clear and representationally-coupled queries to satisfy their needs.

The system requirements are continuously evolving according to the advances in the processing technologies [77]. There comes a point where the data volume, velocity, variety, and analytics needs are so complex and advanced that systems should perform better through either networking/hardware solutions or efficient, lightweight, and optimised processing. Edge Computing pushes data analytics closer to the devices, but these are resource-constrained and cannot support the power needs of advanced data analytics unless optimised solutions are found. For example, in Healthcare applications, energy is harvested from the human body to boost wearable sensors [77]. Healthcare is of time-critical nature needing communication technologies with high bandwidth and availability [75].

Another issue is scalability. A massively deployed network, like Smart Cities, would contain a multitude of dynamic sensors that create data of diverse velocities as well as several interested end-users. This sensor and end-user dynamism along with continuous data will create a complex architecture that needs to process on-the-fly an increased volume of data that may constantly be updated or enriched with new information (e.g. new sensors deployed). Communication technologies should efficiently store, manage, and process all parties involved.

Event-based systems are suitable communication paradigms for dynamic large scale applications that could ease the aforementioned system requirements. Nevertheless, no existing system can address all of the aforementioned IoT challenges. At the same time, Semantic Web technologies along with Data Stream Processing have addressed the interoperability need and the extraction of conceptually and contextually richer information. Resources like ontologies, knowledge bases, thesauri, and dictionaries have been used for realising the IoT data representation, semantic annotation, reasoning, interlinking, and high-level abstraction needs of the users [37, 74]. SPARQL-like queries have also been proposed to provide higher query expressiveness to the users for richer data exploration. Nevertheless, these linguistic resources have their own limitations and other approaches regarding word embedding models could be explored. Finally, approaches regarding summarisation and approximation have been explored to assist with resource constraints and system efficiency through extractive or abstractive methodologies. These approaches are characterised by a trade-off between system performance and data or user quality.

The rest of the chapter is as follows: In Section 2.2 motivational scenarios are given for the domains of Healthcare and Smart Cities, whereas Section 2.3 describes the challenges involved in data, user, and system. Section 2.4 analyses the event-based systems and data stream processing systems, and their limitations with regards to the core requirements and research questions of the thesis' work. Section 2.5 explores background information regarding entities and knowledge graphs as well as linguistic resources and their limitations compared to superior deep neural network embedding models, while Section 2.6 analyses the role of data summarisation and approximation in IoT with an emphasis on entity summarisation. Section 2.7 presents an overall discussion, whereas Section 2.8 concludes and summarises the chapter.

2.2 Motivational Scenarios

IoT could apply to and benefit a multitude of applications, including Healthcare and Smart Cities.

2.2.1 Healthcare or Medical IoT

Health is one of the most important aspects of a person's life. Nowadays, factors like human error, slow response, inaccurate information, healthcare cost, prolonged or frequent diseases, and even the occurrence of pandemics have contributed to a worldwide demand for a more sustainable, time, and cost-efficient healthcare service that can act effectively [14]. The current healthcare model should shift from a medical facilities based (e.g. doctor's private clinic, hospital etc.) to a patient-based one (e.g. home monitoring, mobile applications monitoring etc.). In this way, not only diseases will be monitored more efficiently and effectively, but forecasting may even take place that will lead to prevention and taking prompt actions. This new model will assist patients and healthcare providers, alike.

Healthcare IoT or Internet of Medical Things will be one of the leading technologies in the future [15, 78]. IoT, in this case, will create a unified communication network of healthcare staff, patients, devices, medicine/drugs etc. for constant health monitoring, diagnosis, and emergency intervention, if need be. Specifically, the aim is to monitor, in real-time, devices using sensing (detecting) and actuating (activating) capacities that are put inside or outside the human body. By observing them individually, collectively or historically (i.e. medical history of patients), readings or their inter-relationship could lead to fast, easy, active, and personalised health monitoring of patients that may have not even left their own premises. The possible use cases are enormous [3], ranging from fall detection among elderly or disabled people, temperature monitoring of hospitals' fridges that store vaccines or medicines, logistics regarding emergency transportation of a patient to a hospital when an alert is detected, and financial management once the treatment is completed, to name a few. Nevertheless, security, privacy, trust, and government regulations will always pose significant challenges in health-related applications [79]. People, healthcare providers, and governments will have to adjust to this new technological era.

Existing approaches have proposed Healthcare IoT architectures [75] including cases of pandemics [80], data processing approaches including Machine Learning based ones [77], and even health data fusion solutions in Complex Event Processing systems (CEP) [76]. However, there is no existing standardised architecture that could overcome all of the aforementioned IoT challenges related to users, data, and systems and that could not only apply to Healthcare applications but to generalised domain-agnostic applications.

Scenario 1: Healthcare Use Case

The thesis explores the following scenario for Healthcare:

A doctor (R1) is very busy observing multiple patients daily (R6), which are either ill enough to be in the hospital or healthy enough to be at home and monitored remotely in order not to further strain the hospital resources. The patients are connected to several body sensors (R6) through embedded sensors in hospital equipment, wearable sensors or mobile applications that measure in real-time (R6) their physiological conditions like pulse, heart beat rate, etc. These body sensors could belong to different manufacturers (R3) that install them in different hospital rooms or they could have been purchased by the patients themselves at home. Apart from the sensor readings, information exists regarding the medical history and living conditions of the patients (R3). The doctor is not interested in explicit readings from a patient (R2) and wants a combination of the available medical data from several data sources in order to have a more sufficient picture (R3). Nevertheless, due to the high amount of patients and the valuable time of the doctor, one does not want to manually fuse all of the available medical information or to be overwhelmed by viewing its high volume (R1, R2). The doctor is interested in a summarised view (R1-R6) of how the patient's health status is in order to know if one needs to intervene further by visiting them in the hospital or booking an appointment with them or to detect any possible epidemic outbreaks with patients that are in different hospital rooms. The use case is illustrated in Fig. 2.1, where sensors generate timestamped information records ranging from the patient's health sensors to external static sources deriving from the patient's home or the hospital. A simple summarised health status of a patient is presented.

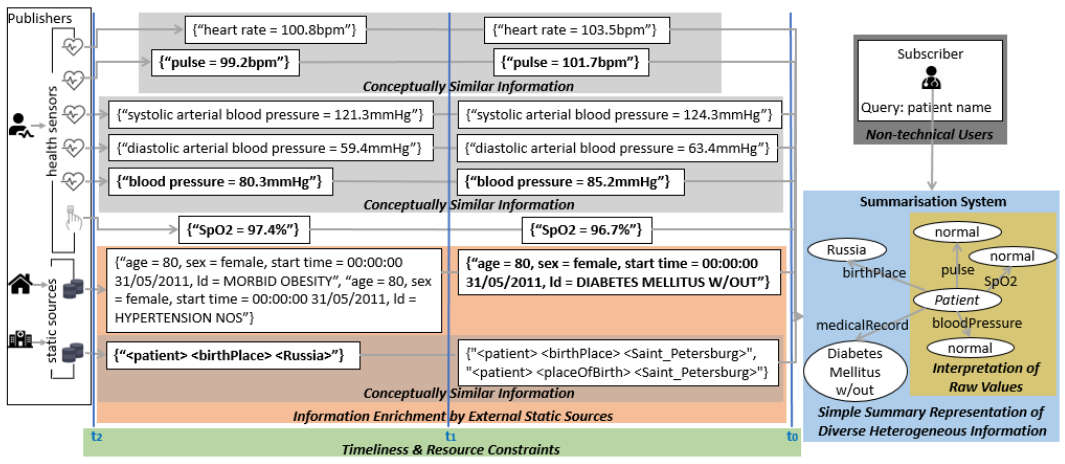

Publishers
health sensors
static sources
{"heart rate = 100.8bpm"}
{"heart rate = 103.5bpm"}
{"pulse = 99.2bpm"}
{"pulse = 101.7bpm"}
Conceptually Similar Information
{"systolic arterial blood pressure = 121.3mmHg"}
{"systolic arterial blood pressure = 124.3mmHg"}
{"diastolic arterial blood pressure = 59.4mmHg"}
{"diastolic arterial blood pressure = 63.4mmHg"}
{"blood pressure = 80.3mmHg"}
{"blood pressure = 85.2mmHg"}
Conceptually Similar Information
{"SpO2 = 97.4%"}
{"SpO2 = 96.7%"}
{"age = 80, sex = female, start time = 00:00:00 31/05/2011, Id = MORBID OBESITY", "age = 80, sex = female, start time = 00:00:00 31/05/2011, Id = HYPERTENSION NOS"}
{"age = 80, sex = female, start time = 00:00:00 31/05/2011, Id = DIABETES MELLITUS W/OUT"}
{"<patient> <birthPlace> <Russia>"}
{"<patient> <birthPlace> <Saint_Petersburg>", "<patient> <placeOfBirth> <Saint_Petersburg>"}
Conceptually Similar Information
Information Enrichment by External Static Sources
t2
t1
t0
Timeliness & Resource Constraints
Subscriber
Query: patient name
Non-technical Users
Summarisation System
Russia
birthPlace
normal
pulse
normal
SpO2
Patient
medicalRecord
bloodPressure
normal
Diabetes Mellitus w/out
Interpretation of Raw Values
Simple Summary Representation of Diverse Heterogeneous Information

Figure 2.1: Healthcare use case.

2.2.2 Smart Cities

Cities are complex entities that involve multiple entities themselves including people, buildings, cars etc. It is often difficult to coordinate all different entities and get the information needed whether one is a citizen, tourist, governing body, or an industrial organisation [12]. The current model of cities should shift from an individually-oriented one to a collectively-oriented one, where a unified network of entities could create a data landscape that when analysed could provide valuable information and insight to interested users.

A collectively-oriented model of cities could be accomplished with the aspiration of Smart Cities [9]. IoT, in this case, will create a unified communication network of people, buildings, roads, cars etc. for sensor monitoring with the aim to contribute to a vast range of information capabilities that may even lead to the prevention of upcoming crisis. For example, by monitoring, in real-time, devices transmitting critical environmental parameters such as temperature, humidity, pressure etc., a smart city could issue a heat warning to vulnerable citizens through notifications. Sensors in a smart building could detect lights or air-conditioning used in empty rooms and actuators could switch them off for energy efficiency [78]. Other use cases include traffic status, noise pollution, air pollution, chemical leakage from factories detection in rivers, radiation levels from nuclear power stations detection, earthquakes, flooding, fire detection, and avalanche prevention [3], to name a few. By far one of the most interesting applications would be the fully autonomous cars [81] that will not only resolve issues like traffic congestion, but will significantly contribute to the decrease of car accidents that endanger people's lives. Although there is a plethora of Smart Cities benefits, challenges like the complexity of a city, lack of infrastructure, growing population, government regulations, user awareness of endless possibilities, and trust of new technologies by citizens will need to be extensively addressed in the coming years.

Several smart cities currently exist, on a small scale, targeting different applications (e.g. water management, traffic control, environmental pollution etc.). The cities are situated in all parts of the world [82] with Barcelona, Amsterdam, Copenhagen, London, New York, Boston, Singapore, Hong Kong, and Dubai being the most prominent ones [83]. Several projects have also contributed to finding sustainable solutions for smart city applications with Waternomics [84], CityPulse [85], SmartSantander [86], OpenIoT [87], Spitfire [74], and PLAY [88] being one of the most prominent ones. All of these frameworks have been created by different communities, with different technologies and applications in mind. Each has its own advantages and disadvantages when it comes to the aforementioned IoT challenges related to users, data, and systems, leading to the lack of a standardised domain-agnostic architecture.

Scenario 2: Smart Cities Use Case

The thesis explores the following scenario for Smart Cities:

A real estate agent (R1) helps customers to buy, sell, or rent commercial or residential smart properties. Smart properties are embedded with several sensors (R6) that measure in real-time (R6) internal or external conditions like the property's temperature, humidity, etc. in the kitchen room or the area's/city's CO_2 emissions, traffic etc. in which the property is situated. These sensors relate to several properties that are located in different regions or even different complexes within a building; therefore, they could belong to different manufacturers (R3). Apart from the sensor readings, information exists regarding the region in which the properties are situated like social events happening nearby that may be different or similar based on the proximity of the properties (R3). The real estate agent is interested in summarised information (R1-R6) regarding a property like its energy consumption or the popularity of the nearby social events so that one is aware of the property value to potential buyers or tenants and to be able to compare the value of different properties in different regions. The use case is illustrated in Fig. 2.2, where sensors generate timestamped information records ranging from spatially-nearby kitchen sensors to external static sources deriving from the property or the venues in the nearby region. A simple summarised information status regarding a property is presented.

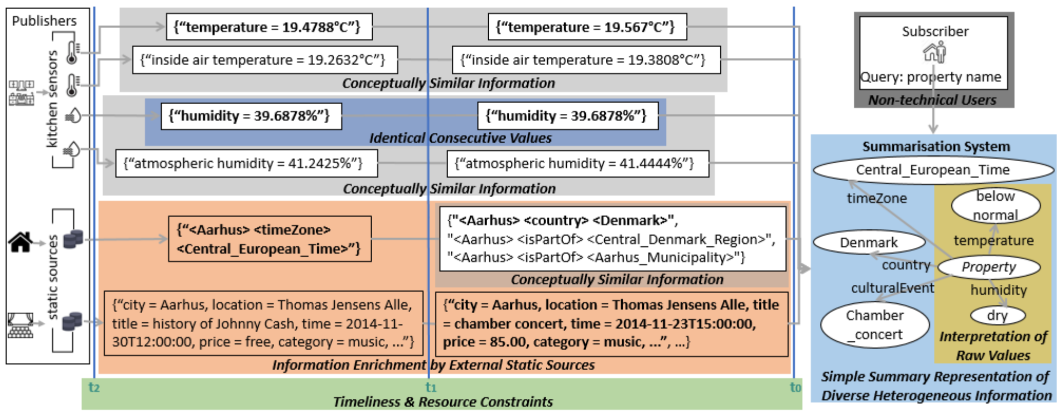

Publishers
kitchen sensors
static sources
{"temperature = 19.4788°C"}
{"temperature = 19.567°C"}
{"inside air temperature = 19.2632°C"}
{"inside air temperature = 19.3808°C"}
Conceptually Similar Information
{"humidity = 39.6878%"}
{"humidity = 39.6878%"}
Identical Consecutive Values
{"atmospheric humidity = 41.2425%"}
{"atmospheric humidity = 41.4444%"}
Conceptually Similar Information
{"<Aarhus> <timeZone> <Central_European_Time>"}
{"<Aarhus> <country> <Denmark>", "<Aarhus> <isPartOf> <Central_Denmark_Region>", "<Aarhus> <isPartOf> <Aarhus_Municipality>"}
Conceptually Similar Information
{"city = Aarhus, location = Thomas Jensens Alle, title = history of Johnny Cash, time = 2014-11-30T12:00:00, price = free, category = music, ..."}
{"city = Aarhus, location = Thomas Jensens Alle, title = chamber concert, time = 2014-11-23T15:00:00, price = 85.00, category = music, ...", ...}
Information Enrichment by External Static Sources
t2
t1
t0
Timeliness & Resource Constraints
Subscriber
Query: property name
Non-technical Users
Summarisation System
Central_European_Time
timeZone
below normal
temperature
Denmark
country
Property
culturalEvent
humidity
Chamber _concert
dry
Interpretation of Raw Values
Simple Summary Representation of Diverse Heterogeneous Information

Figure 2.2: Smart Cities use case.

2.3 Challenges

The challenges of the aforementioned motivational scenarios are split into data, user, and system related.

2.3.1 Data Challenges

These challenges involve heterogeneity, redundancy, and data enrichment.

Heterogeneity: The IoT data contains a massive amount of various patient/property information coming from multiple sensors that belong to different manufacturers; therefore, the data is characterised by different representations (schemata and semantics) (R3). The subscribers would like to be abstracted (R2) by the hurdles this data heterogeneity imposes regarding data sharing, processing, and communication among sensors and would expect a notification that covers as much unique conceptually and contextually diverse patient/property information as possible in a common representational format (R3-R5).

Redundancy: Redundancy (R3) is caused by: 1) Duplication, which occurs due to frequent sampling rates that generate identical data from sensors that have unchanged states for time periods [89] (e.g. Fig. 2.2: humidity = 39.6878%), 2) Conceptual similarity, which occurs due to heterogeneous data deriving from multiple sensors regarding the same thing or sensors that are located nearby (e.g. Fig. 2.1: "heart rate" vs "pulse", the different types of blood pressure with one of them being the mean value, "birthPlace" vs "placeOfBirth" and Fig. 2.2: "temperature" vs "inside air temperature", "humidity" vs "atmospheric humidity", country vs part of region or municipality). The redundancy will result in voluminous unnecessary data that will overwhelm the subscribers (R2) and impose resource limitations on the processing system (R4, R5).

Data Enrichment: The raw sensor health/property data by single sources (e.g. Fig. 2.1: the patient's heart rate, pulse, blood pressure, SpO2 and Fig. 2.2: room's temperature, humidity) will not lead to any satisfactory notifications for the subscribers (R2). An automatic combination of raw data from multiple sources and enrichment of data from external static sources (e.g. Fig. 2.1: living conditions like placeOfBirth, medical history like previous visits in hospital and Fig. 2.2: cultural events nearby, region/location information) will lead to a notification with complementary contextual information [26] concerning a patient/property (R3).

2.3.2 User Challenges

These challenges involve high-level interpretation, user-friendly data representation and non-technical users.

High-level Interpretation: The subscribers are not interested in numerical sensor data that contain specific values (e.g. Fig. 2.1: heart rate = 100.8bpm and Fig. 2.2: temperature = 19.567°C) as they do not depict a meaningful message (R2). The subscribers are interested in a summarised view (R3-R5) of multiple patient/property data including numerical one or automatic interpretations of data (e.g. Fig. 2.1: a patient has tachycardia and in Fig. 2.2: temperature is warm with slight fluctuations) that will help them infer their own knowledge and high-level interpretations (e.g. patient is in critical condition or property is of high quality).

User-friendly Data Representation: The notification that contains heterogeneous information that is coming from multiple real-time and static sources would need to be presented in a suitable and understandable structure for the subscribers (R3).

Non-technical Users: The doctor and the real estate agent are not technology experts (R1); therefore, they find complex queries like SPARQL to be unfriendly [26]. Also, they are not aware a-priori of the sources available in the IoT system [90] to manually select which ones are appropriate [26] nor of the semantics or schemata used in the data coming from sensors of different manufacturers to make more explicit queries (e.g. Fig. 2.1: "peripheral capillary oxygen saturation" or "blood oxygen saturation levels" instead of "SpO2" and Fig. 2.2: "kitchen entrance" instead of "kitchen", "Fahrenheit" instead of "degrees Celsius"). The doctor and the real estate agent prefer a query that is not too simple to provide partial patient/property information or too abstract (query of low expressiveness) to overload them or the system with too much and possibly redundant data [24] (R2). They need a contextually-aware, representationally-decoupled query of low complexity; therefore, high usability, that covers as much complete and diverse information provided by the sources as possible (e.g. the doctor wants to obtain health status information based only on a patient's name) (R1, R2).

2.3.3 System Challenges

These challenges involve scalability, timeliness, and resource constraints.

Scalability: The sensors and patients/properties involved are dynamic, that is they can be created or deleted at any time [19]. Also, the data deriving from these dynamic sensors is continuous (unbounded length of data [91]) and characterised of a range of different velocities that could be high or low depending on the situation (e.g. critical situation of a patient in a hospital will create health readings of higher speed). The data needs to be efficiently transferred, processed without the ability to backtrack over previously arrived data [92], stored, and managed independently of the amount of data sources or velocity (R6).

Timeliness and Resource Constraints: The high number of sensors and patients/properties along with data characterised by high velocity and heterogeneity that could lead to propagation, storage, and management of unnecessary data will result in voluminous data that causes network overhead and slower processing time [33]. The doctor and the real estate agent are interested in up-to-date information so that they can take immediate actions when needed (e.g. upcoming heart attack of a patient). Therefore, the data needs to be processed quickly and in real-time (as soon as the information items are available) with low latency, high throughput, and as low memory consumption as possible to satisfy the subscribers' needs (R6).

An overall map among challenges and requirements is presented in Table 2.1.

Table 2.1: Overall map among challenges and requirements

Challenges	**Requirements**					
	R1	**R2**	**R3**	**R4**	**R5**	**R6**
Data Challenges						
Heterogeneity		✓	✓	✓	✓	
Redundancy		✓	✓	✓	✓	
Data Enrichment		✓	✓			
User Challenges						
High-level Interpretation		✓	✓	✓	✓	
User-friendly Data Representation			✓			
Non-technical Users	✓	✓				
System Challenges						
Scalability						✓
Timeliness and Resource Constraints						✓

*R*1: Usability, *R*2: User Expressibility, *R*3: Data Expressiveness
*R*4: User Effectiveness, *R*5: Data Effectiveness, *R*6: Efficiency

2.4 Event-based Communication Paradigms

Event-based systems have been proposed as suitable communication paradigms for dynamic large scale applications that could resolve issues regarding system performance including scalability, timeliness, and resource constraints. Nevertheless, they pose limitations that result in the lack of any existing system that is able to address all of the aforementioned IoT challenges regarding data and users.

2.4.1 Publish/Subscribe Systems

Publish/Subscribe systems (Pub/Sub) [1] are middleware that were initially proposed with the main objective of flexible communication. At some point, it was evident that the world and its demands will become so complex that point-to-point and synchronous communications would fall short as they could only apply to static and rigid applications and would create many hurdles for dynamic large scale ones. Environments like IoT, for example, pose much more challenges apart from the dynamism and large scale; therefore, a suitable efficient communication scheme (middleware) should be applied as a basis and, then, more sophisticated data processing approaches should be proposed to cater not only for efficiency but effectiveness as well.

As aforementioned, Pub/Sub [1] contain three parties; *publishers*, *subscribers*, and the *event service*. Publishers (also termed *producers*) generate data, which is called *events* or *publications*, while subscribers (also termed *consumers*) register their interest in an event or pattern of events through conditional rules, which are called *subscriptions* and they are equivalent to queries. All publications and subscriptions are sent to the event service, which is responsible for detecting matches when an event fully satisfies all the rule conditions of a subscription. If a match occurs, then, the corresponding events, which are called *notifications*, are sent to the relevant subscribers, asynchronously.

The Decoupling Traits of Publish/Subscribe Systems

Pub/Sub are ideal candidates to act as a communication paradigm for IoT applications as apart from abstracting the users from the underlying communication complexity, they are scalable, distributed, and decoupled in three ways:

Space Decoupling: The interacting parties do not need to know each other. The publishers only send the generated events to the event service without being aware of any information regarding the subscribers, including their number, their requests, who they are etc. At the same time, subscribers subscribe their interests and send these subscriptions to the event service without being aware of any information regarding the publishers, including their number, the nature of all available events etc.

Time Decoupling: The interacting parties do not need to be active at the same time for an interaction to take place. Publishers will send events to the event service even when the interested subscribers are disconnected, and vice-versa, subscribers will receive events generated by publishers that may no longer be connected.

Synchronisation Decoupling: The interacting parties are not blocked from performing concurrent activities when an interaction occurs. Since the interaction is managed by the event service, publishers can independently keep producing events and subscribers can perform other tasks while being notified.

These decoupling traits are suitable when efficiency is considered among things in IoT environments; nevertheless, they are not sufficient by themselves to address all challenges involved like simplicity and usability along with interoperability and heterogeneity.

The Schemes of Publish/Subscribe Systems and Their Limitations

There are three main schemes of Pub/Sub according to Eugster et al. [1]; *topic-based*, *content-based* and *type-based*, with the latter being a sub-category of topic-based Pub/Sub. In this thesis, type-based Pub/Sub are not explicitly addressed, and an additional scheme is analysed that has attracted a lot of attention lately; the *graph-based Pub/Sub*, as termed in this thesis.

Topic-based Publish/Subscribe Systems Events and subscriptions, in this scheme, relate to individual topics, which are expressed as keywords. The topics are groupings or classifications of events based on similarities or content. Subscribers subscribe to specific topics and when a related event is published, all members of the topic group are notified. Topics could refer to either a flat topic (e.g. a name, subject, or topic) or a hierarchy of topics, that is sub-topics (children in a taxonomy) that have the initial topic as a parent. If a subscriber subscribes to a topic with children, then, one subscribes to all sub-topics too by default. A topic could be an exact URL-like notation, a notation with wildcards, or a regular expression that covers a set of possible topics. An example of a topic-based Pub/Sub is depicted in Fig. 2.3 for the topic "Marie_Curie", where all events are sent as a notification to the subscribers.

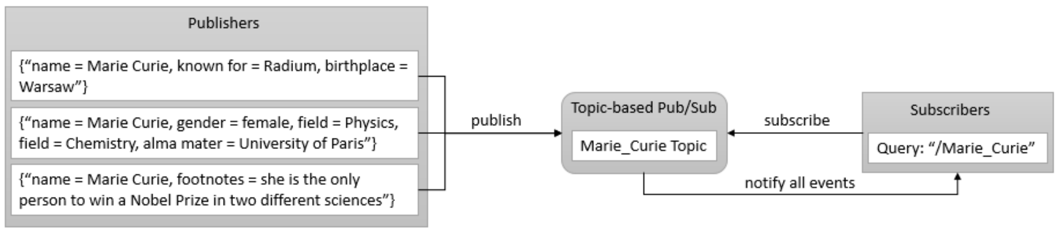

Publishers
{"name = Marie Curie, known for = Radium, birthplace = Warsaw"}
{"name = Marie Curie, gender = female, field = Physics, field = Chemistry, alma mater = University of Paris"}
{"name = Marie Curie, footnotes = she is the only person to win a Nobel Prize in two different sciences"}
publish
Topic-based Pub/Sub
Marie_Curie Topic
subscribe
Subscribers
Query: "/Marie_Curie"
notify all events

Figure 2.3: A topic-based Pub/Sub system example.

Topic-based subscriptions could be linked to keyword-based queries (Fig. 2.6); therefore, they are of low complexity (simple), easy to be defined, and clearly understood by the users. Also, due to their abstraction (e.g. a query refers to an entity in general), they could cover different conceptual data interpretations by humans regarding this entity. These advantages make topic-based Pub/Sub highly usable systems. Nevertheless, publishers and subscribers need to have a shared understanding of the topics and what they represent as the event service is not responsible for any conceptual and contextual interpretation of the topics, but only for the matching. Also, the topic-based subscriptions have very limited query expressiveness (i.e. filtering capabilities) and given their simplicity, abstraction, and shortness the notifications may contain an abundance of information characterised by redundancy and irrelevance to the users' needs. This will not only overwhelm the users and the Pub/Sub system's performance due to the propagation and storing of unnecessary data but it will require the users to filter unnecessary information themselves. Overall, the topic-based Pub/Sub system cannot currently address all of the aforementioned IoT challenges.

Content-based Publish/Subscribe Systems Due to the limited expressiveness of topic-based Pub/Sub, another scheme was proposed, where events and subscriptions relate to the actual content of the events. Events are separate chunks of information that have their own structure and content. Subscribers subscribe to these events by performing content filtering. This filtering typically involves comparison operators (=, <, ⩽, >, ⩾) on attribute-value pairs deriving from the events. Complex subscription rules or patterns can also be created by logical combinations (and, or etc.) of individual constraints. An event is matched by the content-based Pub/Sub and sent as a notification to the subscribers only if it fully satisfies all of the subscription's constraints. An example of a content-based Pub/Sub is depicted in Fig. 2.4, where subscribers subscribe to the complex subscription rule or pattern "gender = female AND field = Physics" and publishers publish separate events that may or may not cover this subscription. In the example, only one of the events fulfils all the constraints of the subscription and is returned as a notification to the subscribers.

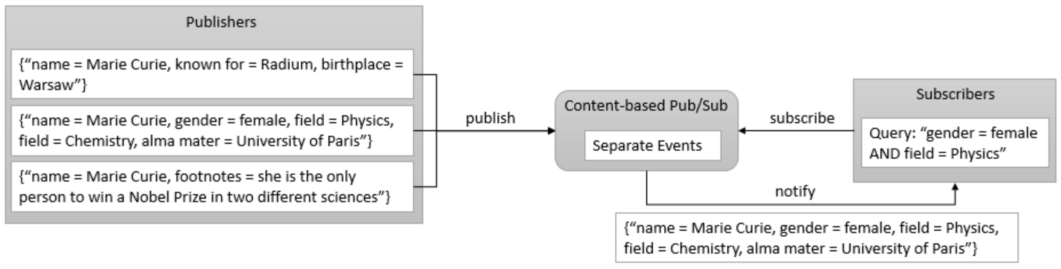

Publishers
{"name = Marie Curie, known for = Radium, birthplace = Warsaw"}
{"name = Marie Curie, gender = female, field = Physics, field = Chemistry, alma mater = University of Paris"}
{"name = Marie Curie, footnotes = she is the only person to win a Nobel Prize in two different sciences"}
publish
Content-based Pub/Sub
Separate Events
subscribe
Subscribers
Query: "gender = female AND field = Physics"
notify
{"name = Marie Curie, gender = female, field = Physics, field = Chemistry, alma mater = University of Paris"}

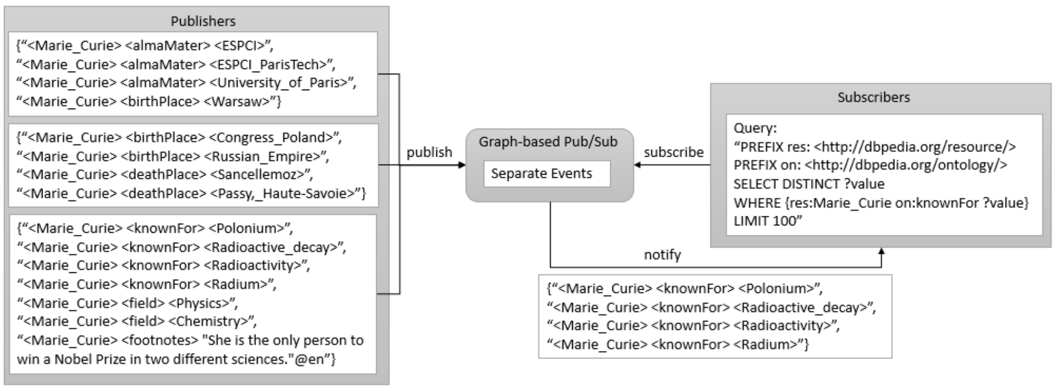
Figure 2.4: A content-based Pub/Sub system example.

Content-based subscriptions could be linked to attribute-value pairs based queries (Fig. 2.6); therefore, they are of medium usability due to their medium complexity. Although their query expressiveness is higher than the one in the topic-based subscriptions (leading also to less memory-heavy notifications), overall, they present an expressiveness of medium level. This occurs because the events may not be conceptually and contextually rich enough (e.g. attribute-value pairs are of medium informational value) in order to give more opportunities to the users for more expressive queries related to their needs. Also, the flexibility given to the users to create their own query constraints may pose several challenges. Specifically, users may not have a clear understanding of the concept of a thing/entity and they could create strict narrow queries based on their intuition on what a thing/entity is that will lead to partial information about it as a notification. This makes a high-level interpretation of results more difficult for users. Furthermore, if users were interested in integrated data information provided by several sources, they would have to perform complex join queries and be aware of all the sources involved in an entity's data generation [26]. They would also have to be aware of the available content provided by the sources or the future changes, which is not the case as subscribers are often unclear on what they actually subscribe to or what information they will get in the end. Hence, some Pub/Sub systems have publishers that advertise the nature of future events for the subscribers' benefit.

One of the biggest limitations of content-based Pub/Sub is assuming the users are aware of the schemata and semantics (representation) of the heterogeneous data provided by the publishers in order to make specific conditional queries without also considering the users' expertise in query creation or languages. Existing approaches [39, 57, 58] have tried to address this issue by providing the users with the opportunity to create approximate/relaxed queries; nevertheless, an approximate/relaxed query might lead to conceptually-similar results that themselves lead to an abundance of information that could be deemed redundant by the users. On the other hand, a strict exact query might lead to unnecessary duplicate results. Duplication and redundancy will not only overwhelm the users, but will create hurdles in the Pub/Sub system's performance due to the propagation and storing of unnecessary data. Overall, the content-based Pub/Sub system cannot currently address all of the aforementioned IoT challenges.

Graph-based Publish/Subscribe Systems This scheme could be considered a sub-category of content-based Pub/Sub; nevertheless, it is emphasised in this thesis since its events are in graph forms rather than attribute-value pairs [93].

Graphs are conceptually and contextually rich pieces of information, where nodes represent things/entities or attribute values and edges represent the relations between the nodes. Events are seen as separate chunks of information that have their own graph structure and content. Subscribers subscribe to these events by performing content filtering, that is SPARQL-like queries, in which points of interest are specific graph nodes and edges. A graph that matches the subscriptions is sent as a notification to the subscribers by the graph-based Pub/Sub. An example of a graph-based Pub/Sub is depicted in Fig. 2.5, where subscribers subscribe their interest to what Marie Curie was known for through a SPARQL query and publishers publish separate event graphs that may or may not cover this subscription (prefixes have been omitted for simplicity). In the example, only one of the events contains the information needed by the subscribers and it is returned to them as a notification.



```
{"<Marie_Curie> <almaMater> <ESPCI>",
"<Marie_Curie> <almaMater> <ESPCI_ParisTech>",
"<Marie_Curie> <almaMater> <University_of_Paris>",
"<Marie_Curie> <birthPlace> <Warsaw>"}
```

```
{"<Marie_Curie> <birthPlace> <Congress_Poland>",
"<Marie_Curie> <birthPlace> <Russian_Empire>",
"<Marie_Curie> <deathPlace> <Sancellemoz>",
"<Marie_Curie> <deathPlace> <Passy,_Haute-Savoie>"}
```

```
{"<Marie_Curie> <knownFor> <Polonium>",
"<Marie_Curie> <knownFor> <Radioactive_decay>",
"<Marie_Curie> <knownFor> <Radioactivity>",
"<Marie_Curie> <knownFor> <Radium>",
"<Marie_Curie> <field> <Physics>",
"<Marie_Curie> <field> <Chemistry>",
"<Marie_Curie> <footnotes> "She is the only person to
win a Nobel Prize in two different sciences."@en"}
```

```
Query:
"PREFIX res: <http://dbpedia.org/resource/>
PREFIX on: <http://dbpedia.org/ontology/>
SELECT DISTINCT ?value
WHERE {res:Marie_Curie on:knownFor ?value}
LIMIT 100"
```

```
{"<Marie_Curie> <knownFor> <Polonium>",
"<Marie_Curie> <knownFor> <Radioactive_decay>",
"<Marie_Curie> <knownFor> <Radioactivity>",
"<Marie_Curie> <knownFor> <Radium>"}
```

Figure 2.5: A graph-based Pub/Sub system example.

Graphs are conceptually and contextually rich as well as dynamic structures that could represent IoT data very well; therefore, graph-based Pub/Sub systems would pave the way for more informative data analytics and more expressive query possibilities [94] than the other Pub/Sub schemes. Nevertheless, graph-based subscriptions are linked to SPARQL-like queries (Fig. 2.6); therefore, they are of low usability due to their high complexity. Also, as a sub-category of content-based Pub/Sub systems, they inherit all of its limitations. They have recently been examined with more possibilities for the future. At the moment, there is no approximate/relaxed queries approach [19, 95, 96], making the representational coupling problem even more evident since graphs have an even richer span of different schemata and semantics. This contradicts the purpose of having the possibility of representing IoT data in rich graph structures compared to narrow, rigid attribute-value ones, but not being able to also perform flexible matching. Apart from representational coupling, another big limitation of graph-based Pub/Sub is assuming the users are experts in query formation and complex languages [26]. The way the queries are formed is very important when graphs are involved since due to the graphs' complexity and dynamic nature, they could contain numerous nodes and edges. This will lead to not only the necessitation of queries that are strict enough to return relevant results but to the possibility of even more options for new queries by the users as graphs get updated with more information making their old queries obsolete and even irrelevant. Overall, the graph-based Pub/Sub system cannot currently address all of the aforementioned IoT challenges.

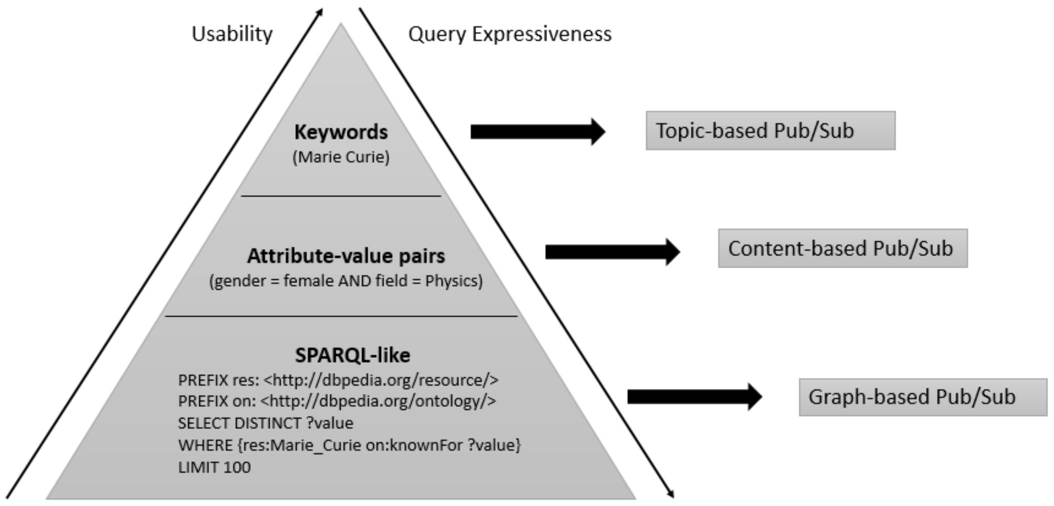


Figure 2.6: Relation between query types and Pub/Sub schemes.

Overview of Publish/Subscribe Systems' Advantages and Disadvantages

It is observed, then, that the current Pub/Sub schemes cannot currently be directly applied to complex IoT environments and address all of the aforementioned challenges. New schemes or schemes that combine the advantages of existing ones should be proposed, which is the aim of this thesis. An overview of the advantages and disadvantages of the different Pub/Sub schemes is illustrated in Fig. 2.7.

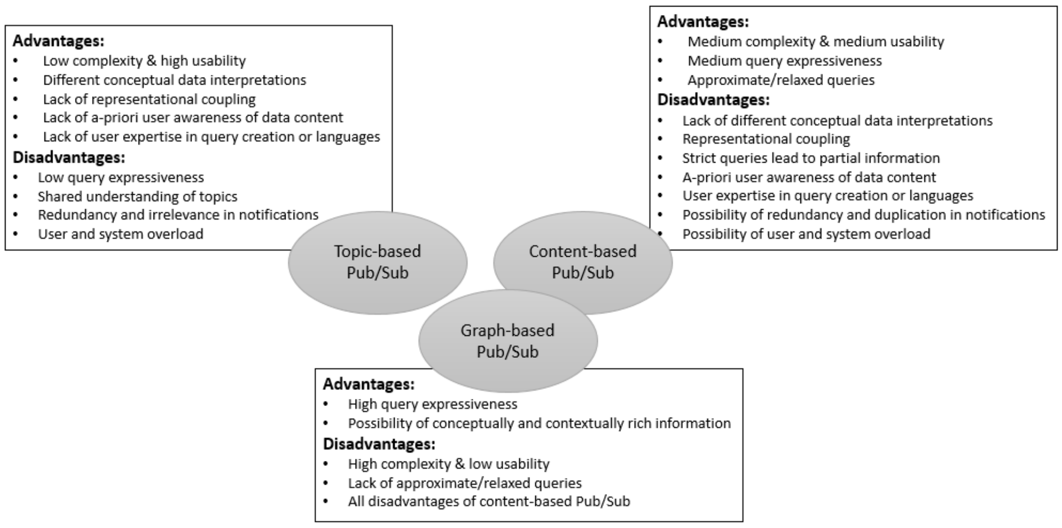

Advantages:
• Low complexity & high usability
• Different conceptual data interpretations
• Lack of representational coupling
• Lack of a-priori user awareness of data content
• Lack of user expertise in query creation or languages
Disadvantages:
• Low query expressiveness
• Shared understanding of topics
• Redundancy and irrelevance in notifications
• User and system overload
Topic-based Pub/Sub
Content-based Pub/Sub
Advantages:
• Medium complexity & medium usability
• Medium query expressiveness
• Approximate/relaxed queries
Disadvantages:
• Lack of different conceptual data interpretations
• Representational coupling
• Strict queries lead to partial information
• A-priori user awareness of data content
• User expertise in query creation or languages
• Possibility of redundancy and duplication in notifications
• Possibility of user and system overload
Graph-based Pub/Sub
Advantages:
• High query expressiveness
• Possibility of conceptually and contextually rich information
Disadvantages:
• High complexity & low usability
• Lack of approximate/relaxed queries
• All disadvantages of content-based Pub/Sub

Figure 2.7: Overview of the advantages and disadvantages of the different Pub/Sub schemes.

2.4.2 Complex Event Processing Systems

CEP originated from Pub/Sub systems, although, contrary to traditional Pub/Sub that observe events individually (single events), CEP have a more expressive query and rule language (a set of conditions and actions) that observes occurrences, patterns, and relationships (e.g. sequencing or ordering) of multiple events (complex events). There is a plethora of existing systems that have focused on distributed architectures for efficiency and contain their own query and rule languages as well as processing methodologies.

Prominent Complex Event Processing Systems and Languages

Some of the most prominent CEP systems and languages are the following:

PADRES [97]: PADRES is a distributed and decentralised content-based Pub/Sub system that can analyse individual and composite events alike. Publications and subscriptions are in the form of attribute-value pairs. Subscriptions support sequence and repetition operators apart from logical. The system advertises future events that will be generated to prepare subscribers. The matching engine examines the subscriptions based on their overlap with the advertisements and transforms them into rules. If an event matches a rule, then, the subscriber is notified. The system uses several windowing policies to analyse events, which can facilitate the timing constraints, sequencing and access to historical data. As with all Pub/Sub systems, its prior emphasis was efficient communication and not sophisticated data analytics methodologies. As a content-based Pub/Sub, PADRES inherits all of its aforementioned limitations.

CEDR [98]: CEDR is a general purpose event streaming system. It emphasises on temporal aspects of data and errors in flow like out-of-order streams. CEDR keeps a history of received notifications to users and only updates them when a change has occurred. It uses a SPARQL-like query language that expresses event patterns with temporal and value correlation, negation, aggregation as well as query directed instance selection and consumption.

Cayuga/CEL [99]: Cayuga is a general purpose complex event monitoring system. It uses a SPARQL-like query language called CEL with typical operators like selection, projection, renaming, union and aggregation. It can also detect event patterns with operators like sequence and iteration. The operators use semantics from a query algebra. Automata are used to define in parallel all possible events satisfying the query rules. Other techniques focus on efficiency regarding query processing, indexing, and garbage collection.

Amit [100]: Amit is described as a situation manager that includes its own language and efficient execution techniques at reducing the complexity of reactive and proactive applications. It processes events received by different sources and detects patterns (situations in Amit's case) to notify interested subscribers. The language offers high expressive capabilities to the users by including conjunctions, negations, parameters, sequences, and repetitions as well as timing operators and the concept of lifespan that acts as a time window.

SASE [101] and SASE+ [102, 103]: SASE is a monitoring system for performing complex queries over real-time data. Its query language is a pattern-based rule one and resembles SPARQL. It supports time validity for a rule in the sense of time windows. Automata are used to define all possible events satisfying the query rules. Other techniques focus on selection, windows, and negations. SASE+ extends the expressiveness of SASE's language by supporting iterations, aggregates, and event selection strategies.

T-Rex/TESLA [104]: T-Rex is a general purpose CEP system that uses its own pattern-based language in the form of rules, called TESLA. Its aim is to keep a good level of query expressiveness by providing simplicity and a small set of operators like content and temporal constraints, parameters, negations, sequences, aggregates, timers, and fully customisable policies for event selection and consumption. The operators use semantics based on a first order, metric temporal logic. Automata are used to define all possible events satisfying the query rules. Other techniques focus on efficiency regarding custom data structures and indexing to tackle the memory burden. Hierarchies of events are also supported.

Esper/EPL[1]: Esper is an open-source CEP system. It uses its own pattern-based language in the form of rules that resembles SPARQL, called EPL. EPL contains queries either by using operators like conjunctions, disjunctions, negations, sequences, and iterations or by using regular expressions. It supports event selection and a range of windowing policies.

ETALIS/EP-SPARQL [105]: ETALIS is a system with event processing and stream reasoning capabilities that uses a language that extends SPARQL and is named EP-SPARQL. Given that languages of CEP systems cannot generally combine streams with background knowledge and perform reasoning tasks, but can deal with rapidly changing event streams, EP-SPARQL tries to address this gap. It is based on event-driven backward chaining rules and features processing and inference capabilities over temporal and static data. Several operators are supported like content and temporal constraints, filtering, aggregates, sequences, joins, and unions as well as windowing policies. A syntax and formal semantics model of the language are provided. The reasoning is based on knowledge bases and ontologies. There is an assumption by the system that users are aware of Resource Description Framework (RDF) and SPARQL.

For more systems and more comprehensive analysis and comparison among them, the reader is directed to Cugola and Margara [16], and Artikis et al. [106, 107].

The Troubling Side of Complex Event Processing Systems

The main aim of CEP systems is twofold: 1) to perform efficient processing and communication of single or complex events, and 2) to give the opportunity to the users to express their data pattern or temporal needs via expressive languages, operators, and rules. Although CEP systems excel at these two points, the fact that there is no standardised CEP system, but several different systems with their own architecture, data models, rule languages, processing methodologies, and vocabulary [16], should be deemed troublesome.

On a language level, almost all systems are based on SPARQL-like queries,

[1]`https://www.espertech.com/esper/`

which are proven difficult for non-expert users. Apart from that, the vast range of different operators, capabilities, syntaxes, and semantics among languages makes even expert users ambiguous [104]. Some systems recognised the difficulty involved and proposed formal syntax and semantic models of their proposed language to explain the meaning of queries and provide query plans to implement those queries; nevertheless, this does not solve the issue when a user has a complete lack of knowledge regarding query languages. Another important issue is that some languages are too expressive and complex to justify the users' data needs [103]; making the systems even less usable.

On a data level, the systems refer to simple or complex events, mostly regarding timestamped tuples. The world is changing rapidly, the data needs are becoming more demanding and environments like IoT will require more advanced methodologies than supporting expensive and expressive queries with high system performance. Hence, systems like ETALIS tried to merge the world of Semantic Web with the one of CEP to create an event processing engine that will integrate background information and perform reasoning to describe the context or domain in which data streams are interpreted. For example, most CEP systems would have to be altered to support dynamic graph-based heterogeneous events that have a wide range of conceptual and contextual diversity.

Therefore, CEP systems may be efficient and query expressive; nevertheless, they suffer from representational coupling and low usability. Users are obliged to be aware of query languages, schemata, and semantics used by the data. They also need to understand data that has temporal complexity and is unbounded as well as real-time in nature. Given the vast volume of CEP sources and the high velocity of streams, these systems are mainly targeting expert users that have a-priori knowledge of the data and a clear understanding of their needs to perform complex rules that may not even completely satisfy their needs due to the lack of more sophisticated approaches like reasoning or knowledge background integration.

2.4.3 Data Stream Processing Models

Another type of system that can process streams of data is data stream processing models. These models analyse continuous streams of information coming from different sources and provide answers for SPARQL-like query languages that involve a range of operators like selections, aggregates, joins etc. There are several languages that have been proposed with their own capabilities and characteristics.

Major Data Stream Processing Languages

Some of the most major data stream processing languages are the following:

CQL [108]: CQL is a general purpose language for continuous queries over streams and stored relations. Abstract semantics are provided that rely only on "black-box" mappings among streams and relations. Operations like stream-to-relation, relation-to-relation, and relation-to-stream are supported like filtering, join, aggregation, and windowing policies. Physical query execution plans are presented along with the syntax and semantics of the language.

ESL [109]: ESL is an expressive stream language that supports data stream mining, streaming XML processing, time-series queries, and RFID event processing. Operators like physical and logical windows are supported along with aggregation and union.

Streaming SPARQL [110]: Streaming SPARQL extends SPARQL to cover RDF data streams and to provide a range of windowing policies. Triple pattern matching, filtering, union, join, graph pattern matching (join over two or more triple pattern matchings), and de-duplication are some of the stateless and stateful operators that are supported. Logical algebra is used to define the operator semantics and allow some static plan optimisations. A physical algebra is used for the execution of queries.

C-SPARQL [111]: C-SPARQL extends SPARQL to cover continuous queries over static and streaming RDF data. Operators that are supported include filtering, grouping, aggregation, union, multiple stream integration as well as windowing policies. A semantic model is created and a query graph model is dedicated to optimisation or rewriting rules. Its aim is to set the basis for more advanced data analytics like reasoning, background information integration, and evolving knowledge over time.

SPARQLstream [112]: SPARQLstream is a language for querying streaming data mapped with ontology models via other specific mapping languages. Direct mapping, join, union, projection, selection, and windowing are operators that are supported. This approach contains not only a separate syntax and semantics related to the main language but also to the mapping language, making it even more complex.

CQELS [113]: CQELS is a native and adaptive query processor for unified query processing over Linked Stream Data and Linked Data. It is an open system approach that can dynamically adapt to input data changes. Windowing, relational (e.g. join, union), and streaming operators are supported. Optimisation methodologies are also considered regarding caching, indexing, query execution, delay, and complexity.

Data Stream Processing Languages Limitations

The data stream processing languages look at streams in isolation or process them incrementally so that the results produced by a rule become input for others; therefore, they cannot observe complex events, temporal sequences, iterations, or complex patterns when it comes to individual data streams deriving from multiple sources like CEP systems can [104]. From this perspective, CEP systems provide higher user expressibility on a language level. On the other hand, the aim of data stream processing languages is to integrate static and streaming RDF data. RDF data streams are semantically richer than just attribute-value pairs or tuples that are covered by CEP systems. The RDF data structure along with capabilities like stream reasoning and evolving knowledge over time, which are highly beneficial to users, make the data expressiveness much higher in these systems.

Another observation is that, like CEP systems, there is no standardised data stream processing system or language. All languages have been created with different aims, expressiveness capabilities, solutions, limitations, syntaxes, and semantics in mind. The fact that they are also based on SPARQL-like queries declares that they all suffer from representational coupling and low usability. All these traits would not only confuse a non-expert user but also an expert one that is unclear which language to choose either for the best expressibility levels according to their needs or for extending to resolve existing limitations.

Some systems may not even have the high system performance observed in CEP systems. For example, when systems are built on top of existing stream data management systems and triple stores, their processing time and accessing of data will significantly be delayed, especially when the data is of large volume. Also, structured query languages, like these, do not scale well to large schemata with a plethora of attributes.

Overall, it is observed that these languages also suffer from low usability and representational coupling since users are obliged to be aware of query languages, schemata, and semantics used by the data. Nevertheless, these languages are a step in the right direction regarding merging the worlds of Semantic Web and stream processing in order to create conceptually and contextually rich data information with reasoning capabilities and high-level inferences to cater for user needs and simplicity. Solutions that can perform more advanced aggregates apart from the count, mix, max, sum, and average need to be found, which can also support simple query languages, independence on domains, ontologies, and databases or triple stores as well as scalable and on-the-fly inference capabilities and data enrichment [105].

2.5 Entities and Knowledge Graphs

IoT environments involve the inter-connectivity of a multitude of entities (or things). Entities are real-world or abstract things [114], which could contain information of high conceptual and contextual diversity. With the advent of Semantic Web [115], Linked Data [116] in the form of knowledge graphs have been used to represent the complexity of entities, in which nodes are entities and their directed labelled arcs constitute relations among them. Data modelling languages like RDF and Web Ontology Language (OWL) have been used to express these representations in specific schemata. In RDF, entity information is presented in the form of triples ⟨subject, predicate, object⟩, where the subject is an entity, the object is either an entity (object-type predicates) or a number/string (data-type predicates), and the predicate is their relation. RDF triples with the same subject form an RDF star-like graph. Popular knowledge bases are Freebase [117], DBpedia [118], and YAGO [119] and they are often stored in relational databases, triple stores, and graph databases for further analysis. An example of a knowledge graph for Marie_Curie is given in Fig. 2.8 (note: only a part of the whole entity information is depicted for visualisation purposes), where (http://dbpedia.org/resource/Marie_Curie, http://www.w3.org/1999/02/22-rdf-syntax-ns#type, http://xmlns.com/foaf/0.1/Person) is one of the triples. This triple is part of the standard schema in RDF and it represents the class membership, category, or typing information of an entity. More than one entity can have the same type and an entity can have more than one type, either general or fine-grained ones. The process of assigning a type to an entity is called *entity typing* and is usually provided in popular knowledge bases like DBpedia as shown in Fig. 2.8.

Semantic Web technologies along with data stream processing have been used in IoT architectures and methodologies as apart from resulting in semantically enriched information compared to plain data streams or events [105] and

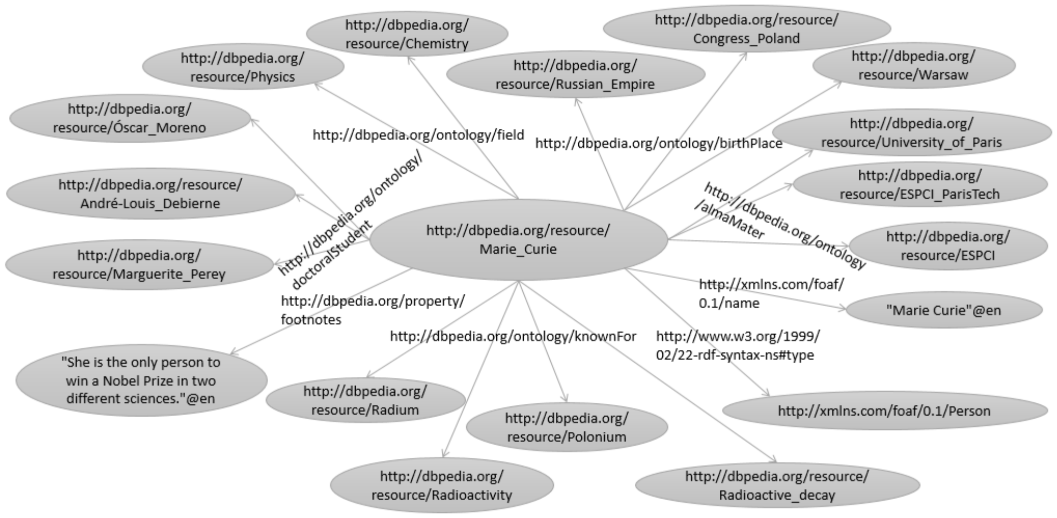

http://dbpedia.org/resource/Marie_Curie
http://dbpedia.org/resource/Physics
http://dbpedia.org/resource/Chemistry
http://dbpedia.org/resource/Óscar_Moreno
http://dbpedia.org/resource/André-Louis_Debierne
http://dbpedia.org/resource/Marguerite_Perey
http://dbpedia.org/resource/Russian_Empire
http://dbpedia.org/resource/Congress_Poland
http://dbpedia.org/resource/Warsaw
http://dbpedia.org/resource/University_of_Paris
http://dbpedia.org/resource/ESPCI_ParisTech
http://dbpedia.org/resource/ESPCI
"Marie Curie"@en
http://xmlns.com/foaf/0.1/Person
http://dbpedia.org/resource/Radioactive_decay
http://dbpedia.org/resource/Polonium
http://dbpedia.org/resource/Radioactivity
http://dbpedia.org/resource/Radium
"She is the only person to win a Nobel Prize in two different sciences."@en
http://dbpedia.org/ontology/field
http://dbpedia.org/ontology/birthPlace
http://dbpedia.org/ontology/doctoralStudent
http://dbpedia.org/ontology/almaMater
http://xmlns.com/foaf/0.1/name
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://dbpedia.org/ontology/knownFor
http://dbpedia.org/property/footnotes

Figure 2.8: The knowledge graph (RDF star-like graph) of entity Marie_Curie.

contributing to richer data analysis and querying [94], they are believed to resolve the interoperability issue among different entities, be it sensors or end-users [78, 105]. Approaches have leveraged existing open protocols and W3C standards of the Web architecture to create ontology models that not only represent real-time or evolving data but also enrich data through external sources (like the Web) with specific schemata/structures [37, 120]. RDF, OWL, and other data modelling languages are used to depict this IoT data in structured schemata. Nevertheless, although it is a step in the right direction as semantic graphs contain conceptually and contextually rich information and can also contribute to reasoning for the benefit of the users, the problem of interoperability still remains as there is no standardisation regarding ontology models, and ontologies themselves can become quite voluminous and complex. Other linguistic resources like thesauri, dictionaries, and taxonomies pose many limitations, as well, resulting in the exploration of more flexible linguistic models, like word embeddings.

2.5.1 Ontologies and Their Limitations

Ontologies are formal conceptualisations of particular domains, which describe the content, characteristics and taxonomies/hierarchies (semantic proximity between topics/concepts) regarding a topic so that a common understanding is extracted for the communication between people and application systems [121, 122]. Ontology models are described by languages that have specific schemata (e.g. relations like subClassOf, type, range, domain etc.). Ontologies have a high expressive power when it comes to describing complex entities or concepts in a detailed level with a specific structure [123]. This structure is understood by machines/systems and expert users assimilated to the Semantic Web. Nevertheless, ontologies pose many disadvantages by themselves and although using them for annotating static big data could prove useful, when integrated with even more complex and dynamic environment requirements like IoT, which involve data streaming, temporal sequences, support for expressive queries, integration with sensor data with a wide range of representations, and spatial information, the limitations are becoming even more important [89].

Complexity and Inefficient Search: Ontologies can become very complex and versatile as more data is available regarding an entity [17] and a plethora of links may occur [78]. Since there is no limitation on the amount of information an ontology can depict, in situations where data is integrated by several sources, the ontology volume could become quite enormous [124]. In the case of streaming data or complex events, where the sources are dynamic and the data unbounded, this limitation is even more evident [122]. The integration of multiple ontologies deriving from different sensors into one standard coherent and complete ontology for search and the search itself will consume many resources that may be prohibitive in some applications [122]. Maintaining and updating a computationally expensive ontology is challenging [123, 125]. Also, the content may be erroneous and does not come with data interpretations as it is difficult to completely define what an object denotes or means [28]; therefore, users are perceiving the data based on their bias, personal experience, and background information (e.g. some words may be unfamiliar to users or the context might be different from what they perceive or even incomplete). Often ontologies' data may not even make sense the way it is described with complex prefixes and long concatenated words that may represent important user information but no actual words [125].

Representational Coupling: There is no fixed vocabulary regarding the semantics used in ontologies. Different communities, manufacturers, and approaches create their own ontology by using different words to describe the same entity, content, or attribute/property [121] and they have individual explicit and rigid assumptions regarding the intended meaning of these words [28]. Synonymy, hyponymy, hypernymy, antonymy, ambiguity, and polysemy are some of the linguistic problems that could be present when using words [27]. This contributes to semantic interoperability issues among systems and the problem of mapping between user queries and available ontologies' semantics. Furthermore, different approaches may not follow a consistent widely accepted ontology schema as they focus on different problems and need to create additional schemata or re-engineer existing basic ones. This issue along with different semantics contribute to representational coupling issues [120] making ontologies ineffective for the heterogeneous nature of IoT data that comes from multiple sources [58] and need to work on a global scale [37]. Some works [126] have focused on how to integrate ontologies in order to solve the schematic mismatch, showing that this is a problem that has not been resolved yet since no standardised ontology model exists. Some other works [63, 65, 73] tried to create or extend a standard model like the W3C Semantic Sensor Networks (SSN) ontology [127] to represent physical and virtual sensors, but it is still not abided by all communities as an official one. As different organisations on an international or national level strive to develop IoT standards, strong coordination needs to take place among them so that developers and users can develop large scale IoT applications and services while saving the development and maintenance cost issues in the future [15].

Advanced Data Analytics Needed: Although ontologies can form knowledge graphs that contain a wide range of conceptually and contextually rich information, it does not mean that they contain data interpretations for the benefit of the user. Advanced data analytics methodologies or complex user queries are still needed in order to provide high-level abstractions or reasoning over the raw ontology information [15, 37, 74]. Also, when data is integrated to provide more complementary information to the user, different ontologies may contain incompatible levels of abstraction/taxonomy; therefore, data commonalities and redundancies may occur if not addressed properly by advanced methodologies [120]. These methodologies should also be able to address challenges related to scalability, timeliness, and resource constraints.

Query Expressiveness vs Usability: Usually ontologies entail SPARQL queries that offer high expressiveness, but at the expense of usability [20]. Users need to be aware of data representations (e.g. complex prefixes, syntaxes, and semantics), the available data content and be experts in complex query languages [26] in order to create specific queries, otherwise, a search match will not occur [25]. Users are not allowed to define their own vocabulary and the ontologies' syntax was created with the focus to be understood by the machines and not by the users. Users need to be experts to understand how to search and explore ontologies. Apart from these issues, users are treated like they are clear on what they are looking for, which is not the case. For example, when an average user searches on the Web, one often starts by general, abstract terms and, then, one narrows down their search according to results [125]. Hence, simpler, user-friendly, understandable keyword searches in RDF datasets, for example, are more intuitive to the users. Nevertheless, they may return an abundance of unnecessary (irrelevant) information since the intention of the user is not clear, the query is too short/simple (no filtering), and there is no single presentation that can cater for all kinds of information needs.

Domain Dependency: Most ontologies currently used are domain-dependent, that is they contain conceptual information related to one domain [125]. These kinds of ontologies cannot apply to heterogeneous IoT data [128] as they may span in more than one domain, and their availability and accuracy rely on the existence of domain experts, which may not be the case for all situations. Some domain-independent ontologies exist, but they are not as effective since they contain general purpose or ambiguous vocabulary and relations to capture specialised properties of any domain [129].

For a more in depth analysis of ontology challenges and evaluation methodologies, the reader is directed to McDaniel and Storey [130].

2.5.2 Other Linguistic Resources and Their Limitations

There are other linguistic resources apart from ontologies (all characterised as corpus-independent) that have been used to either represent knowledge structure or to tackle some of the aforementioned problems either by query rewriting/expansion, semantically relaxed queries, or extraction of synonyms, related words, hypernyms, and hyponyms in data, and even ontology's term expansion. These include domain-specific or general thesauri, dictionaries, and taxonomies [122, 123]. Thesauri and dictionaries represent semantic relations among words in a structured vocabulary, whereas taxonomies represent hierarchical relations among the concepts of a domain. One of the most popular thesauri is WordNet [27] that contains a range of part of speeches and word senses (related to polysemy or the word's context). Relations like synonymy, hypernymy, hyponymy etc. are included. Merriam-Webster[2] and Roget's Thesaurus[3] are also most commonly used thesauri with the last one also containing word senses.

One of the problems of these linguistic resources is that they have lower semantic expressiveness, and less complete and precise domain models than ontologies [28], although they are less complex. Taxonomies have the lowest semantic expressiveness, followed by thesauri/dictionaries [123]. Also, some of them are domain-specific [125], meaning that they cannot be applied to other domains and they possibly need domain-experts for their construction like ontologies. It is also important to note that the quality of these resources depends on their accuracy, completeness, and up-to-date information. Another problem regarding taxonomies is the decision of until which level from the original word/concept another word/concept is considered relevant. For example, Jones et al. [131] considered that a conceptual distance of up to five levels is considered suitable by users, but this is highly subjective and as more information becomes available and more fine-grained structure is represented in a taxonomy, this level threshold will change. The topical context in dictionaries, that is the vocabulary used when a topic is discussed, is an interesting method to use; however, only 80% of the senses/contexts can correctly be defined as people tend to use local context, that is the sequence of words used immediately before or after a polysemous word [27]. This is the fact on which word embeddings hanged on and created powerful models that overcame many aforementioned challenges.

2.5.3 Word Embedding Models

In environments like IoT that contain data with high heterogeneity and redundancy, it is important for the words contained in the data not to be treated as individual ones (semantic coupling). Their semantic relationship should be examined in order to discover synonymous, related, or antonymous words. Word embeddings are mostly Deep Learning models that find the semantic and contextual relationship among words. Contrary to the other linguistic resources that are hand-built, these models are trained on big text-corpora (corpus-dependent) and create vectors for each word that when projected in a vector space will show semantically related words close to one another. Word embeddings are commonly categorised into two types; prediction-based, which leverage local

[2]https://www.merriam-webster.com/thesaurus

[3]http://www.roget.org/

information and either predict a word given its surrounding words (context) or predict the surrounding context words given a word, and count-based, which leverage global statistical information that check the word-context co-occurrence counts throughout a corpus, that is the times certain words appeared within a specified window around a word [132, 133]. An example of a vector space is illustrated in Fig. 2.9, where the position of each word is depicted based on the ConceptNet [134] embedding model. Each partition represents the close relationship among words. For example, {luminance, light} are related words, {darkness, dark} are synonyms to each other but antonyms to {luminance, light}; therefore, all words are semantically related. Related words, in the example, are close in the semantic space. For more details on word embeddings, sense representation, linguistic resources, and vector space models the reader is directed to Camacho-Collados et al. [135].

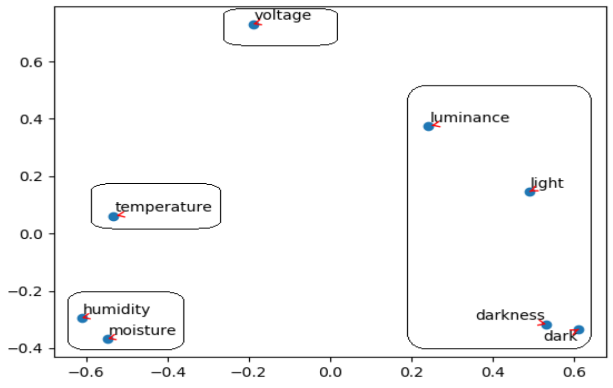


Figure 2.9: A vector space based on the ConceptNet embedding model.

Word embeddings are more flexible models compared to the strict ontologies or the other linguistic resources in terms of dependency in domains, concepts, hierarchies, schemata, and semantics as they are based on context. For example, semantically opposite words (antonyms), but conceptually similar (e.g. death place - birthplace) are represented closely in the vector space, whereas in an ontology, a thesaurus, or a taxonomy they would not be so closely linked, if linked at all. Also, word embeddings can find relations among phrases, whereas phrases might be completely absent in the case of ontologies or thesauri. Therefore, word embeddings can find any kind of semantic relation as long as there is word co-occurrence in the corpora, whereas the corpus-independent resources require an explicit linguistic relation existing in their knowledge structures [125]. This shows that word embedding models could be used to tackle the semantic interoperability issues among systems and form relaxed mappings between user queries and data semantics. The latter would also prove useful regarding the representational and content strictness of complex SPARQL queries as approximate queries could be introduced that can still return relevant search matches. Another advantage is that the lack of a specific structure, in contrast to ontologies, makes them bereft of any specific schemata constraints. This flexibility on schemata and semantics makes them representationally-decoupled; therefore, no standardisation is needed for the effective processing of heterogeneous IoT data that comes from multiple sources. It is also important that word embeddings do not rely on the presence of an expert to construct them as they are created automatically based on text-corpora that cover a range of topics. Overall, the domain-independence, representational decoupling, contextual-awareness, and lack of domain experts make word embedding models better candidates for IoT environments. On the other hand, the training of these models can be computationally expensive, but it can be done offline when real-time requirements are needed. Also, the quality and accuracy of a model are highly dependent on the corpora it has been trained on; therefore, some words, phrases, or even contexts might be missing [125].

2.6 The Role of Data Summarisation and Approximation

Data summmarisation and approximation, in general, have the aim to create more succinct representations of the original data to: 1) provide a more useful subset of the whole data to the user based on specific requirements like informativeness, diversity and coverage, popularity and centrality, and 2) boost the performance of systems as fewer resources are used, especially in distributed systems like Pub/Sub, where data moves from one node to another. These approaches are characterised by a trade-off between system performance and data or user quality as there is always an error or loss of information involved.

2.6.1 Entity Summarisation

The information on knowledge graphs regarding entities is so vast and versatile that when seen as a whole it can overwhelm both users and systems' performance. It is more advisable to focus on a single entity at a time and explore its multitude of individual and shared relations [17]. However, simple entity-based keyword queries result in abundant information as in a knowledge graph a range of different levels of information exists that could refer to importance, concepts, and contexts. This volume will be difficult to be handled by the users [136] and on top of that, personal bias and experience can play an important role in how a user perceives this diverse, and voluminous information. In applications like IoT, where there is a high level of heterogeneity, and conceptual and contextual diversity as well as volume regarding entity information, it is important to find processing methodologies that can analyse effectively and efficiently all the available information even when presented in dynamic, and rich unified structures like knowledge graphs. Entity summaries could be used, in this case, as a subset or a high-level inference of the whole entity information is selected [41, 53, 114].

Summaries, in general, can be used in data integration purposes and can assist in the necessitation of the definition of a mapping between different sensor schemata and semantics [137]. Summaries can also help with reducing the abundance of information that is sent to the users as only a representative subset is created, hence users will not be overwhelmed. Query processing over summaries can improve the efficiency of a resource-constraint processing system when it comes to memory, network overhead, or further processing speed, assuming the error range is small [41]. The size of the summary can act as a trade-off between effectiveness and efficiency [136]. Specific types of summaries like diverse ones may also help with the subjective contextual interpretations of humans as they create a representative subset that focuses on a more diverse coverage of the whole information generated for an entity. Therefore, summaries can beneficially contribute to resolving the aforementioned IoT challenges.

There are two strategies in summaries [17]:

Extractive Summary: The summary is a subset of the original information provided. In entity summarisation, all graph nodes and their links/relations are taken by the original source graphs.

Abstractive Summary: The summary differentiates from the original information provided. In entity summarisation, either new graph nodes and their links/relations are created or a high-level interpretation message is provided, for example, through aggregation.

There are two types in entity summaries [17]:

Relevance-based Entity Summary: The summary is focused on specific conditional rules that users define like specific graph nodes or relations among nodes. Repetitions could be observed in this summary when specific relations or nodes are deemed important to the user's preference.

Diverse Entity Summary: The summary is focused on covering a wide range of the available information about an entity (i.e. conceptual and contextual diversity). Repetitions are avoided in this case and unique information pieces on an entity are preferred.

The thesis focuses on diverse entity summarisation as it is deemed more suitable for heterogeneous IoT environments with a multitude of information. It is used to provide a summarised expressive information that has a high user and data effectiveness, and can provide system efficiency. An example of diverse summarisation for the entity Marie_Curie is given in Fig. 2.10, where the top-5 conceptually and contextually diverse information is shown from the original one (Fig. 2.8) with no repetitions.

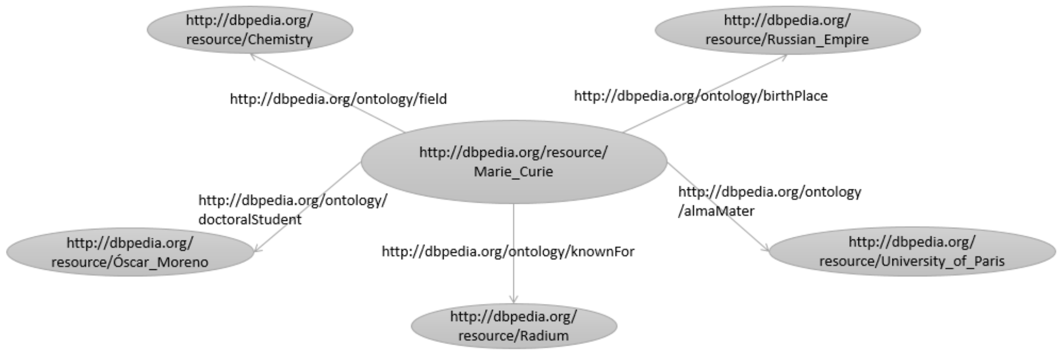

http://dbpedia.org/resource/Chemistry
http://dbpedia.org/resource/Russian_Empire
http://dbpedia.org/ontology/field
http://dbpedia.org/ontology/birthPlace
http://dbpedia.org/resource/Marie_Curie
http://dbpedia.org/ontology/doctoralStudent
http://dbpedia.org/ontology/almaMater
http://dbpedia.org/resource/Óscar_Moreno
http://dbpedia.org/ontology/knownFor
http://dbpedia.org/resource/University_of_Paris
http://dbpedia.org/resource/Radium

Figure 2.10: Diverse entity summarisation of entity Marie_Curie.

Entity summarisation should not be confused with related types of summarisations [17, 53] like: 1) ontology summarisation with the aim to summarise concept descriptions, 2) text/document summarisation that focuses only on unstructured textual data rather than structured RDF data, with the latter being considered more challenging due to its sparseness, shortness, less contextual information, and the appearance of words that might make no sense [51], and 3) graph summarisation [138], which mostly emphasises on summarising structures of a graph that are not star-shaped (i.e. graphs that do not contain multi-hop links) and their edges may not even be labelled (e.g. summarisation is not based on semantics).

2.7 Discussion

The two motivational scenarios for the domains of Healthcare and Smart Cities present an abundance of challenges and requirements that need to be addressed. Existing event-based systems, Semantic Web technologies and data stream processing solutions cannot address all of the aforementioned IoT issues.

Current event-based systems can ease efficiency requirements (R6) by providing scalable and distributed architectures as well as solutions for timeliness and resource-constraint challenges. However, they are bound by data and user challenges. Specifically, Pub/Sub systems are decoupled in space, time, and synchronisation (R6), but they have low to medium decoupling traits regarding usability (R1) and user expressibility (R2) as well as data expressiveness (R3) that affects effectiveness (R4, R5). The topic-based Pub/Sub has highly usable and representationally-decoupled subscriptions (R1), but since the subscriptions are of low expressiveness, then, redundant and irrelevant notifications are provided that may overload systems and users. The content-based Pub/Sub has subscriptions of higher expressiveness, but at the expense of representational coupling and complexity as well as necessitation of user expertise. Also, the notifications are not expressive due to the possibility of redundancy and duplication. The same limitations apply to graph-based Pub/Sub with the difference that an even higher subscription expressiveness is provided that enhances the issues regarding representational coupling, complexity, and user expertise. Also, although richer information is generated (R3), the notifications may still contain redundancy and duplication. CEP systems, on the other hand, may focus on efficient processing and communication of single or complex events as well as high, but complex, query expressiveness, but they suffer from a lack of standardisation in architectures, data models, rule languages, processing methodologies, and vocabularies as well as necessitation of user expertise, complexity, and lack of reasoning or knowledge background integration capabilities. These systems have not been developed with simplicity (R1, R2) and expressiveness (R3) in mind. Finally, data stream processing models may analyse continuous streams of information and provide expressive data (R3) through reasoning or knowledge background integration capabilities, but they also suffer from a lack of standardisation in models, languages, and vocabularies as well as necessitation of user expertise. Also, their efficiency capabilities are of lower quality compared to the ones of Pub/Sub or CEP.

Semantic Web technologies along with data stream processing are used in IoT architectures and methodologies to provide semantically enriched information and deal with interoperability issues (R3). Specifically, knowledge graphs and ontologies are used to represent real-world or abstract things with high conceptual and contextual diversity. Ontologies are formal conceptualisations of particular domains, which describe the content, characteristics, and taxonomies/hierarchies regarding a topic. They have a high expressive power when it comes to describing complex entities or concepts in a detailed level with a specific structure (R3). However, they pose many limitations regarding usability (R1), user expressibility (R2), more advanced data expressiveness capabilities (R3) that affect effectiveness (R4, R5), and efficiency (R6). Specifically, ontologies are complex and versatile, and the lack of a standardised and domain-agnostic ontology lead to inefficiency and cumbersome system performance. Also, the possibility of erroneous or unknown information to the users and the necessitation of domain experts will prove ineffective. Another issue is the lack of a fixed vocabulary and schemata describing the same entity, content, or attribute/property, leading to representational coupling that can affect the subscriptions' simplicity and notifications' effectiveness, alike. Also, ontologies usually go hand-in-hand with complex SPARQL queries that are of low usability to non-expert users and suffer from representational coupling. Finally, the fact that ontologies result in conceptually and contextually rich information (R3) does not mean that further advanced analytics related to high-level abstractions, redundancy awareness, and enrichment is not needed that is also efficient. Other linguistic resources are used as standalone or in conjunction with ontologies like thesauri, dictionaries, and taxonomies, but they suffer from domain dependency, and lower semantic expressiveness and accuracy than ontologies.

2.8 Summary

In this chapter, two motivational scenarios are analysed for the domains of Healthcare and Smart Cities. In the case of Healthcare, a doctor is interested in a summarised view of a patient's health status from data deriving from real-time sensors checking physiological conditions as well as medical history and living conditions of the patient. In the case of Smart Cities, a real estate agent is interested in summarised information regarding a property from data deriving from real-time internal or external sources and information regarding the region the property is situated in.

Both motivational scenarios pose a number of challenges that need to be addressed. Specifically, data heterogeneity leads to representational coupling issues, whereas redundancy leads to voluminous unnecessary data that overwhelms users and systems, alike. Also, data needs to be enriched by external static sources to provide more complementary contextual information to the users as well as interpreted at a high-level since raw numerical sensor data is of no value. Notifications should also be provided in a clear and understandable structure to the users, and all users should be facilitated irrelevant of technical expertise or level of data understanding by providing contextually-aware, representationally-decoupled queries of low complexity that covers as much complete and diverse information provided by the sources as possible. Finally, the volume and dynamism of sensors as well as the continuity and velocity of data involved lead to the challenges of scalability, timeliness, and resource constraints that need to be resolved so that the users are provided with up-to-date and real-time information.

Existing event-based systems, Semantic Web technologies and data stream processing solutions cannot address all of the aforementioned IoT challenges. Pub/Sub systems may ease challenges related to scalability, timeliness, and resource constraints, but they have different capabilities in regards to data and user challenges. CEP systems are efficient and have high query expressiveness, but they are even more complex than Pub/Sub and lack advanced data analytics capabilities. Data stream processing models are more advanced in terms of analytics, but they are not as efficient as Pub/Sub and CEP. Approaches using ontologies, thesauri, dictionaries, and taxonomies may create conceptually and contextually rich information, but at the expense of complexity, inefficiency, representational coupling, low usability, and domain dependency. On the other hand, Deep Learning word embeddings models are superior in tackling interoperability as they do not suffer from dependency in domains, concepts, hierarchies, schemata, and semantics as they are based on context. Also, entity summarisation and approximation can assist with data expressiveness, resource constraints, and system efficiency of IoT environments through extractive or abstractive methodologies, even though they are characterised by a trade-off between system performance and data or user quality.

Chapter 3

Entity-centric Publish/Subscribe Summarisation System

3.1 Introduction

The rise of the Internet of Things (IoT) results in the creation of a wide range of entity-centric data coming from multiple sources and end-users interested in its real-time analysis and high-level states regarding applications like Healthcare and Smart Cities. Nevertheless, IoT environments possess multiple challenges regarding data, users, and systems like data heterogeneity, non-technical users, and scalability, to name a few. Suitable data dissemination paradigms need to be defined that can overcome these challenges and, at the same time, provide expressive notifications to users, effectively, but not at the expense of low usability or high resource consumption and processing time. Publish/Subscribe systems (Pub/Sub) can efficiently realise some of these requirements as they are suitable communication paradigms for dynamic large scale applications since apart from abstracting the users from the underlying communication complexity, they are scalable, distributed, and decoupled in space, time, and synchronisation. However, Pub/Sub systems need to be extended with more sophisticated methodologies to address data and user challenges regarding simplicity and usability along with interoperability and heterogeneity.

This chapter explores the requirements and research questions, and presents a new entity-centric Pub/Sub scheme, which combines the advantages of topic-based and graph-based Pub/Sub schemes (Fig. 3.1). Specifically, the proposed system, which is called *Entity-centric Publish/Subscribe Summarisation System*, shows how methodologies like windowing, data fusion, high-level abstraction, approximation, conceptual clustering, top-k ranking, representationally-decoupled subscriptions, and graph-based notifications can result in expressive entity summaries of high quality using limited resources and processing time at no expense of usability.

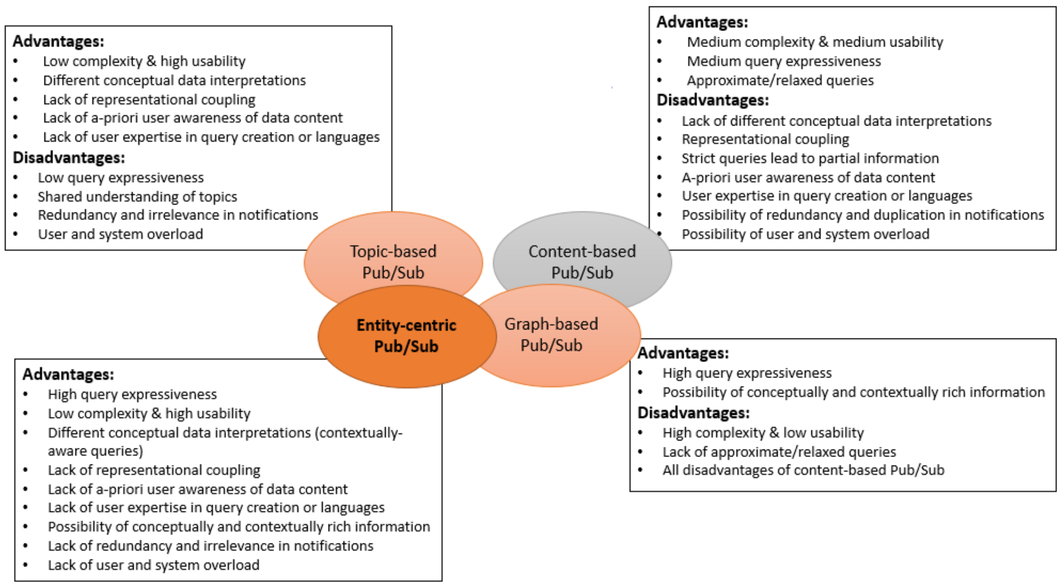

Advantages:
Low complexity & high usability
Different conceptual data interpretations
Lack of representational coupling
Lack of a-priori user awareness of data content
Lack of user expertise in query creation or languages
Disadvantages:
Low query expressiveness
Shared understanding of topics
Redundancy and irrelevance in notifications
User and system overload
Advantages:
Medium complexity & medium usability
Medium query expressiveness
Approximate/relaxed queries
Disadvantages:
Lack of different conceptual data interpretations
Representational coupling
Strict queries lead to partial information
A-priori user awareness of data content
User expertise in query creation or languages
Possibility of redundancy and duplication in notifications
Possibility of user and system overload
Topic-based Pub/Sub
Content-based Pub/Sub
Entity-centric Pub/Sub
Graph-based Pub/Sub
Advantages:
High query expressiveness
Possibility of conceptually and contextually rich information
Disadvantages:
High complexity & low usability
Lack of approximate/relaxed queries
All disadvantages of content-based Pub/Sub
Advantages:
High query expressiveness
Low complexity & high usability
Different conceptual data interpretations (contextually-aware queries)
Lack of representational coupling
Lack of a-priori user awareness of data content
Lack of user expertise in query creation or languages
Possibility of conceptually and contextually rich information
Lack of redundancy and irrelevance in notifications
Lack of user and system overload

Figure 3.1: Advantages of the entity-centric Pub/Sub scheme.

The proposed system emphasises on day-to-day end-users (e.g. doctor) that are not involved in the design of the system so they have no information regarding the system's architecture, data representations or the content provided by the available publishers. Also, they have no knowledge in query languages to create a targeted subscription and they want to be notified with unbiased and conceptually diverse entity information (lack of redundancy) by providing simple subscriptions with minimal configurations that lead to high expressibility. Experts in query languages end-users (e.g. developers) could also avail of the proposed system; nevertheless, for more targeted information, other existing types of Pub/Sub could be used.

The rest of the chapter is as follows: In Section 3.2 the requirements and research questions are analysed that are explored by the proposed system, whereas in Section 3.3 a high-level description of the proposed system is provided that maps the challenges to the different system components. Section 3.4 describes the main architecture of the system by providing more details of its components, their flow among them and how they map to the challenges, requirements, and research questions, while Section 3.5 concludes and summarises the chapter.

3.2 Core Requirements and Research Questions

The proposed *Entity-centric Publish/Subscribe Summarisation System* explores the following core requirements:

- **R1: Usability**: refers to a system with high usability, which can be used easily by all users independent of representational coupling, query language expertise, system knowledge, bias, and background knowledge.
- **R2: User Expressibility**: refers to the ability of users to understand their data needs at a satisfying level and express them by simple subscriptions of high usability with minimal configuration settings.
- **R3: Data Expressiveness**: refers to a domain-agnostic system that can tackle interoperability and heterogeneity by providing rich notifications that contain conceptual and contextual diversity or high-level abstractions.
- **R4: User Effectiveness**: refers to a system that provides notifications of high quality according to the users' needs.
- **R5: Data Effectiveness**: refers to a system that provides redundancy-aware and expressive notifications of high quality according to the wide range of concepts and contexts.
- **R6: Efficiency**: refers to a system that is efficient in terms of memory, processing time, throughput, and scalability.

These requirements are formulated into the following research questions:

RQ1: Can a Pub/Sub be created that offers usability *(R1)* while addressing users' expressibility *(R2)* effectively *(R4)* and efficiently *(R6)*?

- **RQ1.1**: can a simple abstract representationally-decoupled subscription be defined that relies on expert and non-expert users alike *(R1)*?
- **RQ1.2**: can a subscription be defined that its notification will cover a range of different human interpretations independent of its complexity and with minimal configuration settings *(R2)*?

- **RQ1.3**: can a subscription be defined that its notification will not overwhelm the users with data overload *(R2)*?
- **RQ1.4**: can the satisfaction of users based on the received notifications be evaluated according to: (1) how well they address the users' different needs and interpretations *(R4)*?, (2) how much they reduce the information overload to the users *(R4)* and to the system *(R6)*?

RQ2: Can a Pub/Sub be created that offers expressiveness of heterogeneous data *(R3)* effectively *(R4, R5)* and efficiently *(R6)*?

- **RQ2.1**: can a methodology be defined for integrating data from multiple sources *(R3)*?
- **RQ2.2**: can an appropriate publication structure of integrated data be defined that is also understandable to the users *(R3)*?
- **RQ2.3**: can a methodology be defined for semantically abstracting integrated data and providing rich notifications with conceptual and contextual diversity or high-level abstractions independent of domain *(R3)*?
- **RQ2.4**: can the semantically-abstracted rich notifications be evaluated according to: (1) how well they cover the wide range of different concepts and contexts *(R5)*?, (2) how well they reduce information redundancy without sacrificing important information *(R5)*?, 3) how much they boost the system's performance *(R6)*, 4) how many dependencies are needed (e.g. domain experts, external ontologies, memory-heavy models etc.) *(R4, R5, R6)*?

3.3 Overview of the System

Data heterogeneity, in IoT, results in representational coupling issues and redundancy related to duplication and conceptual similarity. These challenges burden the users and the system performance, alike, since a pool of unnecessary data is created in resource-constrained environments that can overwhelm the users. Data dissemination paradigms need to be defined that are representationally-decoupled to cater for the diverse set of schemata and semantics in order to achieve interoperability, discard redundant information, and be efficient. This chapter's proposed Pub/Sub system addresses these issues by observing the different data representations to create diverse entity summaries that cover a wide range of the available concepts and contexts (*Entity Summarisation* component). Necessary information is also provided by representationally-decoupled subscriptions (*Subscription Model* component).

The necessitation of a paradigm that provides data enrichment and high-level interpretations is also important. As IoT environments contain a range of heterogeneous devices and user needs go beyond raw data, more mechanisms are needed that integrate and enrich data from multiple sources to create rich entity-centric information with their diverse or abstractive states in terms of concepts and contexts. The proposed Pub/Sub system addresses these issues by presenting a number of windowing policies that fuse data from multiple sources according to their entity (*Data Integration* component) and by turning the original information into more understandable high-level abstractions or abstractive entity summaries (*Entity Summarisation* component) through representationally-decoupled methodologies. The type of windowing policies and the abstraction is also supported by subscriptions (*Subscription Model* component).

It is essential for the diverse and abstractive entity summaries to be provided in a user-friendly data representation that is easily understood by the users and substantially covers all of the rich information in a unified way. Also, a paradigm should cater for all of the aforementioned challenges and solutions for non-technical users, as well. This means that users should not be obliged to provide complex expressive subscriptions to address the above needs. The proposed Pub/Sub system overcomes these challenges by providing graph-based notifications that cover the entities and their states (*Event Model* component) and requiring simple, contextually-aware, and representationally-decoupled subscriptions (*Subscription Model* component) of high usability.

The proposed paradigm should also address challenges regarding scalability, timeliness, and resource constraints since the data volume, dynamism, continuity, velocity, variety, and analytics needs become more complex and advanced by the day. Since it extends a Pub/Sub system (*Entity-centric Publish/Subscribe Summarisation System*), it can efficiently resolve these issues by providing a platform where multiple dynamic sources and continuous data with varying velocities are managed through windowing policies (*Data Integration* component), whereas data variety, analytics, real-time processing requirements, and network overhead are managed by effective and efficient extractive and abstractive entity summaries that cover a wide range of the available variety (*Entity Summarisation* component).

The final important aspect of every proposed data dissemination paradigm is its evaluation. All components of a system and their outcomes need to be evaluated in order to examine whether specific challenges and requirements have been addressed. The proposed Pub/Sub system contains an extensive evaluation methodology where multiple ground truths and metrics have been newly formalised, adopted, or adapted in order to observe how effective and efficient the system is (*Offline Evaluation* component).

3.4 Architecture and Components of the System

The high-level architecture of the proposed *Entity-centric Publish/Subscribe Summarisation System* is illustrated in Fig. 3.2. It is inspired by typical Pub/Sub architectures that contain their own subscription and event models, a matcher that could be from exact (i.e. boolean) to more advanced (e.g. approximate), a mechanism for data integration (e.g. windows) in advanced architectures, a data analytics component (e.g. entity summarisation, in this case), and an evaluation component that analyses, offline, the performance of the system. All these components map to different research questions and upcoming Chapters 4, 5, and 6, where they are analysed in detail.

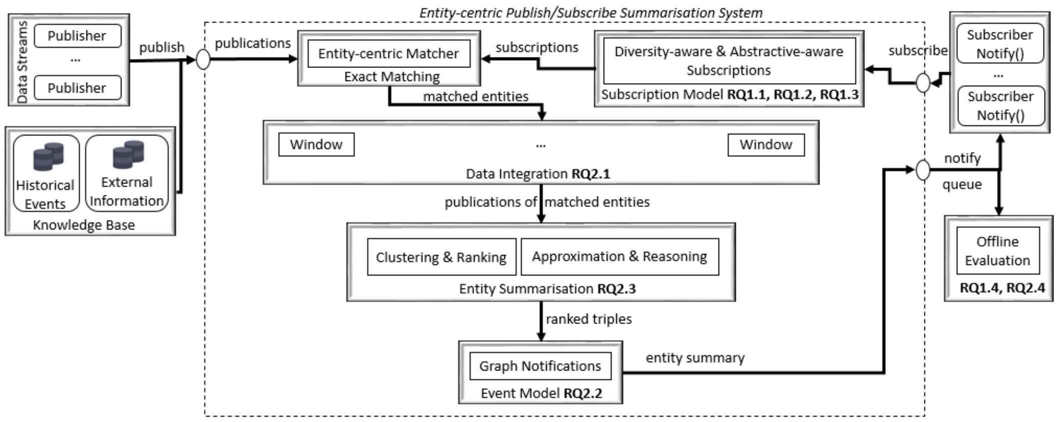


Figure 3.2: The high-level architecture of the Entity-centric Publish/Subscribe Summarisation System.

As depicted in Fig. 3.2, publishers generate publications concerning different entities coming from sensors or static sources, while subscribers generate diversity-aware and/or abstractive-aware subscriptions (*Subscription Model*), where one is interested in an extractive and/or abstractive summary of an entity. All publications and subscriptions enter the *Entity-centric Publish/Subscribe Summarisation System*, where they are stored and managed. The stored publications and subscriptions are examined by the *Entity-centric Matcher.* If there is a match between the entity requested by the subscribers and the entity to which publications refer, then, a match occurs (*Exact Matching*). The publications of the matched entities are integrated into windows, where they are incrementally processed (*Data Integration*). The processing involves the approximation, reasoning, clustering, and ranking of the publications in order to create extractive and/or abstractive entity summaries (*Entity Summarisation*) by choosing one of the three components; PubSum, IoTSAX, or PoSSUM Approach. Once the processing has been finalised, the notification process is triggered that checks the state of all subscriptions in order to notify all interested parties. The notifications are extracted in the form of entity summaries based on the ranked publications/triples (*Event Model*) and are sent to the subscribers and to a queue that along with necessary metadata will be stored externally in order to be evaluated offline *Offline Evaluation.* This is a continuous process, which ends once a pre-defined time duration has been reached.

An overall map among challenges, requirements, research questions, components, solutions, and chapters is presented in Table 3.1 and Table 3.2, which is analysed in more detail in the following components.

3.4.1 Data Integration Component

This component addresses the following research question:

RQ2.1: Can a methodology be defined for integrating data from multiple sources *(R3)*?

The vast volume of sources creates an abundance of real-time data. Stream data management systems and triple stores are not suitable structures to store temporarily or permanently data since the processing time, accessing of data, and memory burdens will result in a delayed and cumbersome system. Windows are more suitable structures as they temporarily store newly generated data that can be processed incrementally without delays. Also, windows, contrary to typical Pub/Sub, give the opportunity for the events not to be seen as separate chunks

Table 3.1: Overal map among challenges, requirements, research questions, components, and chapters

Challenges	Requirements	Research Questions	Components	Chapters
Data Challenges				
Heterogeneity	R2 - User Expressibility	RQ1.2	Subscription Model	4, 6
	R3 - Data Expressiveness	RQ2.3	Entity Summarisation	
	R4 - User Effectiveness	RQ1.4	Offline Evaluation	
	R5 - Data Effectiveness	RQ2.4		
Redundancy	R2 - User Expressibility	• RQ1.2 • RQ1.3	Subscription Model	4, 6
	R3 - Data Expressiveness	RQ2.3	Entity Summarisation	
	R4 - User Effectiveness	RQ1.4	Offline Evaluation	
	R5 - Data Effectiveness	RQ2.4		
Data Enrichment	R2 - User Expressibility	RQ1.2	Subscription Model	4, 5, 6
	R3 - Data Expressiveness	RQ2.1	Data Integration	
User Challenges				
High-level Interpretation	R2 - User Expressibility	RQ1.2	Subscription Model	5, 6
	R3 - Data Expressiveness	RQ2.3	Entity Summarisation	
	R4 - User Effectiveness	RQ1.4	Offline Evaluation	
	R5 - Data Effectiveness	RQ2.4		
User-friendly Data Representation	R3 - Data Expressiveness	RQ2.2	Event Model	4, 5, 6
Non-technical Users	R1 - Usability	RQ1.1	Subscription Model	4, 5, 6
	R2 - User Expressibility	RQ1.2		
System Challenges				
Scalability	R6 - Efficiency	• RQ1.4 • RQ2.4	• Pub/Sub • Data Integration • Entity Summarisation • Offline Evaluation	4, 5, 6
Timeliness and Resource Constraints				

Table 3.2: Overal map among challenges, research questions, and solutions

Challenges	**Research Questions**	**Solutions**
		Data Challenges
Heterogeneity	RQ1.2	Diversity-aware subscriptions
	RQ2.3	Diverse entity summarisation with the use of word embedding models, conceptual clustering, and top-k ranking
	RQ1.4	Summary quality and diversity consensus based on ground truth from judges
	RQ2.4	• Construction of conceptual and relevance ground truths • F-score of summary
Redundancy	• RQ1.2 • RQ1.3	Diversity-aware and abstractive-aware subscriptions that express the desired size of summary
	RQ2.3	Diverse entity summarisation with the use of word embedding models, conceptual clustering, and top-k ranking
	• RQ1.4 • RQ2.4	• Construction of conceptual and relevance ground truths • F-score and duplication reduction of summary
Data Enrichment	RQ1.2	Subscriptions that express the kind of windowing policy
	RQ2.1	Fusion of data coming from multiple sources to entity-based windows
		User Challenges
High-level Interpretation	RQ1.2	Abstractive-aware subscriptions
	RQ2.3	Abstractive entity summarisation with the use of data approximation and approximate reasoning rules with the use of word embedding models
	• RQ1.4 • RQ2.4	F-score of reasoning and approximation error of summary
User-friendly Data Representation	RQ2.2	RDF graphs that represent conceptual entities with their associated extractive and/or abstractive properties or background information
Non-technical Users	• RQ1.1 • RQ1.2	Expressive and representationally-decoupled subscriptions that need only a-priori knowledge of the entity name from the users
		System Challenges
• Scalability • Timeliness and Resource Constraints	• RQ1.4 • RQ2.4	• Pub/Sub system that fuses unbounded data coming from dynamic sources to entity-based windows, which are processed incrementally and in batch to create extractive and abstractive entity summaries • Evaluation metrics regarding latency, memory, message reduction, throughput, compression space-saving, and processing time

of information, but to be combined and form conceptually and contextually rich information without losing important information that breaks the interlink among different pieces of information. In this way, subscribers are presented with a more unified and enriched data picture, which is always up-to-date, instead of themselves integrating the data.

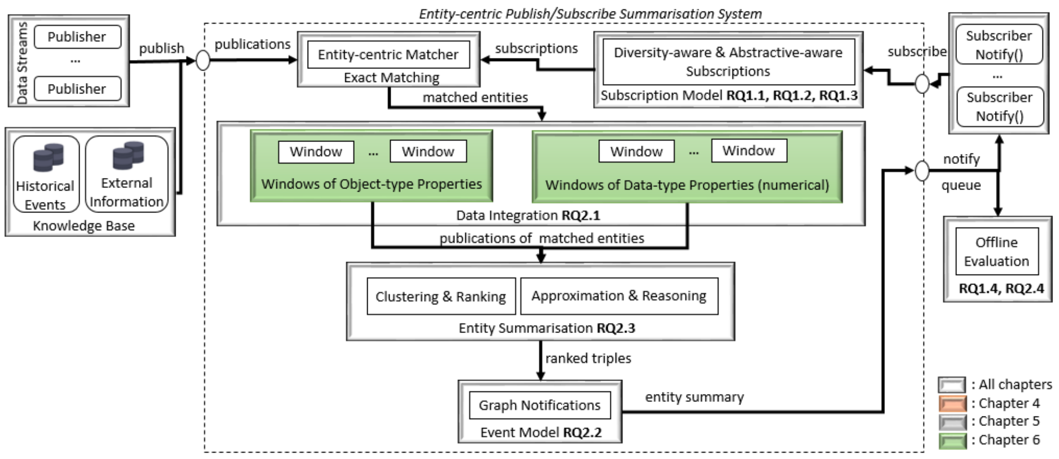


Figure 3.3: The data integration component of the Entity-centric Publish/Subscribe Summarisation System.

The aim of data integration is to group seamlessly data referring to the same entity, which constitutes a match, into windows. As depicted in Fig. 3.3, two windows are created for each entity; one for object-type properties and one for data-type (numerical) properties. As the publications are generated by various sources, they are assigned into a corresponding window depending on their entity and data type. The moment a window is populated, it is processed incrementally and in batch, that is when the window has reached its full capacity based on a subscription, dependent on the type of processing by the *Entity Summarisation* component. Once the processing has been finalised, the window is flushed and awaits new elements. Several windowing policies are supported including Count Tumbling Window, Count Sliding Window, Time Tumbling Window, and Time Sliding Window. More details are given in Chapter 4.

The data integration component focuses only on the suitable structure and the policies of integrating data streams regarding the same entities. Other issues such as noise, outliers and trust or consistency, which are linked to data integration from different sources in uncontrolled environments, are out of scope.

3.4.2 Entity Summarisation Component

This component addresses the following research question:

RQ2.3: Can a methodology be defined for semantically abstracting integrated data and providing rich notifications with conceptual and contextual diversity or high-level abstractions independent of domain *(R3)*?

The plethora of dynamic sources creates an abundance of real-time data that could range from significant to redundant information (duplicates or conceptually similar data). Especially, when entity graphs are considered, their complexity and dynamic nature could lead to numerous nodes and edges. Analysing the data and transforming it into a much smaller set (summary) that contains an expressive and diverse set of important information could not only improve the quality of notifications that the subscribers receive but the system efficiency, overall, in terms of propagation and storing of data. Approaches like summarisation, approximation, and high-level abstraction could facilitate this goal, that is providing extractive and abstractive entity summaries from which subscribers can form different conceptual and contextual interpretations.

Diverse entity summarisation extracts a useful subset of the available entity data based on informativeness, diversity and coverage, popularity and centrality. Existing approaches are static and come from the Semantic Web; therefore, they emphasise on effectiveness rather than efficiency. The approaches address the data heterogeneity and increase the quality of the extracted summary by providing solutions with the use of ontologies, thesauri, and dictionaries. Nevertheless, these are representationally-coupled solutions and may significantly burden the overall system efficiency. Hence, this component focuses on detecting different data representations (e.g. "humidity" vs "atmospheric humidity") with the use of word embedding models since they are characterised by domain-independence, representational decoupling, contextual-awareness, and lack of domain experts compared to the strict ontologies or the other linguistic resources. Then, conceptually and contextually related data is grouped in conceptual clusters and a top-k ranking decides which data from each cluster will be included in the summary. In this way, the whole entity information is reduced in a summary of k size, where k can significantly affect the trade-off between accuracy (effectiveness) and time/space complexity (efficiency) since valuable information may be lost or an error may occur. The entity summarisation only revolves around data presented as graphs (e.g. triples) that could be updated in real-time based on changes or corrections of the available information coming either from real-time sensors or external sources. Other data types, including text, video, and audio that could be linked with entity information, are out of scope.

High-level abstraction entails an abstractive entity summary that is formed by data approximation and reasoning. Approximation is highly used in resource-constraint environments and it provides acceptably smaller and quicker notifications provided the error range is low [41]. Reasoning, on the other hand, is used for extracting high-level inferences, which deduct meaningful information from data, and reducing memory and communication traffic [139]. In this way, the users and the system are not overwhelmed with redundant information and users are not biased based on their expertise. Existing approaches on high-level abstraction are either very computationally-heavy, especially, for complex environments like IoT that need fast processing while using limited resources, or they rely on representationally-coupled reasoning rules or ontologies. Hence, this component focuses on efficient data approximation as well as semantic interoperability among data and reasoning rules with the use of flexible word embedding models to create semantically approximate rules. This results in the rules being able to be shared and reused among different systems and for different application needs. In this component, the approximate reasoning rules are used for on-the fly-inference of data that has been symbolically approximated with statistically-inspired approaches. In this way, the whole entity information is reduced in an abstractive summary, which contains the behavioural shape of segments of data and their high-level inferences.

The Entity Summarisation component consists of three sub-components; PubSum, IoTSAX, and PoSSUM Approach, analysed below.

PubSum Approach Component

The aim of the PubSum Approach is to create diverse extractive summaries of heterogeneous Linked Data, that is triples with entities as objects (object-type predicates). It is achieved by processing three main elements (Fig. 3.4); *Embedding-based Triple Vectors*, *DBSCAN Clustering*, and *Geometric Ranking*. Embedding-based Triple Vectors is responsible for transforming triples, within a window, into vectors with the use of word embedding models and by extracting the typing information of the triples' objects. Density-based Spatial Clustering of Applications with Noise (DBSCAN) Clustering, then, partitions these vectors into conceptual clusters by finding dense regions in the vector space. The triples in these clusters are ranked by Geometric Ranking based on importance and diversity. More details are given in Chapter 4.

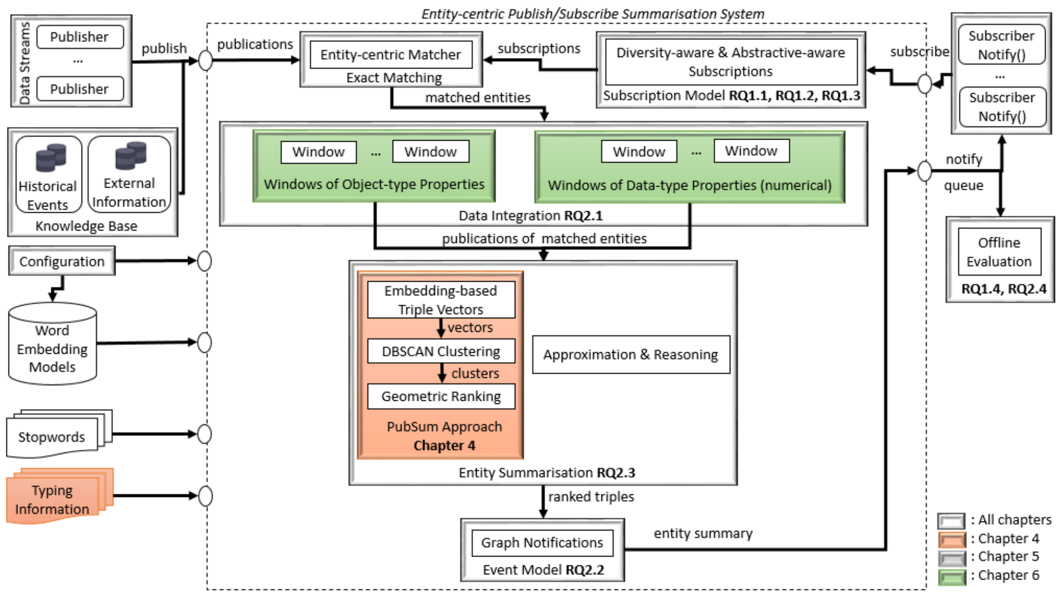


Figure 3.4: The PubSum approach component of the Entity-centric Publish/Subscribe Summarisation System.

IoTSAX Approach Component

The aim of the IoTSAX Approach is to create abstractive summaries of numerical data, that is triples with numerical objects (data-type predicates). It is achieved by processing two main elements (Fig. 3.5); *Data Approximation* and *Reasoning*. Data Approximation is responsible for transforming a large collection of numerical values regarding an attribute (i.e. predicate in triples), within a window, into a concise symbolic representation. Each symbol in the representation is related to a mean value of different temporal numerical sequence segments. Two algorithms are proposed; *Dynamic PAA* that calculates the mean values and defines the segments, and *Dynamic SAX* that constructs the symbolic representation. Reasoning, on the other hand, is responsible for interpreting the temporal pattern or shape of the symbolic representation (e.g. increasing, decreasing etc.) and extracting a high-level inference of the average value of all the numerical values regarding the attribute within a window. The interpretation is accomplished by the *Approximation Interpretation* methodology, while the high-level inference is achieved by the *Approximate Reasoning Rules* algorithm, which is based on semantically related reasoning rules with the use of word embedding models. More details are given in Chapter 5.

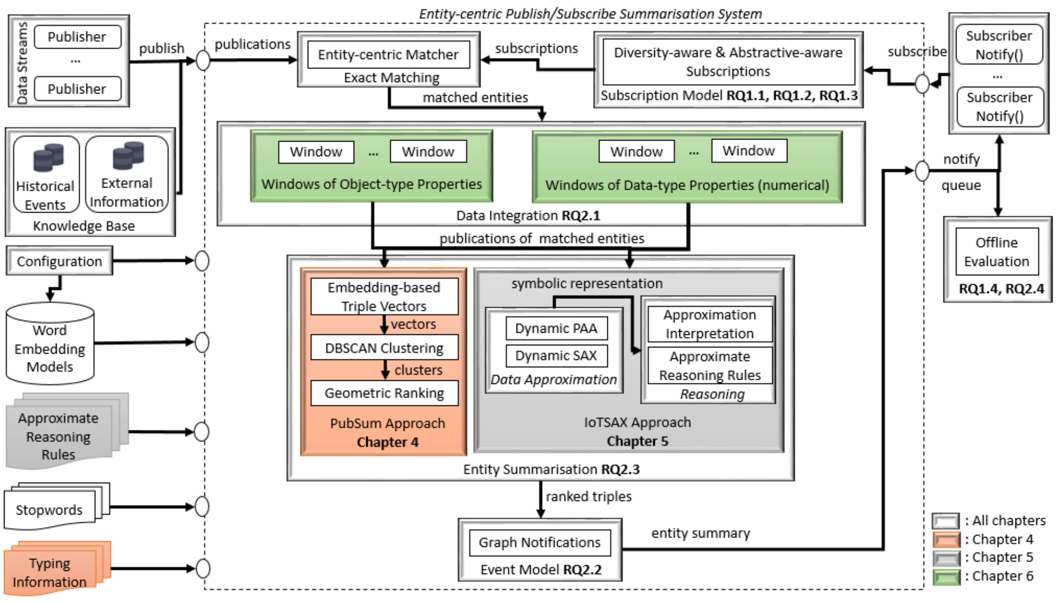


Figure 3.5: The IoTSAX approach component of the Entity-centric Publish/Subscribe Summarisation System.

PoSSUM Approach Component

The aim of the PoSSUM Approach is to create diverse extractive and abstractive summaries of heterogeneous enriched data, that is triples with entities and numerical values as objects (object-type and data-type predicates). It is achieved by processing three main elements (Fig. 3.6); *Embedding-based Triple Vectors*, *DBVARC Clustering*, and *Triple2Rank*. Embedding-based Triple Vectors is responsible for transforming triples, within a window, into vectors with the use of word embedding models and by extracting the typing information of the triples' objects (only for object-type predicates). Density-Based VARiance Clustering (DBVARC) Clustering, then, partitions these vectors into conceptual clusters in two phases; one incremental, where regions of close proximity are observed, and one batch, where the previous clusters are refined based on intra-connected and inter-connected density. The triples in these clusters are ranked by Triple2Rank based on importance, informativeness, and diversity. In the case of windows containing data-type (numerical) properties, the high-level inference extracted from *Reasoning* is used for ranking. More details are given in Chapter 6.

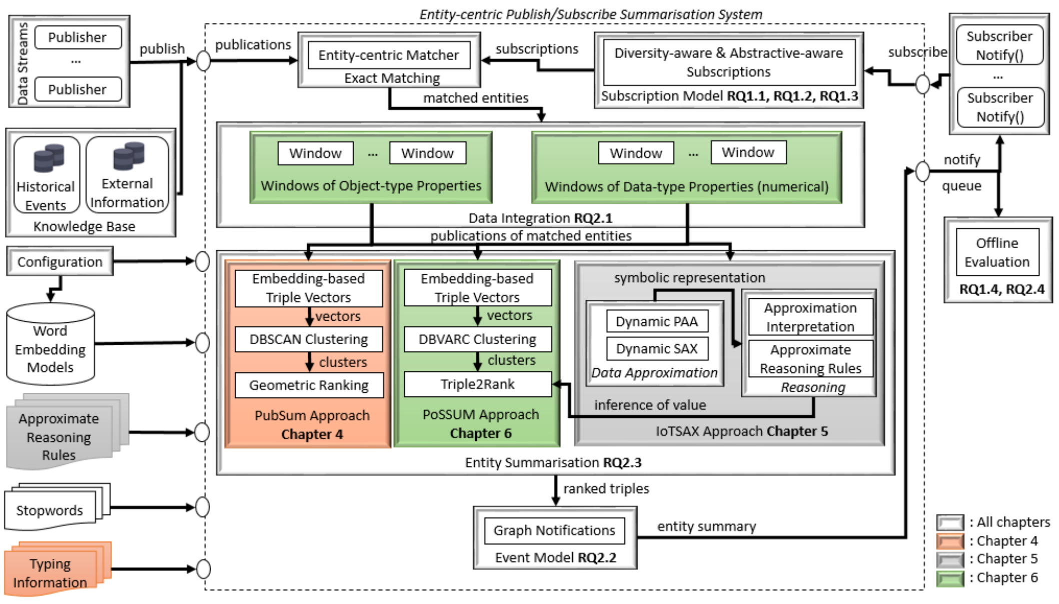


Figure 3.6: The PoSSUM approach component of the Entity-centric Publish/Subscribe Summarisation System.

3.4.3 Subscription Model Component

This component addresses the following research questions:

- **RQ1.1**: Can a simple abstract representationally-decoupled subscription be defined that relies on expert and non-expert users alike *(R1)*?
- **RQ1.2**: Can a subscription be defined that its notification will cover a range of different human interpretations independent of its complexity and with minimal configuration settings *(R2)*?
- **RQ1.3**: Can a subscription be defined that its notification will not overwhelm the users with data overload *(R2)*?

The scheme with the lowest complexity and high usability is the topic-based Pub/Sub since its subscriptions are based on topics, which could be seen as keyword-based queries. Topics could extend to entities that could be seen as descriptions understandable from the human point of view and a more natural way for users to subscribe to, like they would do a search on the Web [122]. Nevertheless, if subscriptions were only based on entities, then, redundant and irrelevant notifications would occur. On the other hand, graph-based Pub/Sub contains SPARQL-like subscriptions that have much higher query expressiveness to present the users with more selective/targeted notifications, which are conceptually and contextually rich. However, these queries are complex for non-experts and suffer from representational coupling as well as a-priori knowledge of the data content. Also, these structured query languages do not scale well to large schemata with a plethora of attributes. Therefore, the proposed subscription model, inspired by both schemes, extends the entity-centric subscriptions by requiring a collection of attribute-value pairs, which are simple, specific, and easy to understand. In this way, the query expressiveness is high enough so that redundancy does not occur while the users are abstracted from the underlying representations and content as they are still only required to be aware of the entity of interest (like in keyword-based queries) and no content filtering is involved.

Another important aspect of the proposed subscriptions is that they are contextually-aware. Since the notifications are extractive and abstractive summaries that cover a wide range of conceptual and contextual information, the users need only define what type of summary they require either in the form of diversity-aware queries or abstractive-aware ones. In this way, users only choose the type of methodology that will be used to cover a range of different human interpretations, while they are abstracted from the specifics concerning these methodologies.

The last aspect is that users can define the type of windowing policy, which dictates how the data will be integrated, and the size of the summaries, which acts as a trade-off between effectiveness and efficiency. Especially when entity graphs are involved, the size could become enormous with numerous nodes and edges due to their complexity and dynamic nature. In this way, the users do not need to create complex join queries and be aware of all the sources related to an entity of interest to integrate and enrich data, and they can decide which data size is suitable so that they are not overwhelmed by overload.

The above characteristics of the proposed subscription model make it suitable for experts and non-experts, alike, without requiring high expressibility, that is expertise in query creation or languages. This low complexity results in a highly usable system that with simple and few parameters allows users to still profit from query expressive power and be provided with as much as possible expressive information derived by the sources through advanced methodologies, which are abstracted from them.

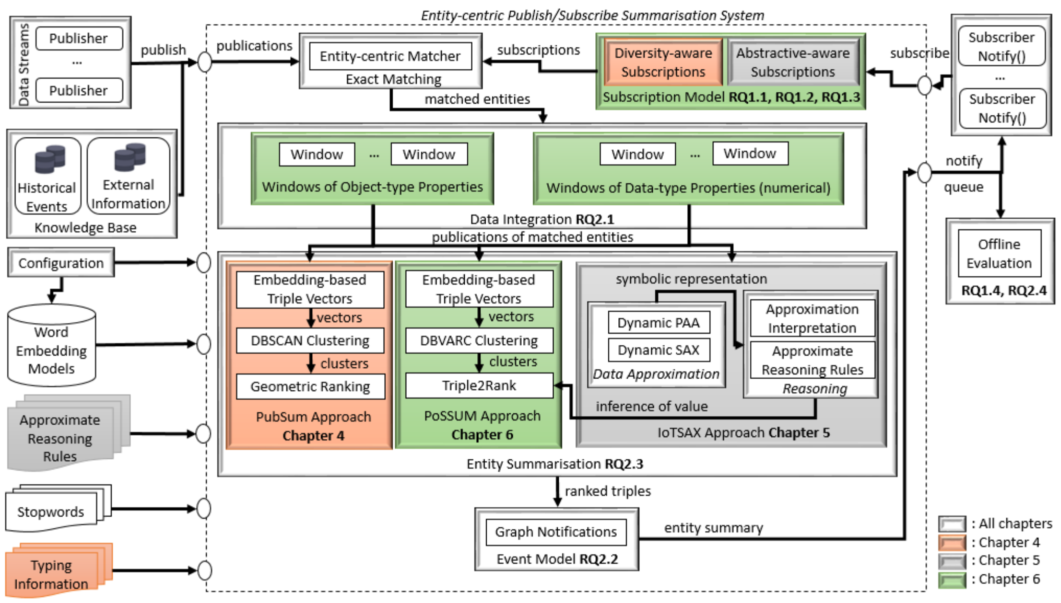


Figure 3.7: The subscription model component of the Entity-centric Publish/Subscribe Summarisation System.

As depicted in Fig. 3.7, the proposed subscription model supports *Diversity-aware subscriptions* and/or *Abstractive-aware subscriptions*, which expect from the users only to be aware of the entity name they are interested in and to provide some additional minimal and simple parameters. These parameters are related to windowing policies, the entity summary type, and the level of filtering that will be involved in the final notification. Therefore, the subscription payload is a set of simple attribute-value pairs and each event needs to fulfil all of its constraints so that it is sent as a notification. Each pair consists of an attribute, an equal operator, and a value. The definition of the subscription model is as follows: Let S be the set of subscriptions, SID the set of subscriber IDs, $SubID$ the set of subscription IDs, T the set of timestamps, ATT the set of attributes, OP the set of operators,

VAL the set of values, and E the set of all entities, respectively, then:

$$\begin{aligned} s \in S \Leftrightarrow s = \Big\{\big(&sID, subID, t, (att, op, val)\big), \ldots : \\ &sID \in SID, subID \in SubID, t \in T, \\ &(att, op, val) \in ATT \times OP \times VAL\Big\} \end{aligned} \tag{3.1}$$

, where $ATT \in \{entity, k, windowType, windowSize, windowSlide, summary\}$, $entity \in E$, $k \in \mathbb{N}$, $windowType \in \{CountTumbling, CountSliding, TimeTumbling, TimeSliding\}$, $windowSize \in \mathbb{N}$, $windowSlide \in \mathbb{N}$, and $summary \in \{Extractive, Abstractive\}$. In Fig. 3.9, an example of a subscription payload that combines diversity and abstractive awareness is given. According to it, a subscriber is interested in an extractive and abstractive summary of the entity Usain_Bolt, that is the top-5 diverse entity information deriving from the analysis of data taken from count tumbling windows of total size 100, that is 100 publications.

3.4.4 Event Model Component

This component addresses the following research question:

RQ2.2: Can an appropriate publication structure of integrated data be defined that is also understandable to the users *(R3)*?

Entities are real-world or abstract things, which could contain information of high conceptual and contextual diversity. This means that simple, narrow and rigid data structures like attribute-value pairs or tuples, which are usually targeted by Pub/Sub systems, are not able to support this kind of complex semantic data. The proposed event model provides a richer, but simple, representation of notifications, in which data coming from different sources concerning an entity can be integrated and represented in a unified format by Resource Description Framework (RDF) graphs. Since these graphs derive from the Semantic Web, they are represented in machine-readable formats, which are compatible with data formats describing knowledge on the Web and they can easily be extended or annotated when new sensors are deployed. Therefore, the graphs are suitable, dynamic, and user-friendly structures for representing conceptual entities with their associated extractive and/or abstractive properties or background information.

The aim of the event model is to construct the final notification that will be

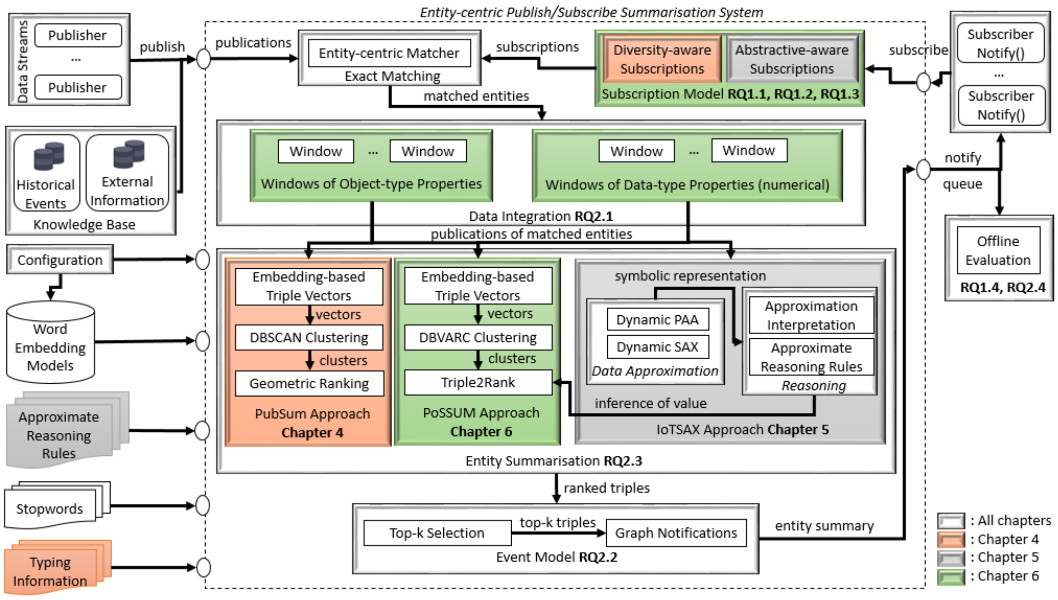
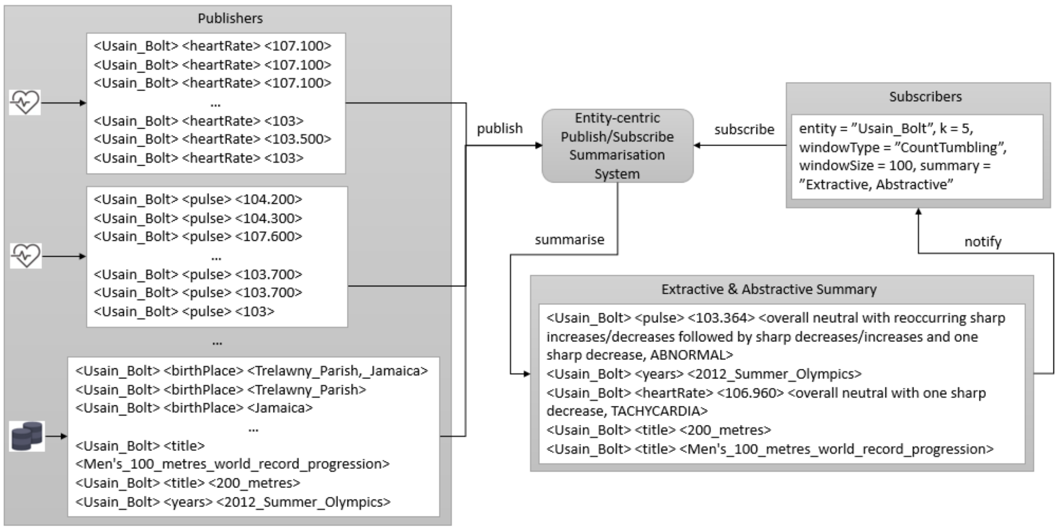


Figure 3.8: The event model component of the Entity-centric Publish/Subscribe Summarisation System.

sent to the interested subscribers. Specifically, in Fig. 3.8, a summary is finalised by performing *Top-k Selection* depending on the corresponding subscriptions' k and it is included in a timestamped *Graph Notification*. The proposed event model supports RDF graphs; therefore, the event payload contains either RDF triples of the form ⟨subject, predicate, object⟩ or RDF quads of the form ⟨subject, predicate, object, context⟩. The RDF triples are used for extractive properties, whereas the RDF quads are used for abstractive ones as they contain information on the high-level abstraction or pattern occurrences of the properties' data. The definition of the event model is as follows: Let EV be the set of events, PID the set of publisher IDs, $PubID$ the set of publication IDs, T the set of timestamps, and $G(e)$ the RDF graph or summary of an entity e, respectively, then:

$$\begin{gathered} ev \in EV \Leftrightarrow ev = \{(pID, pubID, t, g), \ldots : \\ pID \in PID, pubID \in PubID, t \in T, g \in G(e)\} \end{gathered} \tag{3.2}$$

In Fig. 3.9, an example of an event payload that combines extractive and abstractive properties of an entity is given. According to it, the notification is an extractive and abstractive summary of the entity Usain_Bolt that contains the top-5 most diverse and important entity information. This information comes from multiple publishers generating entity data ranging from heart rate and pulse measurements to information from static sources. The summary contains: 1) RDF triples, where the subject is the entity in question, the object is the triple's original value, and the predicate is an attribute/relation between the subject and the object (e.g. <Usain_Bolt> <title> <200_metres>), and 2) RDF quads, where the object is instead an attribute's aggregated value, within a specific window, and the additional context is the pattern occurrence of an attribute's raw values, within a specific window, as well as the reasoning result by the approximate rules (e.g. <Usain_Bolt> <heartRate> <106.960> <overall neutral with one sharp decrease, TACHYCARDIA>).


Publishers
publish
Entity-centric Publish/Subscribe Summarisation System
subscribe
Subscribers
summarise
notify
Extractive & Abstractive Summary

```
<Usain_Bolt> <heartRate> <107.100>
<Usain_Bolt> <heartRate> <107.100>
<Usain_Bolt> <heartRate> <107.100>
...
<Usain_Bolt> <heartRate> <103>
<Usain_Bolt> <heartRate> <103.500>
<Usain_Bolt> <heartRate> <103>

<Usain_Bolt> <pulse> <104.200>
<Usain_Bolt> <pulse> <104.300>
<Usain_Bolt> <pulse> <107.600>
...
<Usain_Bolt> <pulse> <103.700>
<Usain_Bolt> <pulse> <103.700>
<Usain_Bolt> <pulse> <103>
...
<Usain_Bolt> <birthPlace> <Trelawny_Parish,_Jamaica>
<Usain_Bolt> <birthPlace> <Trelawny_Parish>
<Usain_Bolt> <birthPlace> <Jamaica>
...
<Usain_Bolt> <title>
<Men's_100_metres_world_record_progression>
<Usain_Bolt> <title> <200_metres>
<Usain_Bolt> <years> <2012_Summer_Olympics>
```

```
entity = "Usain_Bolt", k = 5,
windowType = "CountTumbling",
windowSize = 100, summary =
"Extractive, Abstractive"
```

```
<Usain_Bolt> <pulse> <103.364> <overall neutral with reoccurring sharp
increases/decreases followed by sharp decreases/increases and one
sharp decrease, ABNORMAL>
<Usain_Bolt> <years> <2012_Summer_Olympics>
<Usain_Bolt> <heartRate> <106.960> <overall neutral with one sharp
decrease, TACHYCARDIA>
<Usain_Bolt> <title> <200_metres>
<Usain_Bolt> <title> <Men's_100_metres_world_record_progression>
```

Figure 3.9: An example of the Entity-centric Publish/Subscribe Summarisation System.

3.4.5 Offline Evaluation Component

This component addresses the following research questions:

- **RQ1.4**: Can the satisfaction of users based on the received notifications be evaluated according to: (1) how well they address the users' different needs and interpretations *(R4)*?, (2) how much they reduce the information overload to the users *(R4)* and to the system *(R6)*?

- **RQ2.4**: Can the semantically-abstracted rich notifications be evaluated according to: (1) how well they cover the wide range of different concepts and contexts *(R5)*?, (2) how well they reduce information redundancy without sacrificing important information *(R5)*?, 3) how much they boost the system's performance *(R6)*, 4) how many dependencies are needed (e.g. domain experts, external ontologies, memory-heavy models etc.) *(R4, R5, R6)*?

The proposed Pub/Sub system needs to be evaluated, especially, since summarisation, approximation, and high-level abstraction techniques take place that transform or eliminate part of the original information. Appropriate real-world datasets, evaluation metrics, ground truths, and baselines are defined that examine effectiveness and efficiency. Specifically, the user and data effectiveness of the entity summaries are examined in terms of quality according to the users' needs, redundancy-awareness, and expressiveness based on how well the available wide range of concepts and contexts of the generated data is covered. The system efficiency of the summarisation approaches and the notifications is examined according to notification size, processing time, memory, throughput, and scalability.

The aim of the offline evaluation is to proceed with the effectiveness and efficiency of the proposed system. Specifically, as the streaming takes place, every newly constructed entity summary from the *Event Model* component is sent to a queue that along with necessary metadata are stored externally. Once the streaming has stopped, all entity summaries are examined based on a range of adopted, adapted, or newly formulated metrics that observe effectiveness and efficiency. More details are given in Chapters 4, 5, and 6.

The overall architecture of the Entity-centric Publish/Subscribe Summarisation System is illustrated in Fig. 3.10.

3.5 Summary

IoT as a trending technology has contributed to the creation of a wide range of entity-centric data coming from multiple sources and users interested in its real-time analysis and high-level states regarding applications like Healthcare and Smart Cities. Nevertheless, IoT environments possess multiple challenges regarding data (e.g. heterogeneity, redundancy, and data enrichment), users (e.g. high-level interpretation, user-friendly data representation, and non-technical users), and system (e.g. scalability, timeliness, and resource constraints). Existing Pub/Sub schemes can cope with the system challenges of IoT, but they fall short when it comes to data and user challenges.

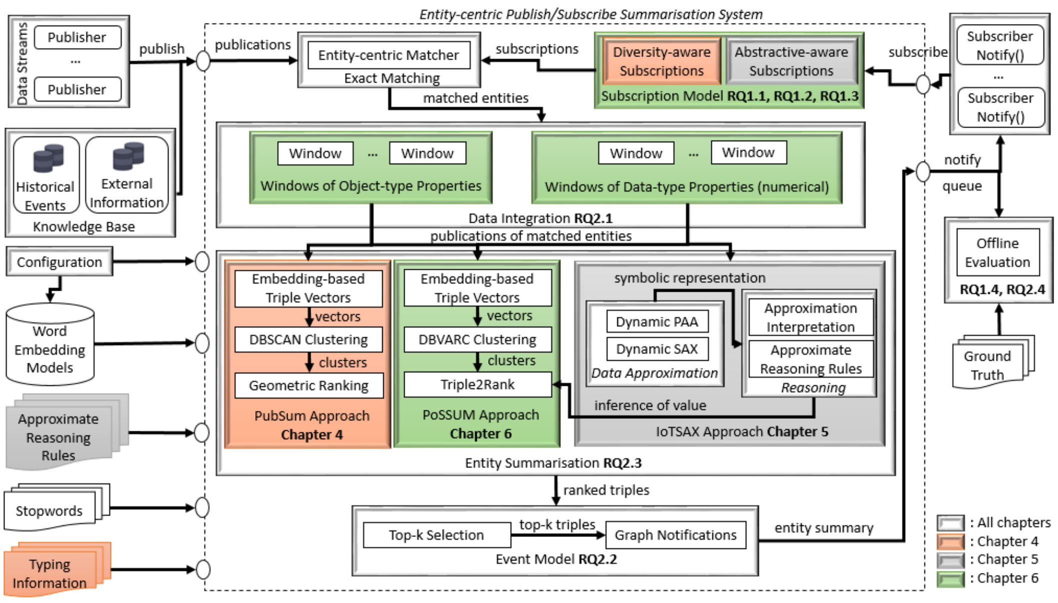


Figure 3.10: The architecture of the Entity-centric Publish/Subscribe Summarisation System.

In this chapter, a contextually-aware middleware solution is proposed and analysed; the *Entity-centric Publish/Subscribe Summarisation System*, which combines the advantages of topic-based and graph-based Pub/Sub schemes to provide entity summaries for numerical and Linked Data. The system addresses research questions related to requirements concerning usability, user expressibility, data expressiveness, user and data effectiveness, and system efficiency by incorporating multiple components. The components are split into Data Integration, Entity Summarisation, Subscription Model, Event Model, and Offline Evaluation, and explore how advanced methodologies like windowing, data fusion, high-level abstraction, approximation, summarisation by conceptual clustering and top-k ranking, representationally-decoupled subscriptions, and graph-based notifications can result in expressive entity summaries of high quality using limited resources and processing time at no expense of usability. These methodologies are evaluated by a proposed evaluation methodology concerning the construction of ground truths, formulation of evaluation metrics, and identification of baselines. The aforementioned methodologies are split into three proposed approaches; PubSum, IoTSAX, and PoSSUM, which are thoroughly analysed in Chapters 4, 5, and 6, respectively. The chapter concludes with an overall map among challenges, requirements, research questions, components, solutions, and chapters.

Chapter 4

Dynamic Diverse Entity Summarisation of Linked Data

4.1 Introduction

This chapter is dedicated to the PubSum approach, which has been published in the papers Pavlopoulou and Curry [68–70].

The PubSum approach is a novel dynamic diverse summarisation methodology with the use of a range of windowing policies, embedding-based density clustering, and a geometric-based top-k ranking. It involves heterogeneous datasets of triples with object-type properties (e.g. <Aarhus> <timeZone> <Central_European_Time>) or information coming from knowledge bases (e.g. DBpedia) for a range of entities including people, things, and places that may contain redundancy in the attributes and/or values due to duplicates or conceptual similarity. Therefore, it is partially linked to the motivational scenarios regarding Healthcare and Smart Cities, described in Chapter 2. Numerical real-time sensor data is not covered in this chapter as out of scope. The approach applies to domain-agnostic environments that demand scalability and timeliness, are of limited resources, and have no standardisation in data representations (i.e. representationally-coupled). Although its main application is extractive entity summarisation, the approach can also be used for transforming triples to semantic vectors, conceptual clustering, and triple ranking.

Several challenges analysed in Chapter 2 are addressed with solutions described in Chapter 3. An emphasis is given on overcoming data challenges regarding heterogeneity and redundancy caused by duplication and conceptual similarity. Specifically, data with different representations is observed and grouped in conceptual clusters if they are conceptually and contextually related. A ranking, then, decides which data from each cluster will be included in a user notification to provide a diverse set of information (extractive summary) that does not overwhelm the subscribers and the processing system. In the meantime, users need only provide representationally-decoupled subscriptions through the proposed subscription model. User challenges are also tackled regarding user-friendly data representation by providing graph-based notifications, and non-technical users' facilitation by demanding only diversity-aware subscriptions that do not include a-priori data, semantic or schematic information, or return partial or abstract user information. Finally, all system challenges regarding scalability, timeliness, and resource constraints are addressed since a Publish/Subscribe system (Pub/Sub) is proposed along with windowing policies and summarisation that can cover dynamic sources, data continuity, real-time processing requirements, and possible network overhead. It should be noted that data enrichment and high-level interpretation are not covered in this chapter as they are out of scope. An emphasis (in orange) on which components are analysed in this chapter from the overall proposed *Entity-centric Publish/Subscribe Summarisation System* along with their associated research questions is illustrated in Fig. 4.1.

The contributions of this chapter are the following:

- A user-friendly entity-centric subscription model that allows subscribers to express in a simple way whether they need to receive a diverse extractive entity summary with top-k diverse filtered information by also providing the desired window traits (type, size, and slide).
- PubSum, a novel dynamic diverse summarisation methodology for heterogeneous Linked Data entity graph streams that is based on: 1) an embedding-based DBSCAN clustering that provides conceptual clusters, and 2) a geometric-based top-k ranking of the clusters that is related to user query relevance, importance, and diversity to create graph-based extractive summary notifications of entities.
- A novel evaluation methodology including:
  - The identification of an evaluation dataset based on Linked Data deriving from DBpedia related to entities like people, things, and places with

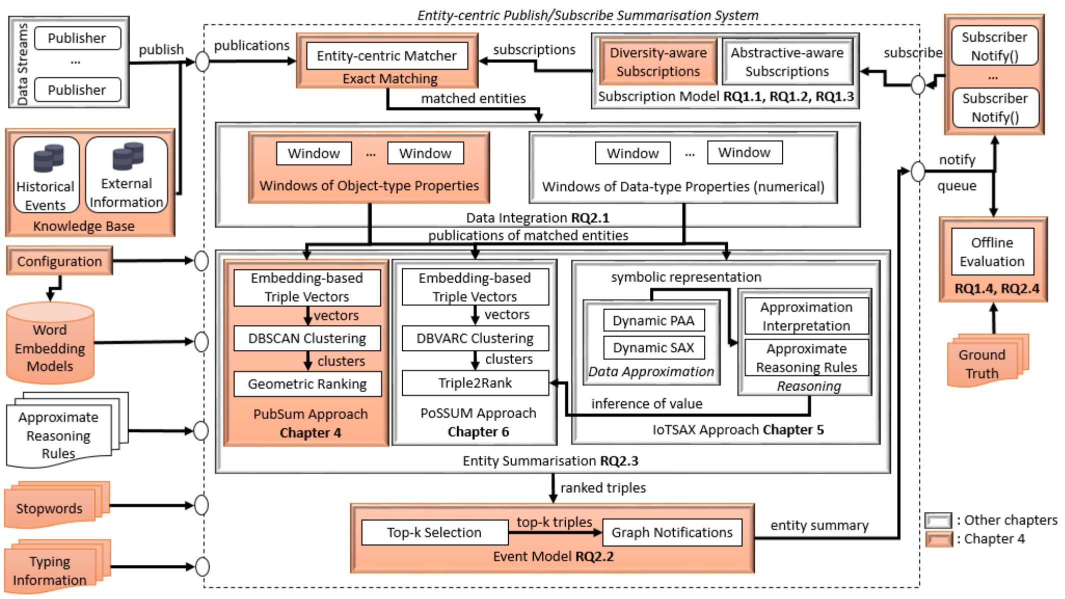


Figure 4.1: The components of the Entity-centric Publish/Subscribe Summarisation System analysed in this chapter.

characteristics involving heterogeneity in data types, semantics, concepts, and contexts.

– The identification of a user-defined ground truth that contains ideal summaries from human judges.

– Identification of FACES [50], a static diverse entity summarisation methodology, and typical Pub/Sub as baselines, and adaptation of FACES to the proposed dynamic diverse Pub/Sub summarisation system.

– Identification of a range of evaluation metrics related to the agreement, quality, redundancy-aware F-score, latency, size reduction, memory footprint, and throughput.

• An extensive evaluation comparison between PubSum approach and baselines by examining user effectiveness, data effectiveness, and system efficiency by using a range of windowing policies. The results include:

– User effectiveness: promising but worse summary quality.

– Data effectiveness: redundancy-aware F-score of up to 0.95 and up to 69.3% duplication reduction.

– Efficiency: 6 times less latency ranging from 29,237ms to 187,395ms and 3 times less memory compared to FACES, up to 92% message reduction, and throughput ranging from 833 to 1,005 events/second.

The rest of the chapter is as follows: In Section 4.2 the necessary background is given related to windowing policies, algorithms, distance measures, and linguistic methodologies, whereas Section 4.3 contains related work and how it maps to the challenges, requirements, and research questions of the thesis. Section 4.4 describes the proposed Extractive Publish/Subscribe Summarisation System, where its architecture and details of the PubSum approach are provided. Section 4.5 analyses the evaluation methodology as well as the results of the experiments, while Section 4.6 concludes and summarises the chapter.

4.2 Background

This section contains the necessary background for the rest of the chapter. Windowing policies, algorithms, distance measures, and linguistic methodologies are explained that are used in this chapter as well as the following ones.

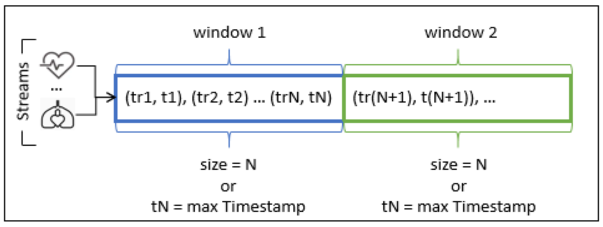
4.2.1 Windowing Policies

This chapter's work as well as the following chapters refer to windowing policies. Windows are structures that are used in stream processing for efficiency. They can deal with unbounded streams of information by splitting the data into buckets of specific sizes, over which further processing occurs. There are multiple window types that have been proposed in the literature; nevertheless, this thesis' work emphasises on the following ones:

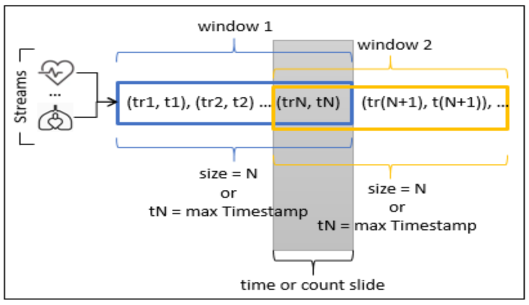
Count Tumbling Window: This window contains data up to a specific user-defined maximum count/size. Then, a new window of the same size is generated with newly arrived data.

Count Sliding Window: This window contains data up to a specific user-defined maximum count/size. Then, a new window of the same size is generated by sliding up to a user-defined slide. This means that the new window contains the last data of the previous window equal to the specified slide in number along with newly arrived data.

Time Tumbling Window: This window contains data generated before a specific user-defined maximum timestamp. Then, a new window of the same size



(a) Count tumbling or time tumbling windows



(b) Count sliding or time sliding windows

Figure 4.2: Different window types and their policies.

is generated with newly arrived data.

Time Sliding Window: This window contains data generated before a specific user-defined maximum timestamp. Then, a new window of the same size is generated (for the next timestamp period) by sliding up to a user-defined time slide. This means that the new window contains the last data of the previous window that was generated during the time span of the slide along with newly arrived data that is generated before the new maximum timestamp (next timestamp period).

The aforementioned window types and their policies are illustrated in Fig. 4.2, where different streams generate a collection of triples (tr_i) that are timestamped (t_i).

The window lifecycle of the aforementioned window types describes all stages of their lives. In this thesis' work, the lifecycle is inspired by Flink [140], but customised to satisfy the system requirements. The window lifecycle stages are the following:

Window Creator: A window is created based on a pre-condition. For example, in this thesis' work, that would involve a matched entity between the generated ones by the publishers and the requested ones by the subscribers.

Window Assigner: The window is populated with data coming from multiple streams. For example, in this thesis' work, windows would be filled with triples coming from multiple publishers that are related to the specific matched entity the window is linked to.

Window Processor: The data within the window is fused and processed based on the approach selected.

Trigger: The trigger specifies the conditions under which the window's elements are considered ready to be processed. The trigger could be either on a per-element basis that enters the window (incremental processing) or on a full capacity basis (batch processing).

Evictor: The evictor is responsible for removing elements from the window after the processing has been done. Depending on the window type, the evictor will either delete all elements from the window or a subset.

4.2.2 Density-based Spatial Clustering of Applications with Noise

In this chapter's work, the Density-based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [141] is used. Clustering is an unsupervised learning methodology that groups data into separate clusters based on similarity, resulting in clusters with elements that are similar to their cluster's elements and dissimilar to the other clusters' elements. In the case of DBSCAN, this is translated to finding the densely-connected regions, if elements were depicted in a semantic space. This could mean that these regions could form conceptual clusters, where the elements of each cluster are conceptually similar to one another.

In order for DBSCAN to find densely-connected regions, it uses two parameters; $minPts$, which is the minimum number of points that can form a cluster, and ε, which is the distance threshold that locates the points that are neighbours of any point. The points are separated into three categories; *core point*, which is the one that has at least $minPts$ number of points nearby within radius ε, *border point*, which is the one that belongs to the neighbourhood of a nearby core point, and *outlier*, which is the one that is not reached by any points. An example is given in Fig. 4.3. The algorithm starts with a random point and observes if there are at least 3 nearby points within its ε radius. If this is the case, then, the point is marked as a core point and along with its border points they form a cluster, otherwise, the point is considered an outlier. The cluster is expanded, accordingly, based on whether the border points are core points themselves. The algorithm stops when all points have been visited.

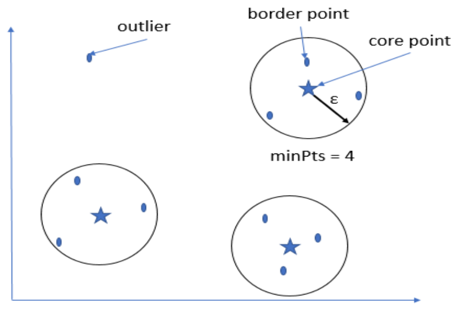


Figure 4.3: An example of DBSCAN clustering.

In general, density-based algorithms are more suitable for evolving data streams compared to other kinds of clustering algorithms (e.g. K-means) due to their following characteristics:

- *No pre-defined number of clusters parameter*: Knowing a-priori the correct value of this parameter is not possible, especially, in streaming and evolving data that could lead to a change of the number of clusters.
- *No restrictions on the size and the shape of the clusters.* In streaming data, an area with similar or related characteristics in a vector space could be of any shape and population.

One of the biggest limitations of DBSCAN is that it applies to static data and that the correct selection of the parameter values can seriously affect the quality of the formed clusters.

4.2.3 Distance Measures

Distance measures explore the distance between elements in a space. They can be used for different purposes. For example, DBSCAN uses distance measures to define the distance between points to form clusters, based on whether the distance is less than or equal to the ε threshold. The two highly-used metrics that are referred to in this thesis are Euclidean distance and cosine similarity. Assuming one wants to find the distance or similarity between two elements $X = \langle x_1, x_2, \ldots, x_N \rangle \in \mathbb{R}^N$ and $Y = \langle y_1, y_2, \ldots, y_N \rangle \in \mathbb{R}^N$, then, the metrics are defined as:

$$Euclidean\ distance = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2} \qquad 4.1$$

$$cosine\ similarity = \frac{\sum_{i=1}^{N} x_i y_i}{\sqrt{\sum_{i=1}^{N} x_i^2}\sqrt{\sum_{i=1}^{N} y_i^2}} \qquad 4.2$$

The two metrics have opposite interpretations. For example, in the case the metric is used to find conceptual similarity of words (elements) in a vector space, then, the lower the Euclidean distance between two words the better, whereas the higher the cosine similarity between two words the better. These metrics are used in this chapter and the following ones.

4.2.4 Linguistic Pre-processing and Importance

Often words (or text, in general) cannot be used directly in the way they are depicted within data for the purposes of analytics. This is especially evident in Resource Description Framework (RDF) triples that may contain words that do not even make sense the way they are described with complex prefixes and long concatenated words that may represent important user information, but no actual words [125]. A range of pre-processing techniques (used in this chapter and the following ones) should take place to ensure the extraction of more useful information from the original words, analysed below:

Tokenisation: splits a composite word into its individual words, called *tokens* (e.g. outsideLuminance has two tokens; outside and luminance).

Lower-casing: transforms a word into lowercase characters (e.g. outsideLuminance to outsideluminance).

Stop-word removal: removes words like "a", "the" etc. that are deemed semantically insignificant (e.g. outsideLuminance to luminance).

Lemmatisation: transforms a word into its lemma, that is its base or dictionary form (e.g. luminance and luminances to luminance).

Part of Speech (POS) tagging: links a word to its part of speech based on its definition and context (e.g. luminance to luminance[NN], which means noun).

Many approaches, after pre-processing words, use different types of importance metrics that symbolise the impact of the word in a corpus or collection of documents. The most popular one is term frequency–inverse document frequency (tf-idf) that FACES, this chapter's work baseline, and a proposed methodology, described in Chapter 6, are based on . It shows the trade-off between the number of occurrences of a word in a document and the number of documents in the corpus that contain the word. This means that the more a word is used in a document, the more representative it is for this document. However, the more the term is used in a collection of documents, the less discriminative it is. The metric is defined as:

$$tf{-}idf = freqOfWordInDocument * \log \frac{numOfDocuments}{numOfDocumentsThatContainWord} \tag{4.3}$$

4.3 Related Work

This section contains the related work, which is split into the following categories:

- **Diverse Entity Summarisation**: It involves summarisation approaches of static RDF graphs based on user importance, diversity, relevance, and popularity.
- **Diversity in Pub/Sub**: It involves Pub/Sub systems that produce top-k diverse notifications.
- **Approximate Semantic Matching in Pub/Sub**: It involves Pub/Sub systems that introduce semantic decoupling in the system.
- **Semantic Engines in Pub/Sub**: It involves Pub/Sub systems that semantically enrich the events.
- **Other Approaches**: It includes graph summarisation, graph approximation, subscription summarisation in Pub/Sub, and triple ranking.

The related work is analysed and mapped according to the aforementioned requirements:

- **R1: Usability**: high usability system independent of representational coupling, query language expertise, system knowledge, bias, and background knowledge.
- **R2: User Expressibility**: users understanding their data needs and expressing them by simple subscriptions of high usability with minimal configuration settings.

- **R3: Data Expressiveness**: domain-agnostic system tackling interoperability and heterogeneity by providing rich notifications that contain conceptual and contextual diversity or high-level abstractions.
- **R4: User Effectiveness**: notifications of high quality according to the users' needs.
- **R5: Data Effectiveness**: redundancy-aware and expressive notifications of high quality according to the wide range of concepts and contexts.
- **R6: Efficiency**: efficient system in terms of memory, processing time, throughput, and scalability.

4.3.1 Diverse Entity Summarisation

Diverse entity summarisation involves the selection of a diverse and important subset of the whole entity information in order to boost the system performance and to avoid overwhelming the users due to the plethora of data [41, 53, 114]. Only offline methodologies are involved in this category [53]. The most notable works are analysed below with each representing a work that focuses on thesauri/ontologies, one that focuses on topic modelling, and one that focuses on Deep Learning.

FACES [50]

Description: This work creates entity summaries based on diversity, uniqueness, and popularity. Initially, the approach uses the WordNet[1] thesaurus to enrich the original triples with related terms followed by a modified Cobweb algorithm [142] that clusters the triples of each entity based on conceptual similarity. Then, an adapted tf-idf metric ranks the triples within each cluster so that the most popular and informative ones within the DBpedia[2] ontology are chosen first for the summary. At least one triple from each cluster is included in the top-k summary depending on the value of k and some pre-defined selection rules.

Review: The work addresses challenges related to heterogeneity and redundancy caused by duplication and conceptual similarity (R3) as data with different representations is grouped in conceptual clusters and a ranking decides the final

[1]`https://wordnet.princeton.edu/`
[2]`https://www.dbpedia.org/`

diverse set of information within the summary. At the same time, the user is obliged to perform only keyword-based queries (i.e. entity name), making the approach highly usable and able to facilitate non-technical users (R1). Finally, the final summary is presented as an RDF graph, which is a user-friendly data representation that contains conceptually and contextually rich information (R3).

On the other hand, the effectiveness (R4, R5) of the approach relies on a combination of thesauri and ontologies, which are linked to aforementioned limitations, described in Chapter 2, regarding complexity and inefficiency, representational coupling, domain dependency and domain-experts, lower semantic expressiveness in thesauri, and correctness/quality. Also, the effectiveness of the solution is only evaluated from the perspective of the users (R4) and not from the data (R5). Specifically, the effectiveness is only based on the ideal summaries of a collection of entities in DBpedia provided by a number of human judges, which could be a highly subjective process and does not represent the level of the redundancy-awareness or the range of concepts and contexts that the summaries contain. Finally, the approach applies to static data; therefore, it is not efficient for Internet of Things (IoT) environments (R6).

ES-LDAext [51]

Description: This work creates entity summaries by first enriching triples with the use of a pre-trained Word2Vec embedding model (specifically, Freebase) and extracting entities from data-type properties with Named Entity Recognition [143]. Then, topic modelling takes place by modifying the Latent Dirichlet Allocation (LDA) [144] methodology, where each entity is considered as a document with a multinomial distribution over its predicates. Each predicate is itself a probability distribution over the subjects and objects. Finally, the triples are ranked based on their probability distributions in the DBpedia ontology and the top-k entity information is selected as a summary.

Review: The work contains the advantages aforementioned in FACES (R1, R3, R4). It also relies on word embedding models that are more flexible and superior than thesauri and ontologies, on which FACES is based (Chapter 2). Nevertheless, the Freebase embedding model is used that is memory heavy and could affect the efficiency of the approach (R6). Also, LDA is used for finding conceptually similar triples that relate to specific topics. LDA is a supervised methodology, that is it needs training data in order to be effective, and it requires manually-tuned hyperparameters (R4, R5). Supervised solutions are not efficient in streaming environments as a prior collection of data to train a model and frequent updates of the training set according to changes (i.e. concept drift) will prove time-consuming (R6). There also seems to be a bias of how important an object is, to be included in a summary, based on the number of related words in the embedding model (R4, R5). Finally, limitations regarding the effectiveness evaluation and inefficiency for IoT environments of FACES, also apply to this approach (R4-R6).

DeepLENS [52]

Description: This work is one of the two approaches (the other is ESA [145]) that use Deep Learning to create entity summaries. Initially, triples are transformed into vectors with the use of a pre-trained FastText [146] word embedding model. Then, the vectors are fed into a multilayer perceptron model and the final scoring for each triple is extracted.

Review: The work contains the advantages aforementioned in FACES and ES-LDAext (R1, R3, R4). Nevertheless, the approach relies on a multi-layer perceptron model, which is a Deep Learning supervised methodology that apart from needing labelled entity summaries and domain experts, needs a number of manually-tuned parameters for optimised results (R4, R5). This hinders the approach's adaptability to streaming or IoT environments due to a lack of efficiency (R6). Also, the approach only applies to static data (R6). Finally, the approach presents the same limitation of FACES regarding the effectiveness evaluation (R4, R5).

Other Relevant Approaches

Other Diverse Entity Summarisation Approaches Some other approaches have addressed diverse entity summarisation (R1, R3, R4) for different purposes that may be unrelated to this thesis' focus. All of these approaches contain the aforementioned limitations and some are poorly evaluated or not evaluated at all (R4-R6). Some works include DIVERSUM [114, 147], FACES-E [148], CD [149], MPSUM [150], REMES [151], and ESA [145]. DIVERSUM focuses on a per-predicate basis summarisation based on novelty, importance, popularity, and diversity by adapting the document-based Information Retrieval to the knowledge graphs. FACES-E improves on FACES by considering data-type properties (strings containing entity references) instead of only object-type properties. CD refers to summaries based on feature information overlap by calculating string/numerical similarities and ontological dependencies as well as feature selection by formulating a Quadratic Knapsack Problem [152]. MPSUM creates summaries with the use of LDA topic modelling and ranking based on the uniqueness of predicates and importance of objects. REMES focuses on multi-entity summarisation based on diversity, relatedness, and importance by using an adapted Quadratic Knapsack Problem to find relatedness between features and multiple entities. ESA is the other approach that uses Deep Learning to create entity summaries by transforming triples into vectors deriving from TransE [153], a graph embedding model (not a word embedding one like this thesis' work or DeepLENS), and feeding them to a Bidirectional LSTM model with a supervised attention mechanism to extract scores of triples.

Relevance-based Entity Summarisation Several approaches have addressed relevance-based entity summarisation (R1, R3, R4). This type of summarisation is not directly linked to the proposed work since it is based on user queries with conditional rules and focuses on the structure of the graphs, unlike this thesis' work that refers only to star-like graphs. Nevertheless, some notable works are RELIN [154], SUMMARUM [155], LinkSUM [156], and DynES [157]. RELIN is the first work to address the entity summarisation need, and it emphasises on both object-type and data-type properties. The summary is based on relatedness and informativeness via a generalisation of PageRank [158] and its centrality-based ranking. SUMMARUM emphasises on summaries based on popularity via PageRank scoring and it is implemented as a Web service. LinkSUM improves on SUMMARUM as the summaries are based on relevance, frequency, exclusivity, and predicate description via PageRank and Backlink [159]. DynES [157] creates summaries of entity cards based on importance and relevance through feature frequency, informativeness, and specificity as well as information retrieval inspired methodologies that are evaluated through crowdsourcing.

4.3.2 Diversity in Pub/Sub

Few approaches have proposed Pub/Sub systems that provide expressive and diverse notifications that do not exactly cover all of the subscription constraints, but they target a high percentage of user preferences. The approaches are analysed below.

PrefSIENA [54]

Description: This work proposes a content-based Pub/Sub scheme that delivers top-k ranked events based on preferential subscriptions. Subscribers use either scores (quantitative approach) or binary relations (qualitative approach) in attribute-value pairs to signify the amount of preference in specific events compared to others within the same subscription. Preferential subscription graphs are, then, created that show the preference relationships (edges) among subscriptions (nodes) of a subscriber. The event matching is based on these graphs for efficiency. Finally, exact string similarity is used among the top-k notifications for increasing their content diversity.

Review: The work addresses challenges related to redundancy caused by duplication (R3), scalability, and timeliness and resource constraints as a Pub/Sub system is extended (R6) that provides the subscribers with the highest ranked (non-common) top-k events based on their preference (R2). Also, efficient preferential-based matching is proposed and heuristics are used for diversity (R6). Finally, a thorough evaluation is performed that observes the user (R4) and data (R5) effectiveness as well as the efficiency (R6) by examining a range of windowing policies (i.e. periodic, sliding, and history-based).

On the other hand, even though the subscriptions have higher expressibility (R2) due to the ability of preferences in attribute-value pairs, they are still representationally-coupled. This makes the approach of medium usability to non-technical users (R1). Also, the approach addresses redundancy only on the basis of duplication or common attributes, without considering conceptual and contextual similarity (R3). This is also suggested by the fact that events are only attribute-value pairs and not rich entity-based graphs. Another issue is that the work assumes that the events have the same number of attributes. Also, although the work could be domain-agnostic, only a limited number of attributes is examined (R3). Finally, the construction of expressive notifications via summarisation is out of scope (R3), but a top-k ranking is performed so that the users and the system are not overwhelmed (R4-R6).

Chen et al. [55]

Description: This work proposes a topic-based Pub/Sub scheme that delivers top-k events based on relevance, recency, and diversity. Subscribers use keyword-based queries to ask about specific topics, and publications (documents in this case) are received as notifications. Two filtering methods are used; individual and group filtering, which define the filtered documents that cater for individual subscriptions and the ones that are shared by different subscriptions to reduce computation, respectively. Finally, notification diversity is calculated based on the pair-wise similarity (cosine similarity) between documents.

Review: The work contains the advantages aforementioned in PrefSIENA regarding redundancy and system performance (R3, R6). Also, non-technical users are facilitated since the subscriptions are only topic-based making the approach highly usable (R1). Indices are also used for efficiency (R6). On the other hand, this work holds the same disadvantages as the topic-based Pub/Sub regarding user expressibility (R2) and irrelevance in notifications (R4-R6), since relevance is only examined based on the subscriptions' topics that could refer to ambiguous terms. Also, the effectiveness of the solution is only evaluated from the perspective of the users (R4) and not from the data (R5). Specifically, the effectiveness is only based on the satisfaction of 3 judges, which is a low number and highly subjective, and does not represent the level of redundancy-awareness. Another issue is that the efficiency (R6) of the approach is highly dependent on the computational complexity and memory of the lists, indices, and term weight aggregations constructed. Finally, the limitations of PrefSIENA regarding redundancy only on the basis of duplication or common attributes, lack of rich entity-based graphs, and summarisation being out of scope, also apply to this work (R3).

Hmedeh et al. [56]

Description: This work is based on novel and diverse items in Web syndication. Subscribers use keyword-based queries to ask about specific topics, and publications (texts) are received as notifications based on novelty and diversity compared to the history of notifications for each subscription. The novelty is based on term discrimination value and a threshold, whereas diversity is based on pair-wise similarity (Euclidean distance). Old items are removed when a new item arrives and shared filtering is taking place for related subscriptions for efficiency.

Review: The work contains the advantages (R1, R3, R6) aforementioned in Chen et al. [55]. Also, an efficient methodology for shared-history filtering of subscriptions is proposed (R6) and an evaluation is taking place for both user

(R4) and data (R5) effectiveness. On the other hand, the data effectiveness of the solution is poorly evaluated as only the filtering rate was examined (R5). Also, this work holds the same disadvantages of the topic-based Pub/Sub regarding user expressibility (R2). Another issue is that the approach is memory-heavy since a history of subscriptions is kept to calculate novelty and diversity in their notifications (R6). Also, the novelty threshold is fixed and depends on the output rate. This fixed value could affect the effectiveness of the approach (R4, R5). Finally, the limitations of PrefSIENA regarding redundancy only on the basis of duplication or common attributes, lack of rich entity-based graphs, and summarisation being out of scope, also apply for this work (R3).

Other Relevant Approaches

Few approaches have addressed event matching based on preferential subscriptions, that is they provide expressive notifications in terms of covering a high percentage of the user preferences (R2). However, these methodologies do not address redundancy of any form within the notifications, the data is attribute-value pairs and not rich entity-based graphs, and summarisation is out of scope (R3). Also, although these approaches provide the users with higher and, at the same time, simpler expressibility (R2), the subscriptions are still in the form of representationally-coupled attribute-value pairs, making the approaches of medium usability to non-technical users (R1). Some notable works are BE*-tree [160] and FX-TM [161]. BE*-tree proposes subscriptions that contain weight-based constraints based on preference and an effective as well as efficient tree-based matcher for hierarchical top-k matching of data and subscriptions with high dimensionality. FX-TM proposes weight-based subscriptions (i.e interval attributes, subscription or event weights, positive or negative weights, wildcards or missing attributes, and dynamic alteration of scores) as well as a matcher that provides the k-highest scored notifications through different indices and structures for each attribute.

4.3.3 Approximate Semantic Matching in Pub/Sub

As aforementioned, most Pub/Sub systems do not support approximate/relaxed queries that can address representational coupling issues between subscriptions and events. The most notable works, addressing this issue from the perspective of semantic decoupling, are analysed below.

Hasan et al. [57]

Description: This work proposes an approximate semantic event processing model for events coming from heterogeneous sources. Initially, the relatedness between the values of subscriptions and the ones of events are examined by distributional semantics (specifically, Wikipedia ESA). Then, the relatedness between the properties is examined by the WordNet thesaurus. Finally, all mappings of the approximate semantic matcher that are higher than a cut-off threshold are considered as matches.

Review: The work addresses challenges related to heterogeneity, user-friendly data representation, scalability, and timeliness and resource constraints as a Pub/Sub system is extended (R6) that offers semantic relatedness exploration of data and provides RDF graphs as notifications. Also, the approach relies on WordNet thesaurus and Wikipedia ESA, which is a semantic model that turns words into vectors based on text corpora in Wikipedia. This distributional semantic model does not contain the aforementioned limitations of ontologies; however, its quality is inferior to advanced Word2Vec embedding models.

On the other hand, the users are still obliged to create queries in the form of triples (R2), which in this case is more linked to attribute-value pairs queries than SPARQL-like ones. This makes the approach not highly usable to non-technical users, even with the added feature of semantic relatedness (R1). Also, the effectiveness (R4) of the approach is linked to the cut-off threshold that is unclear whether it should be defined by the system and apply to all subscriptions in the same manner or should be defined by the users. Finally, data redundancy is not addressed as conceptually-similar notifications may occur, and the construction of expressive notifications via summarisation is out of scope (R3).

Hasan et al. [39]

Description: This work proposes a content-based Pub/Sub scheme with an approximate semantic matcher, in which similarity matrices are constructed via evolving Pareto frontier for top-1, top-k matchers, and handling uncertainty. Wikipedia ESA is used to score the relatedness among words in events of attribute-value pairs and a relaxed subscription model is proposed.

Review: The work contains the advantages (R6) aforementioned in Hasan et al. [57] with the exception of user-friendly data representation as notifications are represented only in the form of attribute-value pairs (no RDF graphs are involved in this work). Also, the users have higher expressibility (R2) in this work as they can define which subscription terms will be approximated and a simpler attribute-value pairs format is used compared to the triple-based one in Hasan et al. [57]. Finally, an interesting evaluation methodology (R4, R6) is proposed and optimisations regarding the order of properties and commonalities among subscriptions are implemented for higher efficiency (R6).

On the other hand, the work also contains the limitations (R1, R3) aforementioned in Hasan et al. [57]. Finally, the similarity matrices may also be too many depending on the properties and values in question, and their size may be too big depending on the number of elements in each event (R6).

Alhakbani et al. [58]

Description: This work proposes a content-based Pub/Sub scheme with an approximate semantic matcher. Initially, related events and subscriptions are clustered together via taxonomy clustering based on assigned topics. Then, tree structures are used to match events to subscriptions that fall within intersected clusters with the use of Wikipedia ESA. Finally, a relaxed subscription model is proposed.

Review: The work contains the advantages (R6) aforementioned in Hasan et al. [39] as it is highly affected by the latter. The same evaluation methodology (R4, R6) is also used and optimisations are suggested regarding higher efficiency (R6). However, the same limitations (R1, R3) apply, as well, and the methodology is based on taxonomies with associated limitations that can highly affect the effectiveness (R4) of the approach.

Other Relevant Approaches

Other Approximate Semantic Matching in Pub/Sub Approaches Other approaches have addressed approximate semantic matching that contain the aforementioned limitations (R1, R3) with some being poorly evaluated or not evaluated at all (R4, R6). Some works include A-TOPSS [162], S-ToPSS [163], and FOMatch [164]. A-TOPSS proposes a subscription model that allows the expression of vagueness, and the use of fuzzy set theory and probability theory so that events of attribute-value pairs match subscriptions with a degree of confidence. S-ToPSS uses synonyms, taxonomies, and mapping functions specified by domain experts for creating an approximate matcher for attribute-value pairs. FOMatch is a work that combines approximate semantic matching and preferential subscriptions with weight-based and threshold-based constraints. It uses synonyms and generalisations from thesauri and ontologies to map related words between the events and the subscriptions. Then, similarity/relatedness metrics are calculated of words and values. A match is taking place if the sum of the products of the metrics and the weight of each predicate is lower than the cut-off, which is calculated based on the sum of all the products of the threshold and the weight of each predicate within a subscription. This work has the added disadvantage of relying on thesauri and ontologies, which are linked to aforementioned limitations.

Approximate Semantic Matching in Graphs Qin et al. [96] have addressed approximate semantic matching in RDF linked data. Initially, RDF data is pre-processed and, then, the GOOGLENEWS Word2Vec embedding model is used to transform the data into vectors in the semantic space. The same process occurs for the queries, which are in the form of triples. A k-nn model (query index) is pre-trained based on the cosine similarity among the query vectors, where each class in the model represents a similar query set with its first query as the representative. Finally, the matching occurs between new triples and the representative query of each class. Although this approach addresses challenges related to heterogeneity and user-friendly data representation as well as relies on superior embedding models, it contains the same limitations aforementioned in Hasan et al. [57] regarding usability, facilitation to non-technical users (R1), data redundancy, and summarisation (R3). Also, it applies to static data and requires supervised learning of the k-nn model based on the query set; therefore, it is not efficient for IoT environments (R6). Finally, a cut-off threshold needs to be pre-determined that affects the effectiveness of the cosine similarity (R4).

4.3.4 Semantic Engines in Pub/Sub

Several approaches have proposed semantic engines in Pub/Sub systems. These approaches involve the use of ontologies with their aforementioned limitations in order to enrich the original events with related semantics or spatial information to address interoperability issues. The most notable works are analysed below.

G-TOPSS [59]

Description: This work is a graph-based Pub/Sub that uses ontologies to enrich data in the form of RDF graphs. The ontologies contain synonyms, taxonomies, and transformation rules for interoperability (e.g. descendant, ancestor, or direct instance of a class).

Review: The work addresses challenges related to heterogeneity, data enrichment, user-friendly data representation, scalability, and timeliness and resource constraints as a Pub/Sub system is extended (R6) that enriches the original event information with related conceptually and contextually one and provides RDF graphs as notifications. Subscription subsumption is also supported. On the other hand, the approach is based on ontologies and is a graph-based Pub/Sub that are both associated with aforementioned limitations. Specifically, subscriptions are specific constraints on edges and nodes of graphs or is-a constraint operators for RDFS taxonomy filtering like descendant, ancestor, or direct instance of a class (R2). This means that even though the subscriptions are not SPARQL-like, they are still representationally-coupled not only based on the graphs, but the taxonomies as well making the approach not highly usable to non-technical users (R1). Another issue is that the effectiveness of the notifications regarding the subscription criteria is not evaluated (R4). Finally, data redundancy and the construction of expressive notifications via summarisation is out of scope (R3).

Esposito et al. [60]

Description: This work is a topic-based Pub/Sub that dynamically builds an RDF ontology based on the incoming heterogeneous events to deal with interoperability. The subscribers and the events are related to a topic, and underlying deserialisation rules find the correct schema. The mapping among the different schemata is achieved by similarity or semantic relations in the form of equivalence, less general, more general, and disjointness. Finally, a reasoner is used that, through SPARQL queries, extracts semantic inferences of the events.

Review: The work contains the advantages aforementioned in G-TOPSS (R6). Also, non-technical users are facilitated since the subscriptions are only topic-based making the approach highly usable (R1, R2). On the other hand, the approach builds an ontology, dynamically, based on the number of events and their attributes. This suggests that the ontology could become quite complex and big (i.e. many hierarchies and classes), making the matching highly inefficient and memory-heavy when stored in a knowledge repository (R6). Also, maintaining and updating this computationally expensive ontology is challenging. Another issue is that if de-serialisation rules are not available, no match is returned. Also, the reasoner used is based on domain experts and domain ontologies with their aforementioned limitations as well as complex SPARQL queries that are representationally-coupled depending on the application in question. Another issue is that the effectiveness of the notifications regarding the subscription criteria is not evaluated (R4). Finally, data redundancy and the construction of expressive notifications via summarisation is out of scope (R3).

Other Relevant Approaches

Other Semantic Engines in Pub/Sub Approaches Other approaches have addressed interoperability in Pub/Sub with the use of ontologies. These approaches contain the aforementioned limitations with some being poorly evaluated or not evaluated at all (R4, R6). Some notable works include Wang et al. [165], Zeng et al. [166], and SCEPter [167]. Wang et al. [165] create an ontology-based Pub/Sub, where the matching is based on the structure (e.g. class and property hierarchy) and semantics of multi-hop RDF graphs. The events are in form of RDF graphs and the subscriptions are representationally-coupled RDF pattern-like. Zeng et al. [166] present a semantic Pub/Sub, where publishers and subscribers can use self-defined schemata in events and subscriptions, respectively. Mapping among the different schemata is accomplished by ontologies and relational operators that re-write the SPARQL-like subscriptions. SCEPter is a Complex Event Processing (CEP) system that uses ontologies that capture semantically related event attributes including their sub-classes and it links domain information to events regarding physical spaces or sources. The events are in the form of attribute-values pairs, and the subscriptions are complex SPARQL queries.

Fusion in Pub/Sub This work does not perceive events deriving from multiple sensors in isolation but fuses them along with complementary background knowledge (e.g. ontologies) for the purpose of information completeness. This type of work is only linked to the proposed work from the perspective of fusion among related events (entity-centric in this chapter's case). However, the approaches rely on ontologies or enrichments sources with the aforementioned limitations and assume that the producers will find the appropriate information sources and resolve representational coupling issues. Also, they are of low usability and they do not facilitate non-technical users as subscribers are obliged to perform complex SPARQL-like queries (R1). Another issue is that the efficiency (R6) of accessing and dereferencing external sources is questionable, especially, since the approaches are either poorly evaluated or not evaluated at all (R4, R5). Finally, redundancy of any form and summarisation is out of scope (R3). Some works include Wun et al. [168], Teymourian et al. [169], and Hasan et al. [170]. Wun et al. [168] fuse attribute-value pairs that result in semantic interpretations with the use of ontologies. Teymourian et al. [169] fuse events in CEP with external knowledge bases to create enriched complex events that match user queries. The work explores different event query rules, like SPARQL queries, from single or multiple events to ones with Boolean or sequential operations etc. Also, different ways of processing the query rules are addressed ranging from offline to real-time processing. Hasan et al. [170] enrich events with background knowledge (i.e. Wikipedia text, relational databases, and Linked Data) for the purpose of information completeness. The methodology proposed is native to the Pub/Sub engine. The events are in the form of RDF graphs and the subscriptions are representationally-coupled SPARQL-like queries with enrichment clauses.

4.3.5 Other Approaches

Graph Summarisation

This type of summary should not be confused with entity summarisation as it mostly emphasises on the structural summary of multi-hop graphs. The graph summarisation approaches are split into static and dynamic ones [138]. In the static case, plain graph summarisation examines only the graph's structure, whereas labelled graph summarisation examines the graph's labels, as well. In the dynamic case, plain graph summarisation examines the temporal structure. Currently, there is no dynamic labelled graph summarisation. Dynamic plain graph summarisation has been used in CEP for events related to videos by Yadav et al. [171, 172], where duplicate nodes and edges are aggregated based on the graph's spatio-temporal properties (R3).

Graph Approximation

This type of work addresses the problem of approximation in graphs (including RDF static [46] or streaming [47] data). Methodologies include sampling, sketches, histograms, and lossless/lossy compression [173]. This type of work is not directly linked to the proposed work since although it emphasises on succinct data representations of the original information for resource-constraint environments (R6), it does not refer to diverse entity summarisation (R3).

Subscription Summarisation in Pub/Sub

This type of work is not directly linked to the proposed work since it examines optimisations only on the subscription side (R4, R6). Approaches include terminologies like *subscription subsumption*, *subscription covering*, *subscription approximation*, *subscription merging*, *subscription pruning*, and *subscription summarisation* (not to be confused with entity summarisation). Notable approaches include Triantafillou et al. [174], in which an attribute-value constraint of a subscription is subsumed by that of another subscription if its values or string are the same or if they are contained in the values or string of the latter subscription, leading to the construction of summary structures, Yoneki et al. [175], in which subscriptions are approximated with Bloom filters and K-means clustering occurs on subscription nodes for sending the events in a group of subscriptions rather than individual subscriptions (load balancing), Jerzak et al. [176], in which attribute constraints in subscriptions or whole subscriptions are encoded by Bloom filters for more efficient matching, and Wang et al. [177], in which subscription partitioning occurs via random, R-tree, and K-means clustering techniques, and summary-based routing via R-trees among a set of servers to address high system throughput.

Triple Ranking

This work addresses the challenge of finding the highest ranked triples in a vast collection of heterogeneous Web data. The ranking function could be related to relevance, informativeness, diversity, or frequency. The methodologies used for ranking could take into account user feedback, interlinking, search engines, or training sets (R4). This type of work is not directly linked to the proposed work since it does not refer to or apply to streaming data (R6) and the ranking is based on the structure of multi-hop graphs rather than the conceptual and contextual diversity of the triples of star-like graphs (R3). Also, these approaches would be difficult to be applied in streaming environments as either the whole knowledge base is used for ranking or user feedback is given or supervised machine learning approaches are used that could prove highly-inefficient (R6). Notable approaches include SemRank [178] and TripleRank [179]. SemRank is a relevance-based approach that ranks semantic associations of heterogeneous knowledge bases. It is a methodology that does not return the same order of results for each search but considers the scope of the search (e.g. investigative or discovery search will return less predictable results to the users, whereas conventional search will return more predictable results to the users). TripleRank ranks triples based on tensors and link analysis. It represents semantic graphs by 3-dimensional tensors, an adjacency matrix is constructed based on links, and tensor decomposition is applied to it to rate groupings of RDF entities and predicates with respect to their topic, authority, and navigational characteristics. Other PageRank-based algorithms can be found at Roa-Valverde et al. [180].

4.3.6 Comparison

In conclusion, no existing work covers all of the requirements well. Comparison among the different categories in relation to the features and the requirements is shown in Table 4.1 and Table 4.2, respectively.

4.4 Extractive Publish/Subscribe Summarisation System

An extension of Pub/Sub is proposed, in this chapter, that can overcome all of the aforementioned issues, limitations, and gaps; the Extractive Publish/Subscribe Summarisation System that is based on an embedding-based DBSCAN clustering and a geometric-based top-k ranking (PubSum approach) that create graph-based diverse entity summaries (notifications) for diversity-aware subscriptions. In this section, an overview of the architecture of the proposed system as well as details of the PubSum approach are described.

Table 4.1: Features as addressed by related work

Approach	**Query Type**	**Data Type**	**Expressiveness Type**	**Model Dependency**	**Unsupervised Learning**	**Domain Agnostic**	**Streaming/IoT**
			Diverse Entity Summarisation				
FACES [50]	keywords	RDF graphs	conceptual & contextual	ontologies & thesauri	✓		
ES-LDAext [51]				embedding models		✓	
DeepLENS [52]							
			Diversity in Pub/Sub				
PrefSIENA [54]	preferential representational coupled attribute-value pairs	attribute-value pairs	de-duplication & lack of commonality		✓	✓	✓
Chen et al. [55]	topics/keywords	text			✓	✓	✓
Hmedeh et al. [56]					✓	✓	✓
			Approximate Semantic Matching in Pub/Sub				
Hasan et al. [57]	semantic decoupled triples	RDF graphs	N/A	distributional semantics & thesauri	✓	✓	✓
Hasan et al. [39]	semantic decoupled attribute-value pairs	attribute-value pairs		distributional semantics	✓	✓	✓
Alhakbani et al. [58]				distributional semantics & taxonomies	✓		✓
			Semantic Engines in Pub/Sub				
G-TOPSS [59]	representational coupled tuples	RDF graphs	N/A	ontologies	✓		✓
Esposito et al. [60]	topics						✓
PubSum	representational decoupled subscriptions	RDF graphs	conceptual & contextual	embedding models	✓	✓	✓

Table 4.2: Requirements as addressed by related work

Approach	**R1 - Usability**	**R2 - User Expressibility**	**R3 - Data Expressiveness**	**R4 - User Effectiveness**	**R5 - Data Effectiveness**	**R6 - Efficiency**
Diverse Entity Summarisation						
FACES [50]	++	N/A	+	++	N/E	N/A
ES-LDAext [51]	++	N/A	++	++	N/E	N/A
DeepLENS [52]	++	N/A	+	++	N/E	N/A
Diversity in Pub/Sub						
PrefSIENA [54]	-+	-+	-+	+	+	++
Chen et al. [55]	++	-	-+	+	N/E	-+
Hmedeh et al. [56]	++	-	-+	+	-	-+
Approximate Semantic Matching in Pub/Sub						
Hasan et al. [57]	+	-+	N/A	+	N/A	++
Hasan et al. [39]	+	+	N/A	++	N/A	++
Alhakbani et al. [58]	+	+	N/A	++	N/A	++
Semantic Engines in Pub/Sub						
G-TOPSS [59]	-+	-+	N/A	N/E	N/A	++
Esposito et al. [60]	++	-	N/A	N/E	N/A	-+

++ the requirement dimension is well covered
+ the requirement dimension is partially covered with positive results
−+ there is an attempt to address the requirement dimension but the solution is not effective
− the requirement dimension is poorly covered
−− the requirement dimension is very poorly covered
N/A the requirement dimension is not addressed or the focus of the research
N/E the requirement dimension is not evaluated

4.4.1 Architecture

The architecture of the approach is illustrated in Fig. 4.4. Publishers generate publications concerning different entities with one of them being Property1 that is situated in Texas. The publishers use different distributions to generate entity data defined by the *Distribution*. This component contains two available distributions; TakeAll, which suggests that publishers should generate all unique available triples, and Zipf, which suggests that publishers should generate triples based on popularity. A subscriber generates a *Diversity-aware Subscription*, where one is interested in an extractive summary of Property1, that is the top-5 diverse entity information deriving from the analysis of data taken from count sliding windows of size 16 and slide 5. All publications and subscriptions enter the *Extractive Publish/Subscribe Summarisation System*, where they are stored and managed.

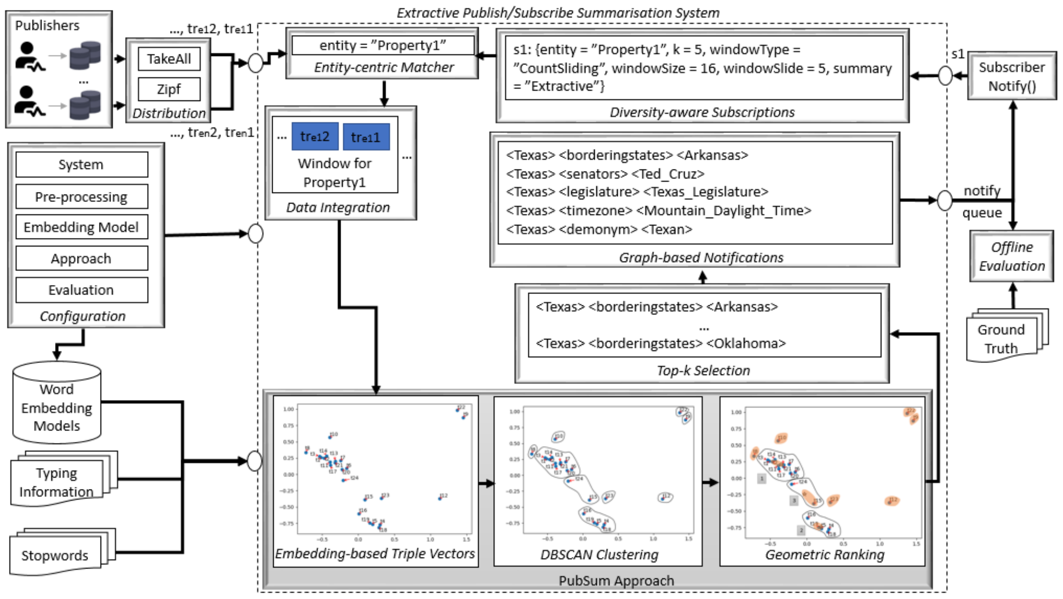

Extractive Publish/Subscribe Summarisation System
Publishers
..., tre12, tre11
..., tren2, tren1
TakeAll
Zipf
Distribution
entity = "Property1"
Entity-centric Matcher
s1: {entity = "Property1", k = 5, windowType = "CountSliding", windowSize = 16, windowSlide = 5, summary = "Extractive"}
Diversity-aware Subscriptions
s1
Subscriber
Notify()
tre12
tre11
Window for Property1
Data Integration
System
Pre-processing
Embedding Model
Approach
Evaluation
Configuration
<Texas> <borderingstates> <Arkansas>
<Texas> <senators> <Ted_Cruz>
<Texas> <legislature> <Texas_Legislature>
<Texas> <timezone> <Mountain_Daylight_Time>
<Texas> <demonym> <Texan>
Graph-based Notifications
notify
queue
Offline Evaluation
Ground Truth
<Texas> <borderingstates> <Arkansas>
...
<Texas> <borderingstates> <Oklahoma>
Top-k Selection
Word Embedding Models
Typing Information
Stopwords
Embedding-based Triple Vectors
DBSCAN Clustering
Geometric Ranking
PubSum Approach

Figure 4.4: Architecture of the Extractive Publish/Subscribe Summarisation System.

The stored publications and subscriptions are examined by the *Entity-centric Matcher*. If there is a match between the entity requested by the subscribers and the entity to which publications refer, like in this example Property1, then, a match occurs. After a match, two kinds of listeners are connected to the system, one for the publishers and one for the subscribers in order to observe whenever new elements have been generated related to the entity for future processing. Afterwards, a window is created based on the selected windowing policy, which accommodates all publications of Property1 as they are generated (*Data Integration*). Other windows may be created later on based on the matched entities; therefore, each window is related to a different entity. In the case of the creation of time windows, a clock monitors the time in order to manage them, whereas in the case of count windows a counter is calculating all elements within a window. As elements get in the window a trigger is activated and they are incrementally processed by the *Embedding-based Triple Vectors* methodology. The Embedding-based Triple Vectors approach is incrementally pre-processing each publication that enters a window by extracting its triple information and transforming it into a vector by using a pre-selected word embedding model. In the example, the vector space of the window triples of the entity Property1 (i.e. its region) is depicted after their transformation. This incremental process ends once the window reaches its full capacity, that is the counter is equal to the subscription's windowSize, which is 16. If the window was a time window, then, the window would reach its full capacity depending on the time of the clock and if it has exceeded the maximum timestamp defined by the subscriber. Afterwards, *DBSCAN Clustering* takes place, where the density-based regions of the embedding-based semantic space of the triples within a window are found resulting in conceptual clusters. In the example, 9 conceptual clusters are depicted. Then, the conceptual clusters are ranked by *Geometric Ranking* based on the importance and diversity of the triples in each cluster. In the example, the most important triple for each cluster is depicted by an orange area. Once the ranking has taken place, the window is evicted (cleared) according to the windowing policy and awaits the new elements. The notification process is, then, triggered that checks the state of all subscriptions in order to notify all interested parties. The ranked elements of the window undergo *Top-k Selection* depending on the subscription's k. In the example, the subscriber will be notified with the top-5 most diverse and important triples of the entity Property1. The extractive summarisation process is, then, completed and the newly constructed timestamped *Graph-based Notification* is sent to the subscriber and to a queue that along with necessary metadata will be stored externally in order to be evaluated offline *Offline Evaluation*. This whole process ends once a pre-defined time duration has been reached. Once the run is completed, the Offline Evaluation is responsible for calculating all the necessary metrics.

The last component to be analysed is the *Configuration* that is responsible for holding static values of parameters that concern a wide range of components. Specifically, it contains parameters regarding the system (e.g. duration of streaming, distribution type, windowing policies, and publisher and subscriber policies), pre-processing (e.g. stopwords removal, POS tagging, and typing information), word embedding model (e.g. type of model and corpus of model), approach (e.g. type of algorithm/approach like PubSum or baselines, PubSum's parameters like DBSCAN parameters as well as distance measure, and FACES-adapted parameters like Cobweb parameters as well as triple store location) and evaluation (e.g. storage of results/notifications, ground truth locations, and metrics to be calculated).

4.4.2 PubSum Approach

The main extractive diverse summarisation approach (PubSum approach) consists of two phases: 1) conceptual clustering, and 2) geometric-based top-k ranking of triples. The first phase is a combination of embeddings and DBSCAN clustering, while the second one consists of similarity metrics and some pre-defined rules.

Conceptual Clustering

The first step of the conceptual clustering phase is to turn a triple extracted from a publication into a vector in order to define its position in a vector space for future clustering. This could be achieved by using either knowledge graph entity embeddings (e.g. TransE [153], RDF2Vec [181]) that focus on the graph structure, or word embeddings (e.g. Word2Vec [182]) that focus on the semantic importance of words. In this thesis, the work focuses only on star-like graphs and due to a diversity-oriented summarisation goal, it is deemed that word embeddings are more suitable for representing triples as vectors. Nevertheless, word embeddings cannot be directly used on triples as they would in a document; therefore, a different approach needs to be defined as analysed in Algorithm 1.

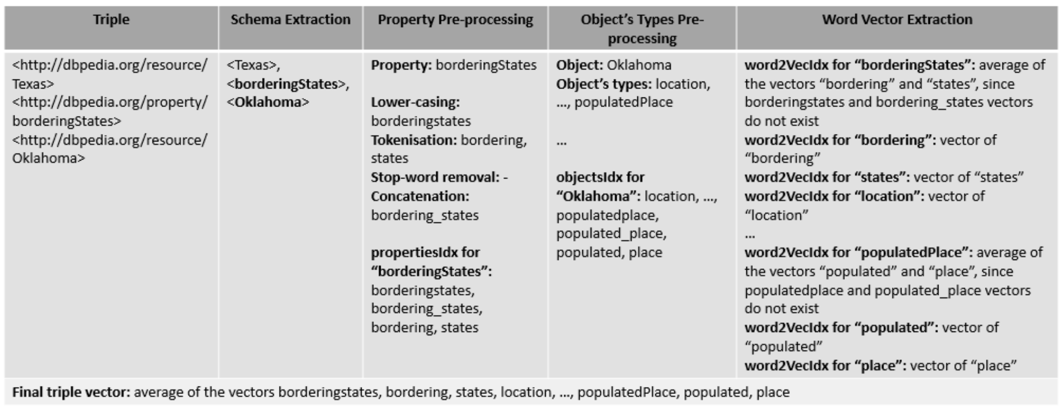
Algorithm 1 is an incremental step of the PubSum approach, where each triple is transformed into an embedding-based vector. In line 1, the Embedding-based Triple Vectors function gets as input the triple tr of the generated publication, a word embedding model $word2VecModel$, a list of the current elements within the window $windowElements$, stopwords $stopwords$, and the typing information of all objects of the triples $objectTypes$. In line 3, the new triple that has entered

Algorithm 1 Embedding-based Triple Vectors

```
1: function Embedding-based Triple Vectors($tr, word2VecModel, windowElements, stopwords, objectTypes$)  ▷ $tr$: triple, $word2VecModel$: embedding model, $objectTypes$: objects' typing information
2:     if $\neg endOfWindow$ then
3:         $processedTr \leftarrow schemaExtraction(tr)$
4:         $duplicate \leftarrow getDuplicates(processedTr)$
5:         if $duplicate$ then
6:             $windowElements.replace(duplicate, processedTr)$
7:         else
8:             $property \leftarrow attributeOfTriple(processedTr)$
9:             $object \leftarrow valueOfTriple(processedTr)$
10:            $propertiesIdx \leftarrow preprocessProperty(property, stopwords)$
11:            $objectsIdx \leftarrow preprocessObject(object, stopwords, objectTypes)$
12:            $word2VecIdx \leftarrow createWord2VecIdx(property, word2VecModel, propertiesIdx)$
13:            for $type \leftarrow objectsIdx$ do
14:                $word2VecIdx \leftarrow createWord2VecIdx(type, word2VecModel, objectsIdx)$
15:            end for
16:            $trVector \leftarrow averageOfAllWordVectorsOfTriple(word2VecIdx)$
17:        end if
18:        $statementVectors \leftarrow StatementVector(timestamp, processedTr, trVector)$
19:        return $statementVectors$          ▷ dynamic list of timestamped triple vectors of window
20:    end if
21: end function
```

the window is pre-processed by extracting its RDF schema for understandability purposes. Line 4, checks the elements of the window for duplicates and if found, then, the new triple will replace any old duplicate ones (line 5-6) as recent data has higher priority. In case of duplication, the whole process will stop since the triple has already been transformed into a vector in the past. If a duplicate is not found, then, in lines 7-9, the property and the object of the triple are extracted. In this way, a triple is transformed into an equivalent document sentence, in which word embeddings models could be directly applied. The subject is not considered since it represents the entity; therefore, it is the same for all elements within the window. Lines 10 and 11 pre-process the property and the object's type/types, respectively. Specifically, pre-processing involves lower-casing, tokenising, removing stop-words and concatenating each token with an underscore. All words ranging from original to tokens, to concatenated ones are stored into a corresponding index ($propertiesIdx$ and $objectsIdx$) so that they are not pre-processed again. The reason an emphasis is given in the object's type/types instead of the object itself is because an object does not represent a word of linguistic value (as opposed to the property), and types are the kind of information that can cluster groups of entities with the same properties [183]; therefore, they offer contextual value in a vector space. In lines 12-15, an index is created ($word2VecIdx$) that contains the equivalent word embedding vectors of the words contained in the $propertiesIdx$ and $objectsIdx$. Specifically, if the $word2VecModel$ does not contain the original word (original property or type), then, priority is given to the concatenated ones and, then, to the average of the vectors of the word's tokens. This means that a missing original or concatenated word is represented only by the average vector of their tokens. Phrases are represented in the same way, which may pose some risks in altering their original meaning; nevertheless, only embedding models that have been trained on these phrases can be accurate. The purpose of the $word2VecIdx$ is for not transforming already visited words into vectors again. The final triple vector is extracted in line 16, where an average of the vectors of the original words or concatenated ones along with those of their tokens is calculated. In line 18, the triple vector is timestamped based on the publication's original timestamp and it is stored in the list $statementVectors$ that is returned in line 19 for further use.

An example of the process is given in Fig. 4.5, where the triple <.../Texas> <.../borderingStates> <.../Oklahoma> is transformed to an embedding-based vector. A vector space for the entity Texas is depicted in Fig. 4.6, where the available triple information is presented without any schematic prefixes for visualisation purposes. It is observed, in Fig. 4.6, that conceptually and contextually related triples are closer in the vector space (e.g. borderingStates - country, candidate - senators etc.).



Triple	Schema Extraction	Property Pre-processing	Object's Types Pre-processing	Word Vector Extraction
<http://dbpedia.org/resource/Texas> <http://dbpedia.org/property/borderingStates> <http://dbpedia.org/resource/Oklahoma>	<Texas>, **<borderingStates>**, **<Oklahoma>**	**Property:** borderingStates **Lower-casing:** borderingstates **Tokenisation:** bordering, states **Stop-word removal:** - **Concatenation:** bordering_states **propertiesIdx for "borderingStates":** borderingstates, bordering_states, bordering, states	**Object:** Oklahoma **Object's types:** location, ..., populatedPlace ... **objectsIdx for "Oklahoma":** location, ..., populatedplace, populated_place, populated, place	**word2VecIdx for "borderingStates":** average of the vectors "bordering" and "states", since borderingstates and bordering_states vectors do not exist **word2VecIdx for "bordering":** vector of "bordering" **word2VecIdx for "states":** vector of "states" **word2VecIdx for "location":** vector of "location" ... **word2VecIdx for "populatedPlace":** average of the vectors "populated" and "place", since populatedplace and populated_place vectors do not exist **word2VecIdx for "populated":** vector of "populated" **word2VecIdx for "place":** vector of "place"
Final triple vector: average of the vectors borderingstates, bordering, states, location, ..., populatedPlace, populated, place				

Figure 4.5: An example of the Embedding-based Triple Vectors process.

Algorithm 2 is a batch step of the PubSum approach, where all triples in the window are conceptually clustered by DBSCAN. In line 1, the DBSCAN

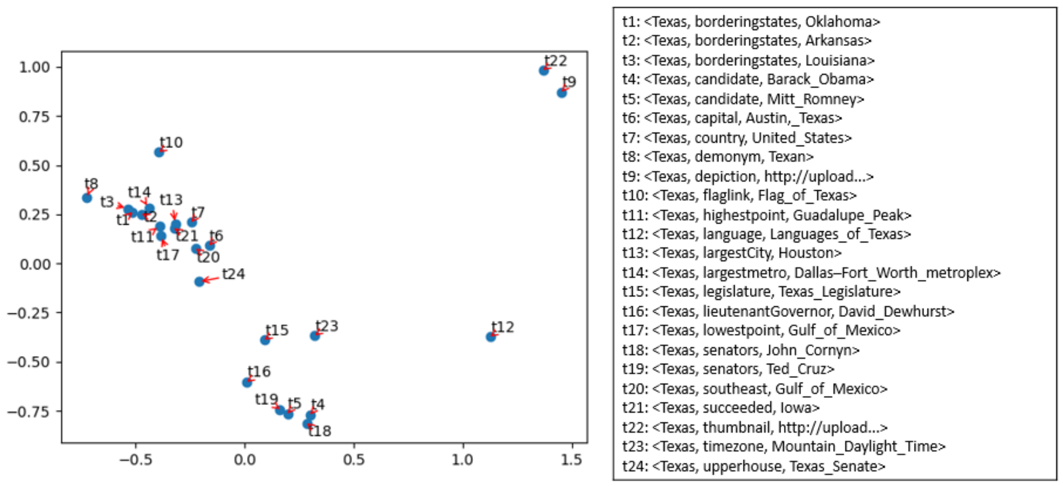


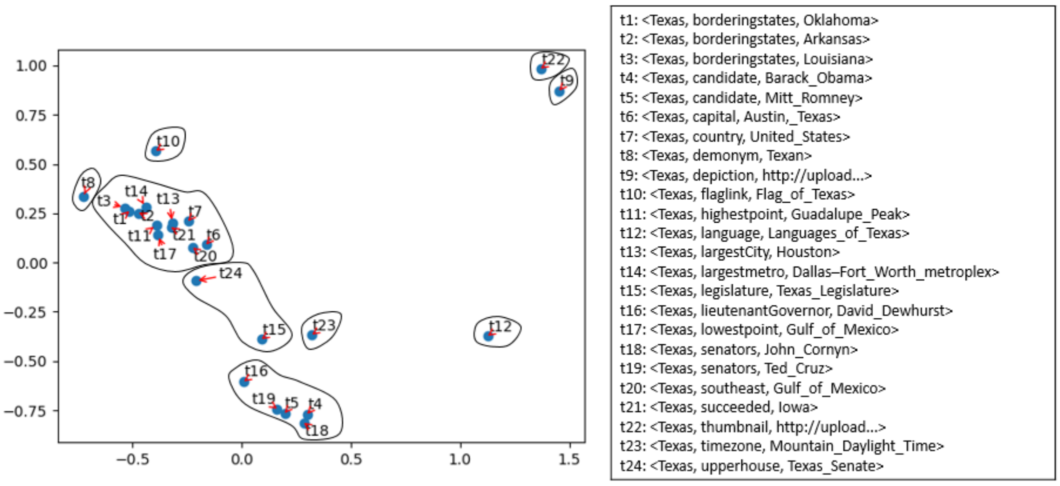
Figure 4.6: The vector space for the entity Texas after the Embedding-based Triple Vectors process with the Word2Vec GOOGLENEWS model.

Clustering function gets as input the timestamped triple vectors of the window $statementVectors$ from the Embedding-based Triple Vectors function. In lines 3-5, all triple vectors are stored into the training set $trainingData$. A DBSCAN clustering model is, then, called in line 6 that uses the training set, Euclidean distance as a distance measure and the pre-defined $minPts$ and ε parameters. The clusters are formed in line 7, where they are mapped to the corresponding triples, in line 8. This mapping is stored in $clusterIDStatements$ that is returned in line 9 for further use.

Algorithm 2 DBSCAN Clustering

```
1: function DBSCAN Clustering($statementVectors$)
2:     if $endOfWindow$ then
3:         for $statementVector \leftarrow statementVectors$ do
4:             $trainingData \leftarrow statementVector.vector$
5:         end for
6:         $dbscan \leftarrow DBSCAN(trainingData, EuclideanDistance, minPts, \varepsilon)$
7:         $clusters \leftarrow getClusters(dbscan)$
8:         $clusterIDStatements \leftarrow ClusterIDStatement(clusters.ID, statementVector)$
9:         return $clusterIDStatements$                ▷ map of timestamped triples to clusters
10:    end if
11: end function
```

An example of DBSCAN clustering is given in Fig. 4.7, where the densely-connected regions of the semantic space of the triples has been found. This translates to identifying regions of similar or related words in a vector space. In the example, there are 9 clusters in total, where conceptually and contextually



t1: <Texas, borderingstates, Oklahoma>
t2: <Texas, borderingstates, Arkansas>
t3: <Texas, borderingstates, Louisiana>
t4: <Texas, candidate, Barack_Obama>
t5: <Texas, candidate, Mitt_Romney>
t6: <Texas, capital, Austin,_Texas>
t7: <Texas, country, United_States>
t8: <Texas, demonym, Texan>
t9: <Texas, depiction, http://upload...>
t10: <Texas, flaglink, Flag_of_Texas>
t11: <Texas, highestpoint, Guadalupe_Peak>
t12: <Texas, language, Languages_of_Texas>
t13: <Texas, largestCity, Houston>
t14: <Texas, largestmetro, Dallas–Fort_Worth_metroplex>
t15: <Texas, legislature, Texas_Legislature>
t16: <Texas, lieutenantGovernor, David_Dewhurst>
t17: <Texas, lowestpoint, Gulf_of_Mexico>
t18: <Texas, senators, John_Cornyn>
t19: <Texas, senators, Ted_Cruz>
t20: <Texas, southeast, Gulf_of_Mexico>
t21: <Texas, succeeded, Iowa>
t22: <Texas, thumbnail, http://upload...>
t23: <Texas, timezone, Mountain_Daylight_Time>
t24: <Texas, upperhouse, Texas_Senate>

Figure 4.7: An example of DBSCAN Clustering for the entity Texas.

related triples (e.g. borderingStates - country, candidate - senators etc.) have been clustered together.

Geometric-based Top-k Ranking

The final phase of the PubSum approach is the geometric-based top-k ranking of triples, which is analysed in Algorithm 3. Algorithm 3 is a geometric-based ranking methodology that measures the importance of a triple within and among all conceptual clusters of an entity. In line 1, the Geometric Ranking function gets as input the $statementVectors$ and $clusterIDStatements$ of Algorithm 1 and Algorithm 2, respectively. The function aims to score the triples of the clusters based on significance, but also to define a selection order that chooses which triples will be selected first for the final summary. Line 2 proceeds with the scoring of each triple that is explained in lines 10-24. For each cluster, its centroid is calculated. Since DBSCAN does not create centroids, the average value of all triple vectors that belong to the cluster constitutes the cluster's centroid as shown in lines 11-13. In lines 14-19, the score of each triple within a cluster is calculated based on its distance to the cluster's centroid. The distance could be either Euclidean distance or cosine similarity. A $scoredStatement$ is then created in line 20 that contains the timestamped scored triple based on the publication's original timestamp and it is stored in the list $scoredStatementsOfCluster$. Triples that are closer to the centroid are more representative of the conceptual cluster than others; therefore, the list is sorted in descending order, in the case of cosine similarity, and ascending,

Algorithm 3 Geometric Ranking

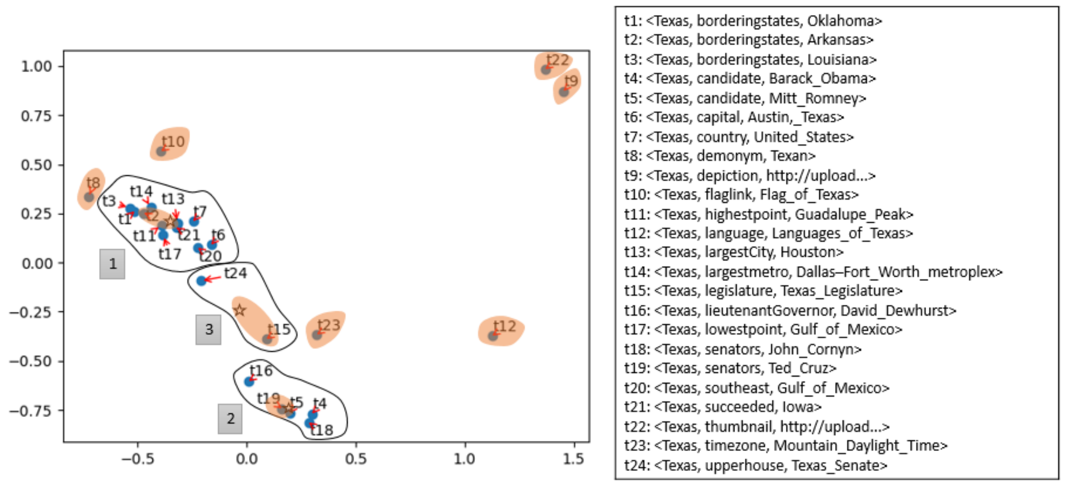
```
 1: function GEOMETRIC RANKING($statementVectors, clusterIDStatements$)
 2:     $scoredStatementsOfClusters \leftarrow score(statementVectors, clusterIDStatements)$
 3:     $sizePerCluster \leftarrow getSize(clusterIDStatements)$
 4:     $orderedClusters \leftarrow sort(sizePerCluster)$
 5:     while $\neg scoredStatementsOfClusters \leftarrow \emptyset$ do
 6:         $scoredStatements \leftarrow startSelectionRound(orderedClusters, scoredStatementsOfClusters)$
 7:     end while
 8:     return $scoredStatements$                ▷ ordered timestamped scored triples of window
 9: end function
10: function SCORE($statementVectors, clusterIDStatements$)
11:     for $cluster \leftarrow clusterIDStatements$ do
12:         $clusterStatementVectors \leftarrow cluster.statementVectors$
13:         $centroid \leftarrow \frac{sumOfClusterStatementVectors}{sizeOfCluster}$
14:         for $statementVector \leftarrow clusterStatementVectors$ do
15:             if $Euclideandistance$ then
16:                 $score \leftarrow euclideanDistance(statementVector.vector, centroid)$
17:             else
18:                 $score \leftarrow cosineSimilarity(statementVector.vector, centroid)$
19:             end if
20:             $scoredStatementsOfCluster \leftarrow ScoredStatement(timestamp, triple, score)$
21:         end for
22:         $scoredStatementsOfClusters \leftarrow sort(scoredStatementsOfCluster)$
23:     end for
24:     return $scoredStatementsOfClusters$
25: end function
26: function STARTSELECTIONROUND($orderedClusters, scoredStatementsOfClusters$)
27:     for $cluster \leftarrow orderedClusters$ do
28:         for $scoredStatement \leftarrow scoredStatementsOfCluster$ do
29:             if $property\ not\ selected\ or\ end\ is\ reached$ then
30:                 $scoredStatements \leftarrow scoredStatement$
31:                 $scoredStatementsOfCluster.remove(scoredStatement)$
32:             end if
33:         end for
34:     end for
35:     return $scoredStatements$
36: end function
```

otherwise, in line 22. If there are ties among the scores, then, a random sorting is made. The list is returned in line 24 for further use. In line 3, the size of each cluster is observed, which is used in line 4 for defining the order in which each cluster will be selected for choosing the best every time triple for the final summary. Priority is given to clusters with bigger sizes, but if the size is tied, then, a random cluster will be selected first. In lines 5-6, the selection round is taking place described in lines 26-36. This is a continuous process until all triples have been selected. In this process, a cluster is visited based on its order (line 27) and the first-sorted $scoredStatement$ within the cluster is selected (line 28). If the property of this triple has not already been selected in another round or if all properties of the cluster have already been selected once in other rounds (line 29), the $scoredStatement$ is chosen as the next to be selected for the final summary in line 30 and it is removed from the collection of $scoredStatements$ in line 31, otherwise, the next highest-sorted $scoredStatement$ within the cluster is selected and so on. This provides more diversity in the top properties of the final summary even among members of the same conceptual cluster. All clusters are visited once in the first round and if the cluster is of size 1, then, the only element is chosen. The final order of the selected $scoredStatements$ is given in line 8 and it is used for top-k selection.

An example of the geometric-based ranking for the entity Texas is given in Fig. 4.8. The stars represent the centroids of each cluster (clusters with only one element are also centroids to the cluster). The orange areas represent the triple that is chosen first from each cluster to be included in the final summary as it is the closest to the centroid, hence the most representative of the conceptual cluster. Priority (grey box) is given to the biggest cluster {t1, t2, t3, t6, t7, t11, t13, t14, t17, t20, t21}, and so on (clusters with only one element have random priority). The final top-5 diverse extractive summary is {t2, t19, t15, t23, t8}.

4.5 Evaluation

This section provides the evaluation of the approach that consists of the FACES-adapted baseline, the methodology followed, and the datasets used as well as the metrics and the final results.



t1: <Texas, borderingstates, Oklahoma>
t2: <Texas, borderingstates, Arkansas>
t3: <Texas, borderingstates, Louisiana>
t4: <Texas, candidate, Barack_Obama>
t5: <Texas, candidate, Mitt_Romney>
t6: <Texas, capital, Austin,_Texas>
t7: <Texas, country, United_States>
t8: <Texas, demonym, Texan>
t9: <Texas, depiction, http://upload...>
t10: <Texas, flaglink, Flag_of_Texas>
t11: <Texas, highestpoint, Guadalupe_Peak>
t12: <Texas, language, Languages_of_Texas>
t13: <Texas, largestCity, Houston>
t14: <Texas, largestmetro, Dallas–Fort_Worth_metroplex>
t15: <Texas, legislature, Texas_Legislature>
t16: <Texas, lieutenantGovernor, David_Dewhurst>
t17: <Texas, lowestpoint, Gulf_of_Mexico>
t18: <Texas, senators, John_Cornyn>
t19: <Texas, senators, Ted_Cruz>
t20: <Texas, southeast, Gulf_of_Mexico>
t21: <Texas, succeeded, Iowa>
t22: <Texas, thumbnail, http://upload...>
t23: <Texas, timezone, Mountain_Daylight_Time>
t24: <Texas, upperhouse, Texas_Senate>

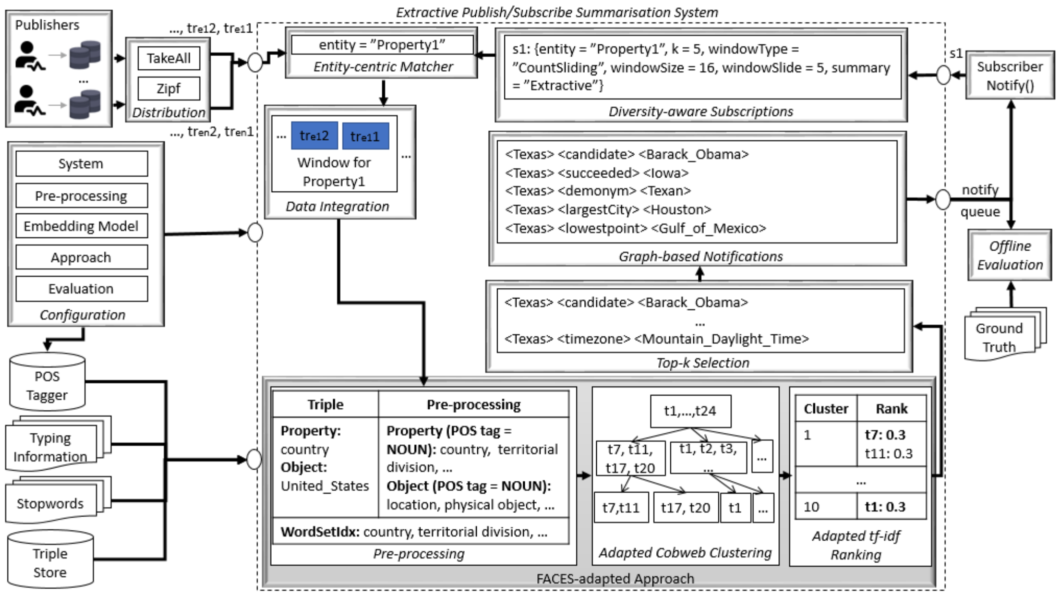
Figure 4.8: An example of the geometric-based ranking for the entity Texas (only the first selection round is depicted).

4.5.1 FACES-adapted Baseline

One of the baselines used for the evaluation of the PubSum approach is FACES, which is a static diverse entity summarisation methodology. FACES is divided into two stages: 1) hierarchical conceptual clustering of triples, and 2) ranking of triples. The first stage is done by a modified version of Cobweb clustering as well as the use of WordNet and a POS Tagger for pre-processing. The second stage is a combination of an adapted version of tf-idf that is based on the popularity and the informativeness of a triple within the DBpedia ontology and some pre-defined top-k selection rules. For more details, the reader is directed to Gunaratna et al. [50].

FACES is emphasised in this thesis as a baseline, compared to the other static entity summarisation approaches, as it is an unsupervised diverse entity summarisation approach that is the most relevant to this thesis' work. Nevertheless, all these approaches, including FACES, cannot be directly applied to the proposed Extractive Publish/Subscribe Summarisation System due to its dynamic and temporal nature. Therefore, FACES-adapted was implemented with its architecture being illustrated in Fig. 4.9.

All aforementioned components concerning publishers, distributions, subscribers, matchers, data integration, windowing policies, notifications, evaluation, and configuration have been extended for the FACES approach. As elements get in the window, a trigger is activated, and they are incrementally processed by



Extractive Publish/Subscribe Summarisation System
Publishers
TakeAll
Zipf
Distribution
entity = "Property1"
Entity-centric Matcher
s1: {entity = "Property1", k = 5, windowType = "CountSliding", windowSize = 16, windowSlide = 5, summary = "Extractive"}
Diversity-aware Subscriptions
Subscriber Notify()
Window for Property1
Data Integration
<Texas> <candidate> <Barack_Obama>
<Texas> <succeeded> <Iowa>
<Texas> <demonym> <Texan>
<Texas> <largestCity> <Houston>
<Texas> <lowestpoint> <Gulf_of_Mexico>
Graph-based Notifications
notify
queue
Offline Evaluation
Ground Truth
<Texas> <timezone> <Mountain_Daylight_Time>
Top-k Selection
System
Pre-processing
Embedding Model
Approach
Evaluation
Configuration
POS Tagger
Typing Information
Stopwords
Triple Store
Triple
Property: country Object: United_States
Property (POS tag = NOUN): country, territorial division, ...
Object (POS tag = NOUN): location, physical object, ...
WordSetIdx: country, territorial division, ...
Pre-processing
t1,...,t24
Adapted Cobweb Clustering
Cluster
Rank
t7: 0.3 t11: 0.3
t1: 0.3
Adapted tf-idf Ranking
FACES-adapted Approach

Figure 4.9: FACES-adapted directly applied to the proposed Extractive Publish/Subscribe Summarisation System.

Pre-processing methodology. The Pre-processing methodology is incrementally pre-processing each publication that enters a window by extracting its triple information and transforming it into a set of words. Specifically, pre-processing involves lower-casing, tokenising, removing stop-words, POS tagging of tokens, and extraction of their hypernyms from WordNet. This process is done on the property and the object's type/types of the triple. All words ranging from original to tokens to hypernyms are stored into a corresponding index ($propertiesIdx$ and $objectsIdx$) so that they are not pre-processed again. An index is created ($wordSetIdx$) that contains all the corresponding words contained in the $propertiesIdx$ and $objectsIdx$ for each triple so that already visited triples are not examined again. In the example, this process is shown for a triple with country and United States as property and object, respectively. This incremental process ends once the window reaches its full capacity. Afterwards, *Adapted Cobweb Clustering* takes place where all triples in the window are conceptually clustered. A training set is constructed based on the appearance or not of words from the $wordSetIdx$ for each triple. A Cobweb clustering model is, then, called that uses the training set and results in the construction of a pruned tree, where each leaf represents a cluster. In the example, {t7, t11}, {t17, t20}, and t1 constitute some of the conceptual clusters. Then, the conceptual clusters are ranked by *Adapted tf-idf Ranking* based on the popularity and informativeness of the triples (tf-idf like metric) within the DBpedia ontology and their diversity within each cluster. For the purposes of ranking, part of the DBpedia ontology is stored in a triple store. In the example, the most important triple for each cluster is depicted in bold. The order, in which each cluster will be selected for choosing the best every time triple for the final summary, is random. As in the case of PubSum, all clusters are visited once in the first round and properties that have already been selected in another round are avoided. The continuous process of selection ends once all triples have been selected. Once the ranking and selection have been finalised, the rest of the process is the same with the window being evicted and notification taking place, where the ranked elements undergo *Top-k Selection* depending on the subscription's k. In the example, the subscriber is notified with the top-5 most diverse and important triples of the entity Property1 (i.e. its region).

4.5.2 Datasets and Methodology

The FACES dataset[3] has been selected for the evaluation, which is based on DBpedia 3.9. The dataset contains 50 entities of different domains (e.g. politician, actor, etc.) with 44 distinct direct features on average per entity. Only object-type predicates are included in the dataset as they provide richer information. The data was pre-processed by keeping only the last part after a "/" or "#" in URIs so that the data makes more sense from the user perspective. Although the dataset is not directly linked to the use cases of Healthcare and Smart Cities, it provides real-world entity information that relates to people, things, and places. These entities could form patients (i.e. people) in a Healthcare use case or properties and their regions (i.e. things and places) in a Smart Cities one, on which external sources or historical events provide real-time additional information along with other sensors.

In order to observe the results of the PubSum approach, all entities need to contain their typing information, that is their class membership, category, or type. This information is also used by the FACES baseline. Nevertheless, this is not the case for all objects in DBpedia as the typing information may not only be missing, but it may be incorrect too. For example, in the FACES dataset after extracting all

[3]http://wiki.knoesis.org/index.php/FACES

available typing information from DBpedia, 54 of the object resources contained noisy/false types, like <Automobiles> <type> <MusicGenre> or <Chemistry> <type> <University>, and around 300 of them had missing types. In order to solve this issue, a combination of the approaches described below took place:

- The Virtuoso SPARQL Query Editor of DBpedia[4] was used by creating a SPARQL query that asked for the type of an object resource, in case there was some missing/additional information not found in the dataset files.

- Inspired by Van et al. [184], the pre-trained entity vectors with naming from the Freebase model[5] were used, and the word vectors of entities were extracted. The concept is that if entities with unknown types are closer in the vector space to entities with known types, then, they will share the same entity sets. This model had high chances of containing entities that related to specific names, places, etc. than other models. For example, <100_metres> and <200_metres> were clustered together, so both of them were given the type <SportsEvent> that <100_metres> had. There might be cases though where <Baywatch>, <Baywatch_Nights>, and <David_Hasselhoff> were clustered together, and although contextually it made sense, in reality, their types are different. This information could be used though for fine-grained entity typing. For example, each of the entities <Angelina_Jolie>, <Ben_Affleck>, and <Courteney_Cox> had types <Agent>, <NaturalPerson>, and <Person>, but a more fine-grained type could be <Actor> since they were clustered together. This is, nevertheless, out of the scope of this thesis' work.

- Some additional dataset files were extracted for each entity in question from DBpedia like "Categories", "Short Abstracts", and "wikiPageWikiLinks" that can potentially help in the extraction of types.

- If none of the above succeeded because an entity was missing, then, a manual type was given. For example, the entities similar to <http://wifo5-03.informatik.uni-mannheim.de/flickrwrappr/photos/Texas> were given the type <Photos>.

Each publisher generates a stream related to an entity. There is only one subscriber that generates subscriptions based on the entities. The streaming rate

[4]`http://dbpedia.org/sparql`
[5]`https://code.google.com/archive/p/word2vec/`

of the publishers is constant and the selection of which triple will be generated each time follows one of the following distributions:

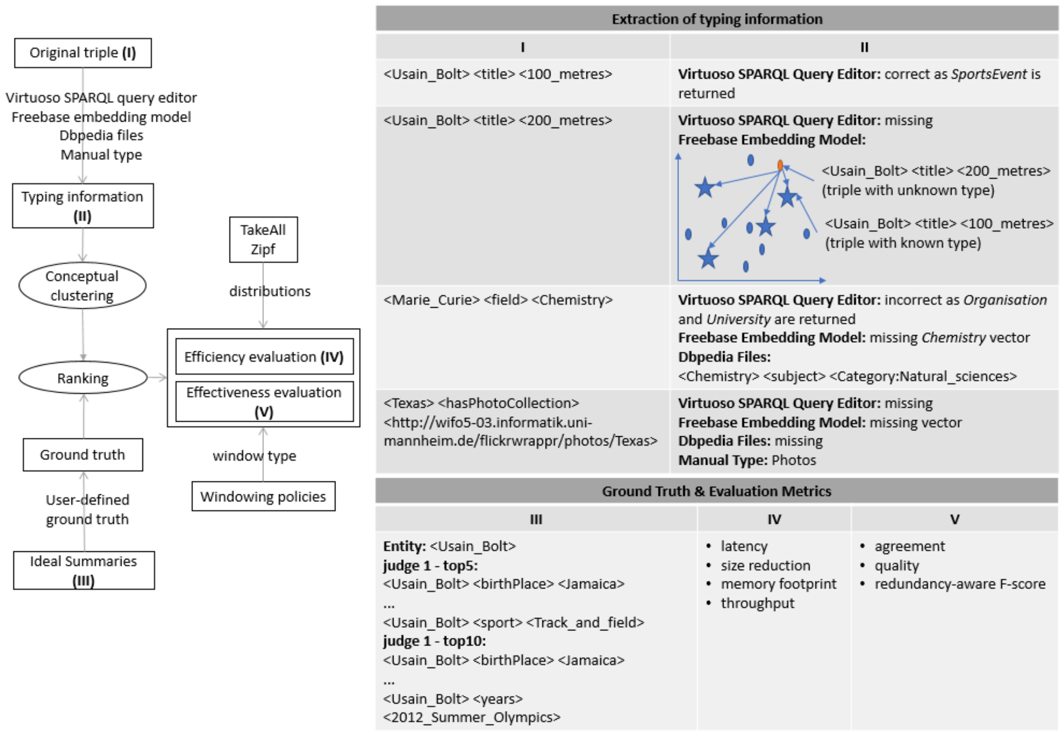
TakeAll: A publisher generates all unique triples that are available for an entity in the dataset.

Zipf: A publisher generates a triple that is available for an entity in the dataset based on popularity. Specifically, each triple in a collection is checked for its popularity by combining the frequency of occurrence of the property and the object in the collection. Then, the selection of which triple a source will generate follows a Zipf distribution [185] based on this popularity.

The evaluation methodology is illustrated in Fig. 4.10. Each original triple (step I) is observed for its typing information (step II). Then, conceptual clustering and ranking take place so that PubSum's approach efficiency and effectiveness are examined. The efficiency (step IV) involves metrics like latency, throughput, etc., whereas the effectiveness (step V) involves metrics like agreement, quality, and redundancy-aware F-score. The agreement and quality are based on a user-defined ground truth (step III), which is provided by FACES. This ground truth contains ideal summaries provided by 15 human judges with a background in the Semantic Web that were asked to select ideal triples for specific entities for k = 5 and k = 10 triples. Each entity has at least 7 ideal summaries from 7 different judges, which constitutes the gold standard. Both efficiency and effectiveness evaluation take place after the final ranking has occurred for all triples. An example of the evaluation methodology is given in Fig. 4.10, where the final types of <100_metres> and <200_metres> are *SocietalEvent*, *Event* and *SportsEvent*, the types of <Chemistry> are *Science* and *NaturalScience*, and the type of <.../photos/Texas> is *Photos*. No schematic prefixes are included in the triples and types for visualisation purposes.

All runs took place in a laptop with Intel(R) Core(TM) i7-6600U CPU@2.60GHz 2.80GHz, and 16GB of RAM. Regarding the implementation, for the RDF models, Apache Jena[6] was used, for the embedding model deeplearning4j[7] was used and for DBSCAN Smile[8] was used. Regarding FACES-adapted, WordNet of extJWNL[9] was used, for tagging Apache OpenNLP[10] was used, for the proposed version of

[6]`https://jena.apache.org/`
[7]`https://deeplearning4j.org/`
[8]`https://haifengl.github.io/smile/nlp.html`
[9]`http://extjwnl.sourceforge.net/`
[10]`https://opennlp.apache.org/`



Extraction of typing information	
I	II
<Usain_Bolt> <title> <100_metres>	**Virtuoso SPARQL Query Editor:** correct as *SportsEvent* is returned
<Usain_Bolt> <title> <200_metres>	**Virtuoso SPARQL Query Editor:** missing **Freebase Embedding Model:** <Usain_Bolt> <title> <200_metres> (triple with unknown type) <Usain_Bolt> <title> <100_metres> (triple with known type)
<Marie_Curie> <field> <Chemistry>	**Virtuoso SPARQL Query Editor:** incorrect as *Organisation* and *University* are returned **Freebase Embedding Model:** missing *Chemistry* vector **Dbpedia Files:** <Chemistry> <subject> <Category:Natural_sciences>
<Texas> <hasPhotoCollection> <http://wifo5-03.informatik.uni-mannheim.de/flickrwrappr/photos/Texas>	**Virtuoso SPARQL Query Editor:** missing **Freebase Embedding Model:** missing vector **Dbpedia Files:** missing **Manual Type:** Photos

Ground Truth & Evaluation Metrics		
III	IV	V
Entity: <Usain_Bolt> **judge 1 - top5:** <Usain_Bolt> <birthPlace> <Jamaica> ... <Usain_Bolt> <sport> <Track_and_field> **judge 1 - top10:** <Usain_Bolt> <birthPlace> <Jamaica> ... <Usain_Bolt> <years> <2012_Summer_Olympics>	• latency • size reduction • memory footprint • throughput	• agreement • quality • redundancy-aware F-score

Figure 4.10: Evaluation methodology of PubSum approach.

Cobweb algorithm by FACES the corresponding class by MOA[11] was modified, and for the proposed version of tf-idf by FACES a TDB Triple Store[12] was used that stores part of the DBpedia ontology.

4.5.3 Metrics

Several metrics have been used to evaluate the effectiveness and efficiency of the approach. The effectiveness evaluation focuses on the correctness, whereas the efficiency one focuses on the performance.

Correctness

Correctness consists of the agreement, quality, and redundancy-aware F-score metrics.

[11]`https://moa.cms.waikato.ac.nz/`
[12]`https://jena.apache.org/documentation/tdb/`

Agreement and Quality The agreement and quality metrics identify if the PubSum approach produces correct and trustworthy summaries that are appealing to human judgement. Specifically, the agreement Agr defines how consistent the ideal summaries of the user-defined ground truth are between one another, and the quality Q_t defines the commonalities between the human-defined ideal summaries and the approach's summaries for each entity. The agreement metric of FACES and RELIN [154] was used as well as a proposed time-dependent adaptation of their original quality metric as defined below:

$$Agr = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} |Summ_i^I(e) \cap Summ_j^I(e)| \tag{4.4}$$

$$Q_t = \frac{1}{n} \sum_{i=1}^{n} |\frac{Summ_t(e) \cap (Summ_i^I(e) \cap WTr_t(e))}{Summ_i^I(e) \cap WTr_t(e)}| \tag{4.5}$$

where n is the number of summaries, $Summ_i^I(e)$ is the i-th ideal summary for an entity e, $Summ_t(e)$ is the approach's summary in time t, and $WTr_t(e)$ are the triples of entity e existing in the window in time t.

This chapter's proposed quality metric is dependent on time since this is a dynamic entity summarisation, and static ideal summaries might contain information that is not yet known to the system; that is, it has not been published yet. Therefore, each $Summ_it^I(e)$ contains only the common triples between the already known triples in the system $WTr_t(e)$ and the ones selected from each judge. Then each of these time-dependent ideal summaries is checked for commonalities with the approach's summary $Summ_t(e)$ that has been extracted at that specific time. In the case of duplicate triples, these commonalities are only counted once. In the quality metric in FACES and RELIN, there is no use of a denominator, because, for example, k = 10 applies for all ideal summaries (e.g. 2/10 or 8/10 common triples), but in this chapter's work case the k is dependent on the commonalities between the $WTr_t(e)$ and the $Summ_i^I(e)$. Therefore, diverse results (e.g. 2/8 or 5/6 common triples) might occur so the denominator is used for normalisation.

Redundancy-aware F-score The metrics of redundancy-precision and redundancy-recall defined in Zhang et al. [186] are used, and through these, the redundancy-aware F-score is calculated. For this chapter's work, "redundant" relates only to duplicate triples. The score is defined as:

$$Red\_pr = \frac{R^-}{R^- + N^-} \quad \text{and} \quad Red\_rec = \frac{R^-}{R^- + R^+} \tag{4.6}$$

$$Red\_F - score = 2 \times \frac{Red\_pr \times Red\_rec}{Red\_pr + Red\_rec} \tag{4.7}$$

where R^- is the set of non-delivered redundant triples, N^- is the set of non-delivered non-redundant ones, and R^+ is the set of delivered redundant ones.

Performance

The performance consists of the end-to-end latency, size reduction, memory footprint, and throughput metrics analysed below.

End-to-End Latency The end-to-end latency is the time it takes between the publication of an event until its delivery to the subscriber as a notification. Since this work's summaries/notifications involve multiple publications, the end-to-end latency is the time it takes between the earliest published event in the fusion until the time of the summary's delivery to the subscriber. It is defined as:

$$end - to - end\ latency = t_{notification} - t_{first\ publication\ in\ notification} \tag{4.8}$$

Size Reduction This metric is split into: 1) the reduction in the number of messages that the subscriber receives, and 2) the reduction in the number of redundant/duplicate messages. It is defined as:

$$size\ reduction = \#forwarded\ or\ duplicate\ messages \tag{4.9}$$

Memory Footprint This metric involves the memory used by the different approaches not during the run, but for the use of Word2Vec models or ontologies. It is defined as:

$$memory\ footprint = memory\ of\ model\ or\ ontology \tag{4.10}$$

Throughput Throughput is the number of triples/events the system is able to analyse in a specific amount of time. It is defined as:

$$throughput = \frac{\#events\ in\ engine}{duration\ of\ streaming} \tag{4.11}$$

4.5.4 Results

The results are analysed for the following approaches: 1) the typical Pub/Sub approach, but with fusion (Non-top-k Fused), where all events regarding an entity are fused in a window and sent as a notification to the subscriber with no conceptual clustering and ranking involved, 2) FACES-adapted, where according to the original authors [50] the use of $Cobweb{-}cut{-}off = 5$ and $Cobweb{-}path{-}level = 3$ yields the best results, and 3) the proposed PubSum approach that uses three pre-trained Word2Vec models (GOOGLENEWS, VECTORS, and PHRASE)[13], where $\varepsilon = 1$ and $minPts = 1$ for DBSCAN yields the best results as well as the choice of Euclidean distance for ranking.

The results are shown below for 50 publishers that generate 100 triples each, where each publisher is related to an entity. One subscriber generates 50 subscriptions, one for each entity, with different window policies. Different distributions are used with $Zipf - exponent = 1$ for Zipf distribution.

Agreement

The average agreement results for all entities are $Agr = 1.917$ and $Agr = 4.579$ for $k = 5$ and $k = 10$, respectively. It is observed that there is a good agreement among judges with almost 2 out of 5 and 5 out of 10 triples being common. Table 4.3 illustrates the average agreement distribution for all entities among judges for the different k values. As expected, the values for $k = 5$ are lower than that of $k = 10$, as the less the number of triples, the less probable an agreement is. This shows that when a user is presented with a smaller summary, then, stricter criteria take place of what this ideal summary should be. Some judges, like judges 6, 7, and 11 have the lowest agreement with the other judges for $k = 5$, while for $k = 10$, judges 6, 12, and 13 have the lowest one. This proves the different levels of ambiguity and expressibility of their needs. More specifically, the different user contextual interpretations of the summaries produced, and the importance of them based on

[13] `https://code.google.com/archive/p/word2vec/`

their needs, are proven. There is though some common ground, which is shown in the agreement values.

Table 4.3: Distribution of agreement among judges

Judges	**Distribution of agreement**	
	k = 5	**k = 10**
judge 1	0.399	0.479
judge 2	**0.430**	0.479
judge 3	0.426	**0.504**
judge 4	0.417	0.491
judge 5	0.406	0.488
judge 6	0.353	0.416
judge 7	0.361	0.478
judge 8	0.391	0.469
judge 9	0.405	0.499
judge 10	0.387	0.459
judge 11	0.335	0.453
judge 12	0.385	0.439
judge 13	0.355	0.435
judge 14	0.413	0.483
judge 15	0.382	0.459

Quality

In Table 4.4 the quality is illustrated only for TakeAll distribution as all unique triples of each entity should be covered. Zipf distribution would select mostly popular triples without possibly covering all of them so it is not tested in this metric. Also, the time windows are not checked for the FACES-adapted approach as it was slow so no notification was extracted for this time window period.

Table 4.4 shows that the quality gets better with higher k, and it is analogous to the agreement for k = 5 and k = 10. This means that the overlap among the ideal summaries and the approach based ones follows the consensus among the ideal summaries that the judges gave. It is also observed that the FACES-adapted approach is the best one, followed by PubSum with PubSum GOOGLENEWS being the best one, followed by PubSum PHRASE, and PubSum VECTORS. Nevertheless, there is not a big difference between the quality values of the FACES-adapted and the PubSum GOOGLENEWS approach.

Table 4.4: Quality for top-k approaches

Distribution	**Approach**	**Quality**	
		k = 5	**k = 10**
Count tumbling window of size 30			
TakeAll	PubSum VECTORS	0.214	0.461
	PubSum PHRASE	0.217	0.447
	PubSum GOOGLENEWS	0.226	0.463
	FACES-adapted	**0.271**	**0.514**
Count tumbling window of size 50			
TakeAll	PubSum VECTORS	0.120	0.269
	PubSum PHRASE	0.127	0.282
	PubSum GOOGLENEWS	0.170	0.331
	FACES-adapted	**0.264**	**0.385**
Count sliding window of size 30 and slide 15			
TakeAll	PubSum VECTORS	0.233	0.465
	PubSum PHRASE	0.183	0.438
	PubSum GOOGLENEWS	0.263	0.482
	FACES-adapted	**0.290**	**0.541**
Count sliding window of size 50 and slide 25			
TakeAll	PubSum VECTORS	0.128	0.252
	PubSum PHRASE	0.137	0.288
	PubSum GOOGLENEWS	0.151	0.354
	FACES-adapted	**0.211**	**0.376**
Time tumbling window of size 0.5m			
TakeAll	PubSum VECTORS	0.096	0.278
	PubSum PHRASE	0.129	0.285
	PubSum GOOGLENEWS	**0.135**	**0.317**
	FACES-adapted	N/A	N/A
Time sliding window of size 0.5m and slide 0.25m			
TakeAll	PubSum VECTORS	**0.101**	**0.248**
	PubSum PHRASE	0.087	**0.248**
	PubSum GOOGLENEWS	0.081	0.235
	FACES-adapted	N/A	N/A

In terms of windows, the quality gets better for smaller windows as they contain fewer triples compared to bigger ones so the commonality between generated triples and ideal ones is less probable. It is also observed that sliding windows have better quality than tumbling ones, and time windows behave more poorly than count windows.

It should be noted that the quality metric of the PubSum approach is dependent on the choice of the two DBSCAN parameters ε and $minPts$. Although there was no difference observed among $\varepsilon = \{0.1, ..., 3\}$, the increase in $minPts$ contributed to less effective clustering. This makes sense as higher $minPts$ and lower ε result in highly dense clusters, which is a strict criterion. Therefore, in this chapter's work case, $\varepsilon = 1$ and $minPts = 1$ is selected as the goal is to relax the latter parameter so that triples can form large as well as very small conceptual clusters.

Redundancy-aware F-score

Redundancy-aware F-score in Table 4.5 is illustrated only for Zipf distribution as TakeAll distribution does not generate duplicate triples. It is observed that top-k filtering results not only in the elimination of duplicate redundant information but in possibly valuable information. Nevertheless, it is shown that the F-score ranges from 0.80 to 0.95. A lower F-score occurs for lower k as stricter content filtering is taking place. There is not much difference among the windowing policies, although it is seen that a higher F-score is observed with the increase in window sizes as the bigger the window, the more probable redundant information exists. The F-scores are very high, mostly because there is a lot of duplication in the generated triples due to Zipf distribution. The less duplication exists in the generated streams, the lower the F-score will be.

End-to-End Latency

In Table 4.6, the end-to-end latencies are illustrated. A similarity in the behaviour between count windows and time windows is observed. The slowest model is FACES-adapted, followed by PubSum GOOGLENEWS, PubSum PHRASE, PubSum VECTORS, and Non-top-k Fused approach. This shows that FACES-adapted spends much time pre-processing the triples and accessing the TDB triple store every time for ranking the triples. The PubSum GOOGLENEWS approach is the most expensive of the three embedding models as it has 3.5GB memory that needs, on average, 139,518ms to be loaded once in the system (included in latency). The other embedding models are much smaller (see Table 4.8) and need far less time with 16,382ms and 3,683ms needed for PubSum PHRASE and PubSum VECTORS, respectively. All these loading times are included in the values of the table. The Non-top-k Fused approach has the best latency since no processing is involved

Table 4.5: Redundancy-aware F-score for top-k approaches

Distribution	Approach	Redundancy-aware F-score		
		k = 5	k = 10	k = 15
Count tumbling window of size 30				
Zipf	PubSum VECTORS	0.807	0.911	0.915
	PubSum PHRASE	**0.810**	**0.914**	0.917
	PubSum GOOGLENEWS	0.805	0.910	0.919
	FACES-adapted	0.806	0.902	**0.952**
Count tumbling window of size 50				
Zipf	PubSum VECTORS	0.844	0.902	0.954
	PubSum PHRASE	0.845	**0.903**	0.955
	PubSum GOOGLENEWS	**0.846**	**0.903**	**0.956**
	FACES-adapted	0.843	0.898	0.947
Count sliding window of size 30 and slide 15				
Zipf	PubSum VECTORS	0.798	0.901	0.920
	PubSum PHRASE	0.796	0.901	0.921
	PubSum GOOGLENEWS	0.805	**0.909**	0.922
	FACES-adapted	**0.808**	0.894	**0.944**
Count sliding window of size 50 and slide 25				
Zipf	PubSum VECTORS	**0.850**	**0.908**	**0.959**
	PubSum PHRASE	0.844	0.901	0.956
	PubSum GOOGLENEWS	0.842	0.900	0.952
	FACES-adapted	0.844	0.898	0.943
Time tumbling window of size 0.5m				
Zipf	PubSum VECTORS	**0.857**	**0.903**	**0.949**
	PubSum PHRASE	0.853	0.900	0.948
	PubSum GOOGLENEWS	0.847	0.895	0.943
	FACES-adapted	N/A	N/A	N/A
Time sliding window of size 0.5m and slide 0.25m				
Zipf	PubSum VECTORS	**0.854**	**0.902**	0.950
	PubSum PHRASE	0.852	0.901	**0.951**
	PubSum GOOGLENEWS	0.848	0.897	0.948
	FACES-adapted	N/A	N/A	N/A

when sending notifications, but it is not that much quicker compared to the smaller embedding models with PubSum VECTORS being very close to it.

Other observations include that the Zipf distribution results in lower latency compared to the TakeAll one. Also, the latency increases with the window size as although the fusion and top-k diversity are incremental within the window, the

Table 4.6: End-to-end latency in ms

Approach	**End-to-end latency**	
	TakeAll	**Zipf**
Count tumbling window of size 30		
Non-top-k Fused	**27582**	**28442**
PubSum VECTORS	29237	30991
PubSum PHRASE	41984	42467
PubSum GOOGLENEWS	167000	166781
FACES-adapted	1180982	1243604
Count tumbling window of size 50		
Non-top-k Fused	**38179**	**35546**
PubSum VECTORS	40112	40641
PubSum PHRASE	55335	52655
PubSum GOOGLENEWS	177003	177369
FACES-adapted	1342192	1172200
Count sliding window of size 30 and slide 15		
Non-top-k Fused	**27772**	**26541**
PubSum VECTORS	31068	31839
PubSum PHRASE	43438	44245
PubSum GOOGLENEWS	167851	166693
FACES-adapted	1277535	1112800
Count sliding window of size 50 and slide 25		
Non-top-k Fused	**38666**	**37084**
PubSum VECTORS	43720	39853
PubSum PHRASE	53594	52414
PubSum GOOGLENEWS	176583	177406
FACES-adapted	1495831	1204236
Time tumbling window of size 0.5m		
Non-top-k Fused	**42153**	**41333**
PubSum VECTORS	50758	45907
PubSum PHRASE	60555	56187
PubSum GOOGLENEWS	187395	183301
FACES-adapted	N/A	N/A
Time sliding window of size 0.5m and slide 0.25m		
Non-top-k Fused	**43822**	**42261**
PubSum VECTORS	48762	46566
PubSum PHRASE	61302	59723
PubSum GOOGLENEWS	185180	183173
FACES-adapted	N/A	N/A

summary is sent after the window is populated; therefore, the population time is also considered. Latency is independent of k as the processing is done, and, then, only the top-k selection of the ranked triples takes place.

The reason the latencies seem large is that they are end-to-end, that is the summary that is created each time for an entity is the accumulation of the top entity information in the window that contains the times these facts were created. So the timestamp of whichever fact contributed to the summary, affects the summary's end-to-end latency.

Size Reduction

In Table 4.7, the number of forwarded messages for the TakeAll and Zipf distribution is illustrated. For the TakeAll, the number of forwarded messages is reduced within the ranges of 34% to 92% depending on the k and the window policy for the Top-k Fused approach (including PubSum and FACES-adapted) compared to the Non-top-k Fused one. For higher k values, more information is sent; therefore, the message reduction decreases. Smaller windows create more messages as they get populated more quickly with triples so more regular notifications are sent. Sliding windows create more messages compared to tumbling ones of the same window size as more frequent windows are created due to the slide so more notifications are sent. This is also the reason why smaller size windows with smaller slides produce more messages.

The Zipf distribution's results bear similar observations to those of TakeAll, although Zipf creates more messages in total as repetitive triples may be produced. In TakeAll, only the unique triples are generated by the sources. Therefore, for the time windows, TakeAll has a similar number of messages for the tumbling or sliding window as at some point the sources stop generating any more streams; therefore, more time will not have any effect. On the other hand, for Zipf, it is seen that the time sliding window produces more messages as again more frequent windows are created due to the slide so more notifications are sent.

TakeAll does not produce repetitive triples, but for Zipf, it is observed that from all messages, the Non-top-k Fused approach contains 56.7% to 69.3% duplicates, depending on the window policy. This percentage is particularly high in the case of Zipf, as popular triples will be produced more frequently than others. Smaller windows have less duplication, and sliding windows have lower duplication than their equivalent tumbling windows. The top-k approach can discard this duplicate

Table 4.7: Number of forwarded messages

Distribution	Approach	Number of forwarded messages			
		k = 5	**k = 10**	**k = 15**	**k = 20**
Count tumbling window of size 30					
TakeAll	Non-top-k Fused	1410	1410	1410	1410
	Top-k Fused	**235**	**470**	**705**	**940**
Zipf	Non-top-k Fused	4320	4320	4320	4320
	Top-k Fused	**720**	**1440**	**2160**	**2880**
Count tumbling window of size 50					
TakeAll	Non-top-k Fused	700	700	700	700
	Top-k Fused	**70**	**140**	**210**	**280**
Zipf	Non-top-k Fused	5000	5000	5000	5000
	Top-k Fused	**500**	**1000**	**1500**	**2000**
Count sliding window of size 30 and slide 15					
TakeAll	Non-top-k Fused	2040	2040	2040	2040
	Top-k Fused	**340**	**680**	**1020**	**1360**
Zipf	Non-top-k Fused	7500	7500	7500	7500
	Top-k Fused	**1250**	**2500**	**3750**	**5000**
Count sliding window of size 50 and slide 25					
TakeAll	Non-top-k Fused	850	850	850	850
	Top-k Fused	**85**	**170**	**255**	**340**
Zipf	Non-top-k Fused	7200	7200	7200	7200
	Top-k Fused	**720**	**1440**	**2160**	**2880**
Time tumbling window of size 0.5m					
TakeAll	Non-top-k Fused	364	364	364	364
	Top-k Fused	**30**	**60**	**90**	**120**
Zipf	Non-top-k Fused	2892	2892	2892	2892
	Top-k Fused	**240**	**480**	**720**	**960**
Time sliding window of size 0.5m and slide 0.25m					
TakeAll	Non-top-k Fused	360	360	360	360
	Top-k Fused	**30**	**60**	**90**	**120**
Zipf	Non-top-k Fused	5598	5598	5598	5598
	Top-k Fused	**480**	**960**	**1440**	**1920**

information; therefore, reducing the overall forwarded messages.

Memory Footprint

In Table 4.8, the memory footprint is illustrated. The memory does not refer to the execution of the approaches but to the models or ontologies that they are using. It is observed that FACES-adapted demands more memory compared to PubSum. This occurs because FACES-adapted uses the whole DBpedia in order to rank the available triples based on popularity and informativeness. For that reason, in this adapted implementation, a TDB Triple Store is used that demanded extra memory cost and several accesses for each summary extraction during execution. The memory cost indicated, in Table 4.8, only contains part of the whole DBpedia that contained the format of triples in the FACES dataset. The memory of storing all of the DBpedia would be higher. In terms of the PubSum approach, the PubSum GOOGLENEWS model is trained on a bigger corpus; therefore, it is the biggest one among the embedding models. This model is followed by PubSum PHRASE and PubSum VECTORS, which is the smallest model as it is trained on a small corpus. Non-top-k Fused approach did not demand any extra memory cost as it does not use any embedding models or ontologies.

Table 4.8: Memory footprint and throughput in triples/sec

Approach	**Memory footprint**	**Throughput**	
		TakeAll	**Zipf**
Non-top-k Fused	N/A	910	**1024**
PubSum VECTORS	**54.39 MB**	**914**	1005
PubSum PHRASE	519.81 MB	893	979
PubSum GOOGLENEWS	3.35 GB	833	890
FACES-adapted	11.8 GB	746	757

Throughput

In Table 4.8, the average throughput for all windowing policies for each approach is shown as no important differences were observed. It is seen that the TakeAll distribution has lower throughput than the Zipf, probably because in the latter much more events were generated so they entered the system more frequently. Non-top-k Fused approach along with PubSum VECTORS have the best throughput, followed by PubSum PHRASE, PubSum GOOGLENEWS, and, lastly, FACES-adapted. This behaviour is dependent or analogous to the memory consumption and end-to-end latency for each approach. As FACES-adapted is the most memory and latency costly approach, its throughput will be the lowest. In terms of windows, bigger ones have higher throughput by a few events, whereas sliding windows have lower throughput to tumbling ones of the same size by a few events.

Discussion

According to this chapter's results, it is concluded that Non-top-k Fused approach is the best one in relation to memory, throughput, and latency. Nevertheless, it sends all of the available information to the user that contains duplicate or conceptually redundant information. With the increase of sources, more and more data is generated with different variations of duplication or conceptual similarity; therefore, the Non-top-k Fused approach would perform worse. On the other hand, the Top-k Fused approaches reduce significantly the amount of data that is sent as a summary to the user. This means that the information sent to the user might not contain some non-redundant information due to filtering, but it manages to send a summary with quality analogous to the agreement among judges.

The worst model in terms of memory, throughput, and latency was FACES-adapted, although it performed slightly better than the rest of the approaches in terms of the quality of summaries. This shows that an existing thesaurus might be strict when it comes to synonyms or hypernyms, whereas a probabilistic model based on text-corpora, like word embedding models, covers a wider range of synonyms based on context. For example, semantically opposite words (antonyms), but conceptually similar (e.g. death place - birthplace) are taken into account as well as phrases. Also, this shows that finding all possible hypernyms and using a memory-heavy ontology in real-time for ranking can significantly decrease the performance of the system.

In conclusion, there is a trade-off between user effectiveness, data effectiveness, and system efficiency among a Non-top-k Fused and a Top-k Fused approach. This is because the latency, throughput, and memory are better in Non-top-k Fused approach, but the number of forwarded as well as redundant messages and data expressiveness are worse, and vice versa, for the Top-k Fused approach. It is also observed that a thesaurus/ontology-based Top-k Fused approach (like FACES-adapted) might be slightly better in terms of quality of summaries compared to an embeddings-based Top-k Fused approach (like the proposed PubSum), but it behaves worse in terms of system performance. It is concluded, then, that slightly more processing time, memory, and throughput for finding diverse data with the use of embeddings-based approaches, can lead to less data being sent upstream for further processing, as well as data that is seen as expressive as the agreement among judges without containing redundant information (duplicate or conceptual one).

4.6 Summary

Existing approaches have tried to resolve requirements regarding usability, user expressibility, data expressiveness, user effectiveness, data effectiveness, and efficiency. Nevertheless, no existing approach covers all requirements well. Diverse entity summarisation approaches create sophisticated conceptually and contextually diverse summaries of high usability, but they are static methodologies that cannot be directly applied to IoT environments or they may rely on thesauri and ontologies associated with aforementioned limitations. Pub/Sub approaches producing top-k diverse notifications only address duplication or commonality among attributes/terms in relation to redundancy and do not apply to rich entity-centric graph data. Pub/Sub approaches performing approximate semantic matching of medium usability do not address redundancy of any form and summarisation is out of scope. The same applies to Pub/Sub approaches semantically enriching events with the additional disadvantage that ontologies or domain experts are also used.

In this chapter, the PubSum approach is proposed, which is a novel dynamic diverse summarisation methodology for heterogeneous Linked Data entity graph streams. The approach consists of two phases: 1) conceptual clustering by a combination of word embedding models and DBSCAN clustering, and 2) geometric-based top-k ranking of triples by a combination of similarity metrics and some pre-defined rules. The approach creates graph-based diverse extractive entity summaries (notifications) for representationally-decoupled and diversity-aware subscriptions. It is evaluated for a range of windowing policies along with baselines including typical Pub/Sub and FACES, a thesaurus/ontology-based diverse entity summarisation approach. The results include promising but worse summary quality compared to FACES, redundancy-aware F-score of up to 0.95, up to 69.3% duplication reduction, 6 times less latency and 3 times less memory compared to FACES (depending on the size of the embedding model), up to 92% message reduction, and throughput ranging from 833 to 1,005 events/second.

Chapter 5

Dynamic Abstractive Summarisation of Numerical Data

5.1 Introduction

This chapter is dedicated to the IoTSAX approach, which has been published in Pavlopoulou and Curry [71] (early access).

The IoTSAX approach is a novel dynamic abstractive summarisation methodology that is based on symbolic approximation and approximate rule-based reasoning. It involves datasets of numerical real-time sensor data or time series that have heterogeneous attributes and contain redundant or extreme (sharply fluctuated) values. Therefore, it addresses both motivational scenarios regarding Healthcare and Smart Cities described in Chapter 2; nevertheless, additional information coming from static sources is out of scope. For example, the abstractive summarisation covers health sensors (e.g. pulse, heart beat rate, etc.) or property sensors (e.g. temperature, humidity, etc.), depending on the use case. The approach applies to domain-agnostic environments that demand scalability and timeliness, are of limited resources, and have no standardisation in data representations (i.e. representationally-coupled). Although its main application is abstractive entity summarisation, the approach can also be used for approximation and dimensionality reduction of time series or voluminous numerical sensor data, detection of patterns or extreme values or (sharp) fluctuations of time series or numerical sensor data, and reasoning in environments that have no standardisation in data representations.

Several challenges analysed in Chapter 2 are addressed with solutions described in Chapter 3. Specifically, the approach tackles data challenges regarding heterogeneity by providing a representationally-decoupled approximate rule-based reasoning methodology and subscription model, and redundancy caused by duplication by providing an abstractive summary that does not overwhelm the subscribers and the processing system. An emphasis is given on overcoming user challenges. Specifically, high-level interpretation of the numerical sensor data is targeted by proposing a sophisticated symbolic approximation approach along with reasoning to provide meaningful inferences to the users, user-friendly data representation is provided by the use of quad-based notifications, and non-technical users are facilitated by demanding only abstractive-aware subscriptions that do not include a-priori data, semantic or schematic information, or return partial or abstract user information. Finally, all system challenges regarding scalability, timeliness and resource constraints are addressed since a Publish/Subscribe system (Pub/Sub) is proposed along with windowing policies and summarisation that can cover dynamic sources, data continuity, real-time processing requirements, and possible network overhead. It should be noted that conceptually and contextually diverse notifications, conceptual similarity redundancy, and data enrichment are not covered in this chapter as they are out of scope. An emphasis (in grey) on which components are analysed in this chapter from the overall proposed *Entity-centric Publish/Subscribe Summarisation System* along with their associated research questions is illustrated in Fig. 5.1.

The contributions of this chapter are the following:

- A user-friendly entity-centric subscription model that allows subscribers to express in a simple way whether they need to receive an abstractive entity summary by also providing the desired window traits (type and size).
- IoTSAX, a novel dynamic abstractive summarisation methodology for heterogeneous numerical entity graph streams that is based on: 1) an enhanced SAX [187] by providing dynamic segment points as well as alphabet size for symbolic representations that follow data fluctuations, and 2) a novel approximate rule-based reasoning that is based on data approximation interpretation and embedding models that create quad-like entity-based abstractive summary notifications.
- A novel evaluation methodology including:

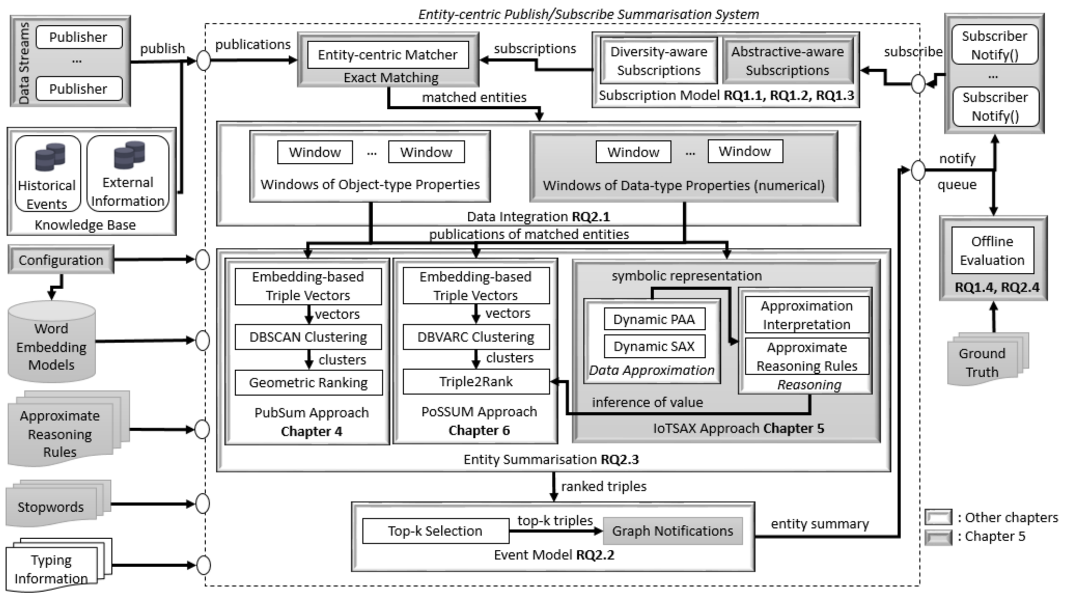


Figure 5.1: The components of the Entity-centric Publish/Subscribe Summarisation System analysed in this chapter.

  – The construction of an evaluation dataset based on real-world sensor data related to the domains of Healthcare and Smart Cities with characteristics involving heterogeneity in data types, semantics, concepts, and contexts.
  – The construction of a relevance ground truth based on semantically extending original datasets with the use of thesauri and ontologies.
  – Identification of SAX and typical Pub/Sub as baselines and adaptation of SAX to the proposed abstractive Pub/Sub summarisation system.
  – Identification of a range of evaluation metrics related to F-score, approximation error, latency, size reduction, compression space-saving, and throughput.

• An extensive evaluation comparison between IoTSAX approach and baselines by examining data effectiveness and system efficiency. The results include:

  – Data effectiveness: 2 to 3 times better in approximation error, and F-score of up to 0.87 for approximate rule-based reasoning.
  – Efficiency: 2 to 3 times slower in end-to-end latency ranging from 37,395ms to 69,414ms, similar throughput ranging from 13.231 to 97.393 events/second, up to 98% message reduction, and better compression space-

saving percentage when data redundancy occurs ranging from 71.75% to 94.99%.

The rest of the chapter is as follows: In Section 5.2 the necessary background is given related to Mathematics and approximation in time series, whereas, Section 5.3 contains related work and how it maps to the challenges, requirements, and research questions of the thesis. Section 5.4 describes the proposed Abstractive Publish/Subscribe Summarisation System, where its architecture and details of the IoTSAX approach are provided. Section 5.5 analyses the evaluation methodology as well as the results of the experiments, while Section 5.6 concludes and summarises the chapter.

5.2 Background

This section contains the necessary background for the rest of the chapter. Mathematical terminologies from Statistics are explained as well as the aim of approximation in time series and popular approaches along with their limitations.

5.2.1 Statistics

Terminologies from statistics are used in the rest of the chapter and are analysed below:

Z-normalisation: transforms all elements of a vector into a normalised vector, whose mean (average value) is approximately 0 and the standard deviation (variation of a set of values) is in a range close to 1.

Probability Density Function (PDF): defines a probability distribution, that is the likelihood of a random variable. The area under the curve of a probability distribution indicates the interval to which a variable belongs. The total area in this interval of the graph equals the probability of a variable occurring. Histograms are simple ways in which probability distributions can be illustrated as all observations are grouped into bins based on their number of occurrences.

Quantile Function: specifies the value on which the probability of a random variable equals to a given probability, if it is less than or equal to that value.

Standard Normal Distribution: is a normal distribution with a mean of 0 and a standard deviation of 1.

Skewness: depicts the asymmetry of a probability distribution of a random variable from its mean. A distribution can be positively skewed if the mode (most frequent value) is left to the mean (that means there is a high concentration of values on the left-hand side of a distribution), negative skewed if the mode is right to the mean, or symmetrical if the mode and mean are the same.

Non-parametric Kernel Density Estimation (KDE): is an algorithm that approximates a probability distribution since it does not match a well-known one (e.g. Gaussian). It uses a bandwidth and kernel function that control the window of observations from a data sample and the contribution of samples, respectively, towards estimating the probability of a new point.

5.2.2 Time Series Approximation

Time series approximation has been widely used for reducing the dimensionality of the original data. The aim is to create a succinct representation that has a small approximation error compared to the original data for resource-constraint environments with timeliness requirements.

Popular Time Series Approximation Approaches

Among the abundance of the existing time series approximation approaches, a few are the most popular ones. Discrete Fourier Transformation (DFT) [188] transforms a signal from the time domain to the frequency one. The original signal is represented by the first few Fourier coefficients. Discrete Wavelet Transformation (DWT) [189] transforms a vector into a set of smooth values and wavelet coefficients that consist of the average and the differences of every other two values of the vector. The process is iterative by considering as a vector the wavelet coefficients of the previous step until only a single value remains. Principal Component Analysis (PCA) [190] transforms the original data representation to a new orthogonal base by calculating their covariance or their Singular Value Decomposition (SVD) [191]. Piecewise Aggregation Approximation (PAA) [192] transforms a time series vector into a reduced vector of a pre-defined segment length by taking the average of the original vector values within each segment. SAX [187] transforms a time series into a set of letters. Initially, normalisation and PAA are applied to the original vector and, then, the reduced vector is represented by letters based on breakpoints according to the Standard Normal Distribution.

Symbolic Representation and Symbolic Aggregate Approximation

As aforementioned, Symbolic Aggregate Approximation (SAX) [187] is a lossy compression methodology that transforms a time series into a set of letters/symbols. The aim of a symbolic representation of a time series can be formalised as follows:

A univariate time series $X = \langle x_1, x_2, ..., x_N \rangle \in \mathbb{R}^N$, that is a sequence of N data points measured at successive points in time, can be approximated into a symbolic representation $S = \langle s_1, s_2, ..., s_n \rangle \in \mathbb{A}^n$, where s_i is a letter of an alphabet $\mathbb{A} = \{a_1, a_2, ..., a_k\}$ of k letters. The approximation should:

- boost the system performance; therefore, $n \ll N$ and $k \ll n$, meaning that the approximate sequence should have a much smaller dimension than the original one and that only a small number of meaningful letters should be used.
- satisfy the lower bounding principle, that is a distance measurement can be found on the reduced/symbolic vector that is guaranteed to be less than or equal to the true distance measured on the original vector.
- provide automatically tuned parameters n and k.

SAX consists of three steps; z-normalisation, PAA, and discretisation. The z-normalisation ensures that all normalised values of a time series have mean and standard deviation close to 0 and 1, respectively. PAA divides the normalised time series into user-defined equal segments and calculates the mean value of the data in each segment, which constitutes a PAA coefficient. The last step is symbolising each PAA coefficient to a letter. This step is based on the assumption that normalised time series follow a Standard Normal Distribution; therefore, according to a user-defined alphabet size, equiprobable regions are defined by breakpoints based on the Standard Normal Distribution. Each PAA coefficient falls within a specific region between breakpoints that defines the letter with which it will be symbolised. An example of SAX approximation on the first 100 respirationRate (RESP) time series samples for $n = 5$ and $k = 5$ is illustrated in Fig. 5.2. In this case, the whole time series is approximated by the symbolic representation "bbcdd".

The Power of Symbolic Aggregate Approximation and its Limitations

There have been several studies that have shown the superiority of SAX and PAA, which SAX is based on, compared to other approximation techniques [139, 193–195]. Specifically, PAA and SAX are simple and computationally much less costly

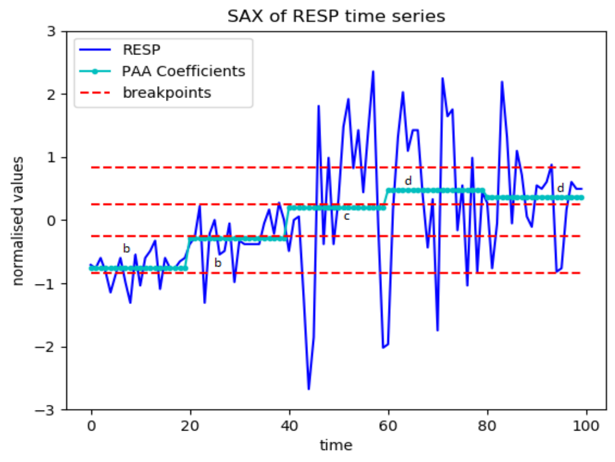
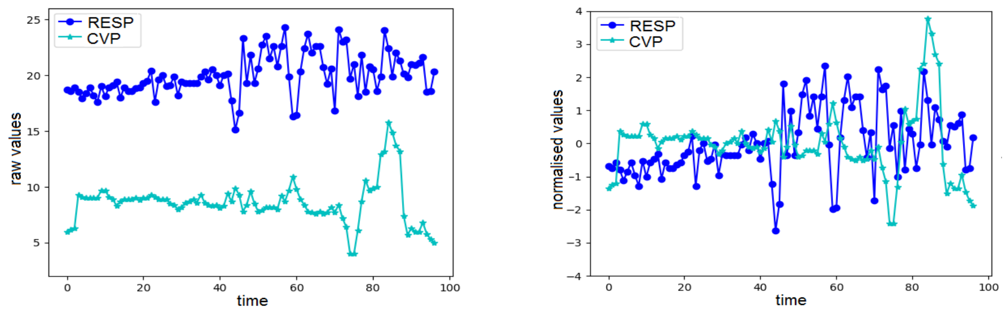


Figure 5.2: An example of SAX approximation.

without requiring complex matrix computations, such as in PCA and SVD. They have no limitation on the time series length, whereas in the case of DFT and DWT the length should be of the power of two. Even though their parameters can affect their performance, the loss of the original data's nature and quality may be more evident in the cases of DFT, DWT, SVD, and PCA when the best number of coefficients or principal components should be defined. They do not require a global calculation if the data is not normalised; therefore, they could be implemented incrementally or in batches for quicker processing time, unlike SVD and PCA. Like the other approaches, they also satisfy the lower bounding principle. They are the best at retaining the main characteristics of the original time series by also having high dimensionality reduction; therefore, there is a high correlation between the reduced/symbolic vector and the original one. SAX has also the added advantage that it can provide symbolic representations in order to observe patterns or occurrences of specific numerical ranges, unlike the rest of the approaches.

On the other hand, SAX is limited when it comes to real-time processing environments like Internet of Things (IoT). Specifically, it uses a fixed/non-adaptive segment number n and alphabet size k that is defined by the user. These parameters are data-dependent; therefore, they need to be manually tuned for their best values, which is not applicable in streaming data. Also, the fixed segment number may result in poor aggregation performances as in sensor data there might be a range of data activities within a segment, from low to high data generation and small to sharp fluctuations that could indicate extreme/abnormal values. All this original information will be lost in non-adaptive segments as the average value



(a) Difference between raw RESP and CVP time series (b) Difference between normalised RESP and CVP time series

Figure 5.3: Normalisation maintains the shape of the original time series but loses the amplitude.

will be defined through PAA.

Another limitation is the normalisation of the raw data since the authors suggest that this is a meaningful way of comparing time series. Nevertheless, in this chapter's work, reasoning and comparison among time series is not based only on their shape but on their amplitude, which is information that will be lost with normalisation. This is more clearly illustrated in Fig. 5.3, where the distance between the time series in Fig. 5.3b seems smaller compared to what it actually is as depicted in Fig. 5.3a, and the original values that would be used in reasoning are lost. Finally, the assumption that normalised data follows a Standard Normal Distribution may not be the case in IoT data.

5.3 Related Work

This section contains the related work, which is split into the following categories:

- **Time Series Approximation**: It involves approaches that transform time series into succinct representations.
- **Conceptual and Contextual Awareness**: It involves either semantic rule engines and languages or machine learning and approximation approaches that turn raw numerical IoT data into rich information.
- **Approximate Semantic Matching in Pub/Sub**: It involves Pub/Sub systems that introduce semantic decoupling in the system.

The related work is analysed and mapped according to the aforementioned requirements:

- **R1: Usability**: high usability system independent of representational coupling, query language expertise, system knowledge, bias, and background knowledge.
- **R2: User Expressibility**: users understanding their data needs and expressing them by simple subscriptions of high usability with minimal configuration settings.
- **R3: Data Expressiveness**: domain-agnostic system tackling interoperability and heterogeneity by providing rich notifications that contain conceptual and contextual diversity or high-level abstractions.
- **R4: User Effectiveness**: notifications of high quality according to the users' needs.
- **R5: Data Effectiveness**: redundancy-aware and expressive notifications of high quality according to the wide range of concepts and contexts.
- **R6: Efficiency**: efficient system in terms of memory, processing time, throughput, and scalability.

5.3.1 Time Series Approximation

Approximation is highly used in resource-constraint environments and it provides acceptably smaller and quicker notifications, provided the approximation error compared to the original data is low [41]. As aforementioned, there are several popular approaches; nevertheless, the focus of this chapter is on SAX. Due to the aforementioned SAX's advantages, it has been used in its original form or optimised one by several works. These works are either related to offline methodologies or ones that deal with sensor data. Specifically, the fact that SAX can be used to approximate massive and high-dimensional data, like in the case of IoT, made it possible to be applied in sensor-based research either as an approximation approach for optimisation purposes or for finding spatio-temporal correlations [29, 42]. The most notable offline and sensor-related works are analysed below.

Puschmann et al. [42]

Description: This work uses approximation and topic modelling to identify underlying structures and relations in sensor data. Initially, the approach uses SAX to turn raw sensor data to approximate symbolic representations that are analysed in higher-level abstractions through rules. Then, Latent Dirichlet Allocation (LDA) takes place, which is a popular topic modelling methodology that is used in text. Finally, the work emphasises on the challenges involved in extracting information from multiple heterogeneous sources for creating more enriched contextual information.

Review: The work effectively addresses challenges related to data enrichment and high-level interpretation as heterogeneous data is merged and abstracted in order to turn raw data into valuable meaning regarding data patterns and correlations between traffic flow and weather circumstances (R3). On the other hand, the solution uses LDA, which is a supervised methodology, that is it needs training data in order to be effective, and it requires manually-tuned hyperparameters (R5). Supervised solutions are not efficient in streaming environments as a prior collection of data to train a model and frequent updates of the training set according to changes (i.e. concept drift) will prove time-consuming (R6). Another issue is that although the approach could be extended to other domains, only one use-case is evaluated (R5), which is the correlation between the traffic flow and weather conditions, and several manual rules are created for high-level abstractions that are dependent on the meaning of the data in question. Finally, the quality of the extracted data abstractions is partially evaluated (R5).

Regarding SAX, the approach keeps the raw values of the data (no normalisation) and finds its true distribution based on KDE (R3). The distribution's equiprobable regions are examined to define the symbols to be used for the symbolic representation. Nevertheless, a fixed/non-adaptive user-defined segment number and alphabet size are used that relate to the aforementioned limitations (R3, R5, R6), PAA's original version is used without any optimisations, and no extreme values or data fluctuations are observed.

SensorSAX [43]

Description: This work uses approximation to create high-level abstractions of data in order to reduce data communication and original information loss. Initially, the approach enhances SAX regarding variable (non-fixed) segment lengths based on the streaming activity of data via its standard deviation. Then, an enhanced Parsimonious Covering Theory [196] is suggested to infer time-dependent sensor data along with the use of Hidden Markov Models [197] for detecting abstractions happening over time.

Review: The work effectively addresses the challenge of high-level interpretation as raw data turns into valuable meaning regarding high-level abstractions (R3). On the other hand, only one type of sensor data is evaluated (R5) that relates to coastal observations making the approach non-applicable to heterogeneous sensors (R3). Also, the Parsimonious Covering Theory approach is dependent on the use case (R3) and along with Hidden Markov Models the solution uses advanced machine learning techniques that are too complex for real-time systems (R6). Finally, the quality of the extracted data abstractions (R5) and the efficiency (R6) of the approach are partially evaluated along with the usability of a GUI that is not evaluated (R1).

Regarding SAX, the approach uses adaptive segments based on the streaming activity of the data (R6), that is a larger segment will be used if the activity is low or a smaller one if the activity is high for better reconstruction of the original data. Nevertheless, the adaptivity is dependent only on the standard deviation of the data and not on its variance (R3), and a threshold need to be manually-defined to observe if it exceeds an appropriate standard deviation value for a segment to be allocated (R5). Apart from that, a fixed/non-adaptive user-defined alphabet size is used, the raw data is normalised, the data is assumed to follow a Standard Normal Distribution, SAX's original version is used without any optimisations (only PAA is optimised), a fixed SAX word length is used, and no extreme values or data fluctuations are observed, with all relating to aforementioned limitations (R3, R5, R6).

SAX-ARM [44]

Description: This work uses approximation and discovers rules to describe deviant event patterns from multivariate time series. Initially, the approach normalises the raw data and turns it into approximate symbolic representations via SAX. Then, association rule mining is used for discovering frequent rules among the symbols of deviant events.

Review: The work effectively addresses the challenge of high-level interpretation as raw data turns into valuable meaning regarding relations and simultaneous frequent patterns among deviant events (R3). It is the only related work that addresses extreme values (R3, R5). On the other hand, the approach applies to static data; therefore, it is not efficient for IoT environments (R6). Another issue is that association rules are simple and efficient methodologies, but they are dependent on the use case (R3). Finally, the effectiveness of the solution is poorly evaluated as only one synthetic dataset is examined with limited variables, few association rules are evaluated for their quality, and the approximation error is not addressed (R5).

Regarding SAX, the approach normalises the time series by applying the inverse normal transformation to ensure it follows the normal distribution, the alphabet size is dependent on the deviant events percentage (e.g. if top 10% and bottom 10% need to be monitored, then, a = 10), and multivariate time series are addressed rather than univariate ones (R3). Nevertheless, the alphabet size is still not completely dynamic and only the first and last alphabets are considered as only extremes are important in the analysis, a fixed/non-adaptive user-defined segment number is used, the raw data is normalised, PAA's and SAX's original versions are used without any optimisations, and no data fluctuations are observed, with all relating to aforementioned limitations (R3, R5, R6).

Zan et al. [45]

Description: This work enhances all of the parameters of SAX. The dynamic segments are defined by recursive calculations and the fitting of linear regression to the data of each segment according to a threshold, whereas the dynamic alphabet size for the symbolic representation is based on the skewness of the approximate data distribution.

Review: It is the only related work that enhances all of the parameters of SAX, that is the segment number and the alphabet size are both dynamic (R3, R5). It is also thoroughly evaluated with a large collection of datasets (R5). On the other hand, the approach applies to static data; therefore, it is not efficient for IoT environments (R6). Also, it is computationally expensive (R6) as many iterations of calculations take place, that is a first version of dynamic segment numbers is defined by sampling frequency, then, PAA occurs that is followed by final dynamic segment numbers by further segmenting each previously-defined segment based on linear regression. Even the linear regression is recursive based on how well it fits the original data. Another issue is that a threshold needs to be manually-defined to observe the fitness of the linear regression (R5). Regarding SAX, the raw data is normalised, the data is assumed to follow a Standard Normal Distribution, the breakpoints are extracted as in the SAX's original version, and no extreme values or data fluctuations are observed, with all relating to the aforementioned limitations (R3, R5, R6).

Other Relevant Approaches

Approaches have addressed the problem of stream approximation for streams containing numerical data (R3, R6). Methodologies include synopsis methods (i.e. sampling, wavelets, sketches, and histograms), frequent patterns (i.e. frequent items and frequent itemsets), and clustering, analysed in the surveys of Aggarwal et al. [48] and Gupta et al. [49]. These methodologies could also apply to time series and some have also been used in wireless sensor networks or Pub/Sub systems [198]. This type of work is not directly linked to the proposed work since it refers to other types of approximation. Also, all of these methodologies have their own disadvantages regarding the scope of the proposed work. Specifically, all of them apart from histograms are costly in time and space (R6) making them not so suitable for streams [48, 92], sampling and sketches cannot be used for the monitoring of extreme values or fluctuating data rates [40, 48, 91] (R3, R5), the true distribution of the data is lost in sketches and histograms [48] (R3, R5), and frequent patterns and clustering address another problem.

5.3.2 Conceptual and Contextual Awareness

There are a number of works that have tackled conceptual and contextual awareness from raw numerical IoT data. The main approaches are supervised or unsupervised learning models (e.g. neural networks, clustering), manual rules, fuzzy rules, ontology-based, and probabilistic models (e.g. Markov models) in order to transform low-level context to high-level one [61, 199]. Nevertheless, there is no consensus on which approach is better as each of them has its own advantages and disadvantages. This chapter's work emphasises on rule-based reasoning since it is a simple, unsupervised, time-efficient, and low-memory way to infer high-level abstractions of sensor data. The most related sub-categories of contextual-awareness approaches to the proposed work are semantic rule engines and languages as well as machine learning and approximation that are analysed below.

Semantic Rule Engines and Languages

Description: This type of work involves systems or languages that contain rules with an emphasis on semantic reasoning and interoperability. Approaches include SRE [62] that uses Lua to create semantic rules for sensor decoupling when a topology change occurs (e.g. sensor being replaced or deleted), LOV4IoT [63] that is a Machine-to-Machine Measurement framework that uses datasets and sensor-based linked open rules by sharing and reusing existing ontologies/datasets/rules by domain experts, and C-SWRL [64] that is a rule language that extends SWRL to reasoning over stream data.

Review: This type of work could address challenges related to heterogeneity and high-level interpretation as sensor data is annotated with contextual information based on ontologies (e.g. W3C Semantic Sensor Networks (SSN) ontology) and time-aware reasoning occurs (R3). On the other hand, ontologies are linked to the aforementioned limitations, in Chapter 2, regarding complexity and inefficiency, representational coupling, domain dependency and domain-experts, which could significantly affect the effectiveness and efficiency of the approach (R5, R6). These limitations contribute to the constructed reasoning rules to be domain-specific and representationally-coupled as not all applications or representations could be available in the ontologies. A high volume of rules is needed in order to cover most cases resulting in less efficiency and heavy memory when stored in a repository (R6). Also, maintaining and updating these computationally expensive rules and ontologies is challenging. Another issue is that the approaches are poorly evaluated (only efficiency is evaluated and not effectiveness) with some addressing only one use-case or only a few number of rules (R5). These approaches also mostly refer to data stream processing models that do not contain the advanced scalable and temporal capabilities of Pub/Sub systems (R6). They also extend or use as basis languages that suffer from the limitations of data stream processing languages regarding streams processed in isolation, no standardisation in syntaxes, semantics and expressiveness, and complex representationally-coupled SPARQL-like queries. Finally, usability (R1), user expressibility (R2), user effectiveness (R4), redundancy, and summarisation (R3) are out of scope.

Machine Learning and Approximation

Description: This type of work involves approaches that use machine learning and/or approximation along with rule-based reasoning in order to infer high-level abstractions of data. Notable approaches include Ganz et al. [65], Wei et al. [66], and aforementioned Puschmann et al. [42]. Ganz et al. [65] use a combination of SensorSAX approximation, an extended k-means clustering, a Markov model, and an SWRL rule-based system to create topical ontologies (by extending the SSN ontology)), which is domain knowledge by providing a basic conceptual understanding. K-means is used for grouping data into clusters to form unlabelled concepts (SAX patterns), whereas the Markov model is used to create temporal relations among the concepts. Labelling occurs by rule-based reasoning. Wei et al. [66] integrate sensor data with Semantic Web and Linked Data for reasoning and annotation. The reasoning is addressed by rules based on ontologies that create the Semantic Sensor Web ontology, and rule approximation is supported in the sense of spatial proximity.

Review: This type of work could address challenges related to heterogeneity and high-level interpretation as time-aware reasoning occurs based on rules deriving from the Web or ontologies (R3). On the other hand, these approaches contain the same limitations as the ones described in the semantic rule engines and languages category regarding ontologies, domain-specific, representationally-coupled reasoning rules, volume of rules, and SPARQL-like queries (R5, R6). Also, these approaches are complex and inefficient for IoT environments as either too many machine learning techniques are used or access is needed to multiple Web resources (R6). Another issue is that these machine learning techniques as well as the approximation techniques have their associated parameters that need to be manually-tuned for optimised results or they may be supervised (R5). Also, some approaches are either poorly evaluated with only one use-case or not evaluated at all (R5, R6). Finally, usability (R1), user expressibility (R2), user effectiveness (R4), redundancy, and summarisation (R3) are out of scope.

Other Approaches

Some other approaches have addressed conceptual and contextual awareness from raw numerical IoT data (R3). All of these approaches contain the aforementioned limitations (R1-R6). Some works include approaches that use machine learning or edge/fog technologies for Healthcare [77] as well as frameworks for Smart Cities. Specifically, Waternomics [84] installed smart meters in Italy (Linate airport), Greece (individual households in Thermi) and Ireland (NUI Galway and Coláiste na Coiribe) to observe water consumption and water availability that could lead to possible leak detection, end-user awareness, and decision quality. CityPulse [29, 85] is a distributed framework with the aim to perform semantic discovery, data analytics, and interpretation (reasoning) of IoT data and social media data streams integrated from multiple sources. SmartSantander [86] is a centralised platform that accesses heterogeneous data generated in several European cities by focusing on the management of low-level sensor streams. OpenIoT [87] is a middleware that enables semantic interoperability of IoT in the cloud via the use of ontologies like SSN. Spitfire [74] provides a uniform way to search, interpret, and transform sensor data by using semantic technologies. PLAY [88] is a scalable cloud-based event-based system that integrates on-the-fly both real-time and historical data, and processes complex events deriving from large-scale, distributed, and heterogeneous environments. Several other frameworks are discussed and evaluated at Fahmideh et al. [200].

5.3.3 Approximate Semantic Matching in Pub/Sub

This category has been analysed in Chapter 4. It is linked to the proposed work from the perspective of approximate subscriptions (R2) as redundancy and summarisation are out of scope (R3). It contains approaches that: 1) although they contain semantically-decoupled subscriptions, they are still in the form of attribute-value pairs that have medium usability (R1), 2) some rely on taxonomies, thesauri or ontologies with a plethora of limitations (R1-R6), and 3) some contain manually-tuned parameters (R4).

5.3.4 Comparison

In conclusion, no existing work covers all of the requirements well. Comparison among the different categories in relation to the features and the requirements is shown in Table 5.1 and Table 5.2, respectively.

Table 5.1: Features as addressed by related work

Approach	**Stream-ing/IoT**	**Raw Values**	**True Distribution**	**Dynamic Segments**	**Dynamic Alphabet Size**	**Extreme Values**	**(Sharp) Fluctuations**	**Unsupervised Learning**	**Domain Agnostic**
Time Series Approximation									
Puschmann et al. [42]	✓	✓	✓						dependent on rules
SensorSAX [43]	✓			✓				✓	
SAX-ARM [44]			normal distribution assumption		dependent on % of extreme values	✓		✓	dependent on association rules
Zan et al. [45]				✓	✓			✓	✓
IoTSAX	✓	✓	✓	✓	✓	✓	✓	✓	✓

Approach	**Query Type**	**Semantically Decoupled Reasoning Rules**	**Approximate Reasoning Rules**	**Model Dependency**	**Unsupervised Learning**	**Domain Agnostic**
Semantic Rule Engines and Languages						
[62–64]	N/A			ontologies		
Machine Learning and Approximation						
Ganz et al. [65]	N/A			ontologies	✓	
Wei et al. [66]	N/A		spatial	ontologies	✓	
Puschmann et al. [42]	N/A					✓
Approximate Semantic Matching in Pub/Sub						
[39, 57, 58, 162–164]	semantic decoupled or fuzzy subscriptions	N/A	N/A	distributional semantics (most) and taxonomies or thesauri and ontologies	✓	dependent on approach
IoTSAX	representational decoupled subscriptions	✓	semantic	embedding models	✓	✓

Table 5.2: Requirements as addressed by related work

Approach	**R1 - Usability**	**R2 - User Expressibility**	**R3 - Data Expressiveness**	**R4 - User Effectiveness**	**R5 - Data Effectiveness**	**R6 - Efficiency**
Time Series Approximation						
Puschmann et al. [42]	N/A	N/A	++	N/A	+	-+
SensorSAX [43]	N/E	N/A	+	N/A	+	+
SAX-ARM [44]	N/A	N/A	++	N/A	-	N/A
Zan et al. [45]	N/A	N/A	++	N/A	++	N/A
Semantic Rule Engines and Languages						
[62–64]	N/A	N/A	+	N/A	N/E	-+
Machine Learning and Approximation						
Ganz et al. [65]	N/A	N/A	+	N/A	+	N/E
Wei et al. [66]	N/A	N/A	+	N/A	N/E	N/E
Approximate Semantic Matching in Pub/Sub						
Hasan et al. [57]	+	-+	N/A	+	N/A	++
Hasan et al. [39]	+	+	N/A	++	N/A	++
Alhakbani et al. [58]	+	+	N/A	++	N/A	++

++ the requirement dimension is well covered
+ the requirement dimension is partially covered with positive results
−+ there is an attempt to address the requirement dimension but the solution is not effective
− the requirement dimension is poorly covered
−− the requirement dimension is very poorly covered
N/A the requirement dimension is not addressed or the focus of the research
N/E the requirement dimension is not evaluated

5.4 Abstractive Publish/Subscribe Summarisation System

An extension of Pub/Sub is proposed, in this chapter, that can overcome all aforementioned issues, limitations, and gaps; the Abstractive Publish/Subscribe Summarisation System that is based on data approximation and reasoning (IoTSAX approach) that create quad-based entity summaries (notifications) for abstractive-aware subscriptions. In this section, an overview of the architecture of the proposed system as well as details of the IoTSAX approach are described.

5.4.1 Architecture

The architecture of the approach is illustrated in Fig. 5.4. Publishers generate publications concerning the entity Patient1 and one's measurements regarding respiration rate and heart rate, among others. A subscriber generates an *Abstractive-aware Subscription*, where one is interested in an abstractive summary of the entity Patient1 deriving from the analysis of data taken from count tumbling windows of size 100 (only count tumbling windows are supported for this implementation). All publications and subscriptions enter the *Abstractive Publish/Subscribe Summarisation System*, where they are stored and managed. In the example, the triples are the following: $t_{rr}1$: <Patient1> <respirationRate> "18.700", $t_{rr}2$: <Patient1> <respirationRate> "18.600", $t_{hr}1$: <Patient1> <heartRate> "96.400", $t_{hr}2$: <Patient1> <heartRate> "95.800". The available triple information is presented without any schematic prefixes for visualisation purposes.

The stored publications and subscriptions are examined by the *Entity-centric Matcher*. If there is a match between the entity requested by the subscribers and the entity to which publications refer, like in this example Patient1, then, a match occurs. After a match, two kinds of listeners are connected to the system, one for the publishers and one for the subscribers in order to observe whenever new elements have been generated related to the entity for future processing. The publications of Patient1 enter windows as they are generated with each window related to a different measurement, that is one window for the respiration rate and one for the heart rate, among others (*Data Integration*). As elements get in the window a trigger is activated and they are incrementally processed. First, a counter is calculating all elements currently in the window, then, the object or

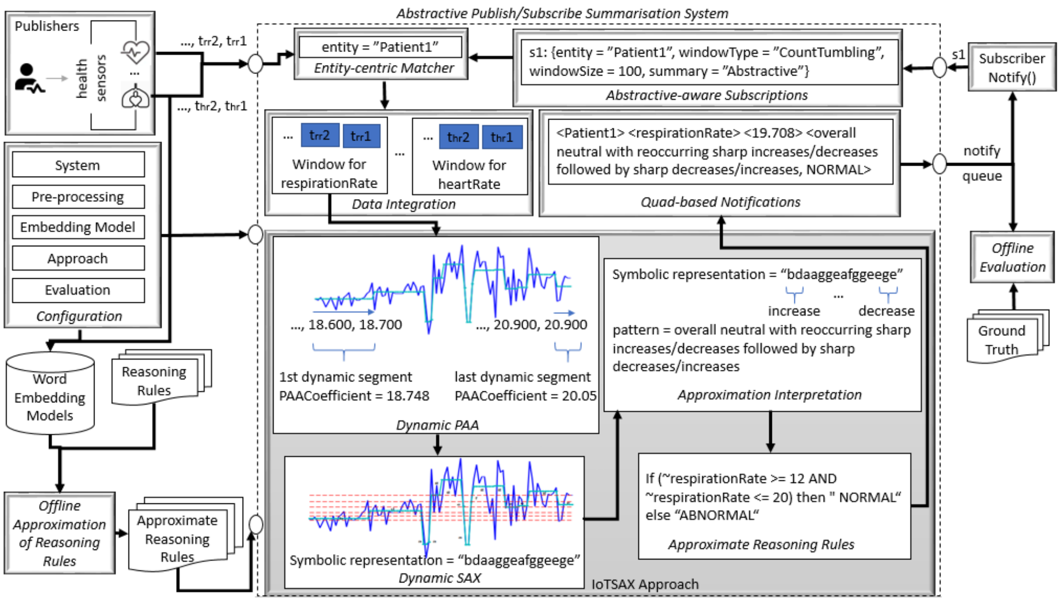

Abstractive Publish/Subscribe Summarisation System
Publishers
health sensors
..., trr2, trr1
..., thr2, thr1
entity = "Patient1"
Entity-centric Matcher
s1: {entity = "Patient1", windowType = "CountTumbling", windowSize = 100, summary = "Abstractive"}
Abstractive-aware Subscriptions
s1
Subscriber
Notify()
trr2 trr1
Window for respirationRate
thr2 thr1
Window for heartRate
Data Integration
<Patient1> <respirationRate> <19.708> <overall neutral with reoccurring sharp increases/decreases followed by sharp decreases/increases, NORMAL>
Quad-based Notifications
notify
queue
Offline Evaluation
Ground Truth
System
Pre-processing
Embedding Model
Approach
Evaluation
Configuration
..., 18.600, 18.700
..., 20.900, 20.900
1st dynamic segment
PAACoefficient = 18.748
last dynamic segment
PAACoefficient = 20.05
Dynamic PAA
Symbolic representation = "bdaaggeafggeege"
increase
decrease
pattern = overall neutral with reoccurring sharp increases/decreases followed by sharp decreases/increases
Approximation Interpretation
Word Embedding Models
Reasoning Rules
Offline Approximation of Reasoning Rules
Approximate Reasoning Rules
Symbolic representation = "bdaaggeafggeege"
Dynamic SAX
If (~respirationRate >= 12 AND ~respirationRate <= 20) then " NORMAL" else "ABNORMAL"
Approximate Reasoning Rules
IoTSAX Approach

Figure 5.4: Architecture of the Abstractive Publish/Subscribe Summarisation System.

value of the triple (publication's payload) is extracted, and *Dynamic PAA* takes place. The Dynamic PAA is checking incrementally the values that enter the window to find dynamic segments and to calculate the segment's PAA coefficient. In the example, the first and the last segments are depicted for the first 100 respiration rate values, along with their corresponding PAA coefficients. This incremental process ends once the window reaches its full capacity, that is the counter is equal to the subscription's windowSize, which is 100. Afterwards, *Dynamic SAX* takes place where according to the probability distribution of the samples and the calculation of a dynamic alphabet size via the skewness of the distribution, breakpoints are extracted that along with the pre-defined PAA coefficients approximate all 100 samples into the symbolic representation "bdaaggeafggeege". Once a concise symbolic representation has been generated, the window is evicted (cleared) and awaits the new values. The notification process is, then, triggered that checks the state of all subscriptions in order to notify all interested parties. The notification process is responsible for the creation of quads. A quad consists of four-part information including the original triple's subject (entity) and predicate (measurement) as well as the mean value of all the PAA coefficients of the respiration rate window (in this example) as an object, and contextual information. The contextual information is generated based on two components; *Approximation Interpretation* and *Approximate Reasoning Rules*.

The Approximation Interpretation is responsible for extracting a generalised behavioural pattern of the 100 respiration rate values. This is done by observing the distance between subsequent symbols of the symbolic representation "bdaaggeafggeege" and translating it to increasing or decreasing patterns. In the example, the first and the last distances are depicted. Then, according to the occurrence of the patterns as well as their sequence, a final pattern is extracted. The Approximate Reasoning Rules contains a semantically related rule to the attribute in question (respiration rate in the example) deriving from the *Offline Approximation of Reasoning Rules*. In this case, there is an exact match; therefore, the precise reasoning rule is selected. The rule's aim is to extract a high-level inference on what the actual average value of the attribute/predicate really means, resulting in the fact that the respiration rate is within a normal value range, in this window. The abstractive summarisation process is, then, completed and the newly constructed timestamped *Quad-based Notification* is sent to the subscriber and to a queue that along with necessary metadata will be stored externally in order to be evaluated offline *Offline Evaluation*. This whole process ends once a pre-defined time duration has been reached. Once the run is completed, the Offline Evaluation is responsible for calculating all the necessary metrics.

The *Offline Approximation of Reasoning Rules* process takes place externally (Fig. 5.5). For this process, a pre-selected word embedding model is used along with pre-defined reasoning rules and necessary pre-processing models. All attributes found in the publications as well as the ones found in the reasoning rules undergo *Pre-processing* and are stored in an index (Processed Words Index) so that they are not pre-processed again in the future. Then, the word embedding model turns the pre-processed attribute into a vector that is also stored in an index for future use (Word2Vec Index). The next step is to perform *Scoring* based on which reasoning rules' attribute is closest to the attribute in question and, finally, to extract the *Top-1*, resulting in the one that is the most semantically related one. Therefore, each new attribute is mapped to an existing reasoning rule. The mappings are stored in the Approximate Reasoning Rules that are used in the online reasoning process of the IoTSAX Approach.

The last component to be analysed is the *Configuration* that is responsible for holding static values of parameters that concern a wide range of components. Specifically, it contains parameters regarding the system (e.g. duration of streaming, windowing policies, and publisher and subscriber policies), pre-processing

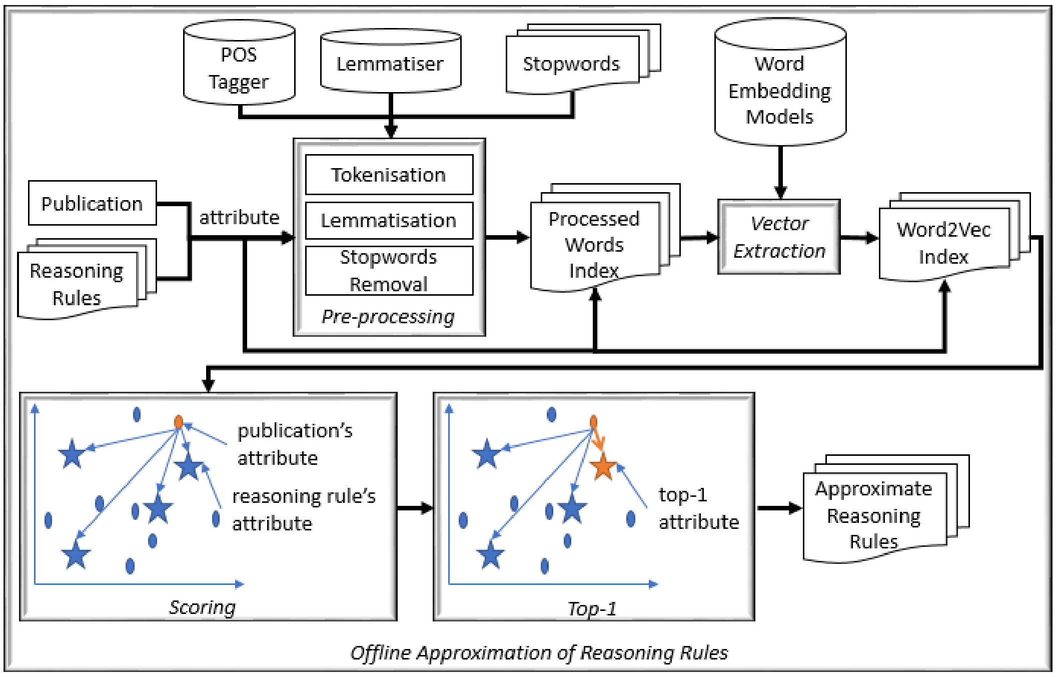


Figure 5.5: Architecture of the Offline Approximation of Reasoning Rules.

(e.g. stopwords removal, Part of Speech (POS) tagging, and lemmatisation), word embedding model (e.g. type of model and corpus of model), approach (e.g. type of algorithm/approach like IoTSAX or baselines, IoTSAX's parameters like threshold and max/min alphabet size, and baselines' parameters like the number of segments, alphabet size, and alphabet), and evaluation (e.g. storage of results/notifications, ground truth locations, and metrics to be calculated).

5.4.2 IoTSAX Approach

The main abstractive summarisation approach (IoTSAX approach) consists of two phases: 1) IoTSAX approximation, and 2) approximate rule-based reasoning.

IoTSAX Approximation

To avail of the advantages of SAX and overcome its disadvantages, IoTSAX is proposed, which is a dynamic SAX approximation that could be used in IoT environments. IoTSAX defines, incrementally, the best segments and alphabet size within a data window. In this way, the dynamic segments observe and retain the sharp fluctuations in the data due to extreme/abnormal values, whereas the dynamic alphabet size leads to more fine-grain representations of the approximate data based on its original skewness. Also, no data normalisation takes place so that its amplitude is not lost, and the true PDF of the data is found based on the non-parametric KDE [201]. An example of IoTSAX approximation on the first 100 RESP time series for thr = 4, kMax = 10, and kMin = 3, is illustrated in Fig. 5.6. The segments follow the sharp fluctuations of the data, compared to Fig. 5.2, and the skewness-based alphabet size $k = 7$ leads to a fine grain representation of the approximate data, which in this case it is symbolically represented by "bdaaggeafggeege". More details are given in Algorithm 4 and Algorithm 5.

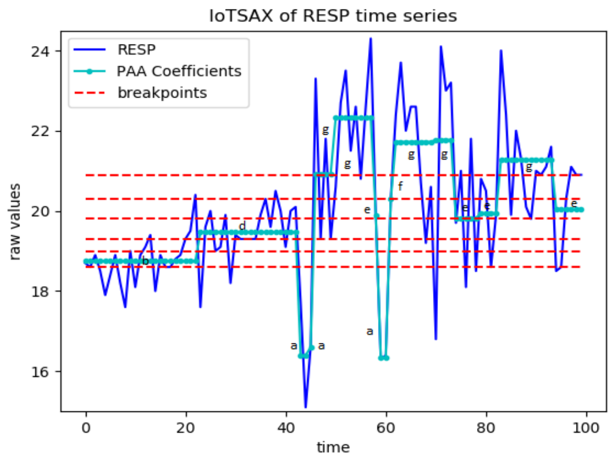


Figure 5.6: An example of IoTSAX approximation.

Algorithm 4 is the first step of IoTSAX approximation, where the best segments and PAA coefficients are found incrementally. In line 1, the DynPAA function gets as input two consecutive data points, whereas in line 3, the two points are transformed into a linear equation and the slope or derivative of the equation is calculated. The slope is later enhanced in line 4 by taking into account the amplitude between the two points. A difference between two consecutive enhanced slopes is found in line 5 and if it is higher than a user-defined threshold (line 6), then, the end of a dynamic segment has been found (line 12). Nevertheless, splitting consecutive sharp enhanced slopes that are first negative and then positive, and vice versa, into different dynamic segments, is avoided since they could be represented together by a mean value (PAA coefficient) in lines 7-11. Lines 14 and 15 are necessary updates of a possible end of a dynamic segment and enhanced slope for the next incremental rounds. Lines 17-19 dictate that if the window has reached its end, then, the end of the final dynamic segment has been reached. Lines 20 - 22 show that if an end of a dynamic segment has been found, then, the average value of the points within the segment constitutes its PAA coefficient.

Algorithm 4 DynPAA - Dynamic PAA

```
1: function DynPAA($x_{N-1}, x_N, thr$)                              ▷ $x_i$: data point
2:     $dynSegmentsPoint \leftarrow 0$
3:     $slope \leftarrow (x_N - x_{N-1})/(index(x_N) - index(x_{N-1}))$
4:     $enSlope \leftarrow slope * |x_N - x_{N-1}|$
5:     $diffOfEnSlopes \leftarrow ||enSlope| - |prEnSlope||$
6:     if $diffOfEnSlopes > thr$ then
7:         if $index(x_{N-1}) - prDynSegmentsPoint = 1$ then
8:             if $prEnSlope * enSlope > 0$ then
9:                 $dynSegmentsPoint \leftarrow index(x_{N-1})$
10:            end if
11:        else
12:            $dynSegmentsPoint \leftarrow index(x_{N-1})$
13:        end if
14:        $prDynSegmentsPoint \leftarrow index(x_{N-1})$
15:    end if
16:    $prEnSlope \leftarrow enSlope$
17:    if $endOfWindow$ then
18:        $dynSegmentsPoint \leftarrow index(x_N)$
19:    end if
20:    if $dynSegmentsPoint \neq 0$ then
21:        $\bar{X}_i \leftarrow \frac{1}{numOfPoints} \sum_{j=endOfPreviousSegment}^{dynSegmentsPoint} X_j$
22:    end if
23:    return $\bar{X}_i$                                               ▷ PAACoefficient (if any)
24: end function
```

Algorithm 5 is the last step of IoTSAX approximation, where the best dynamic alphabet size is defined as well as the PDF of the data. In line 1, the IoTSAX function gets as input the window data points X, the PAA coefficients of the window $\bar{X}$ from DynPAA function, a system-defined alphabet $\mathbb{A} = \{a, ..., z\}$, and $kMax = 10$ and $kMin = 3$ that define the maximum and minimum values, respectively, that the dynamic alphabet size can take. The alphabet $\mathbb{A}$ could apply to any environment. Also, according to the original SAX authors [187] and other studies [195], the range of an alphabet size is not too critical and a range of 5 to 8 seems to yield the best results; therefore the alphabet size range is set from 3 to 10. If the end of the window has been reached (line 2), then, KDE is called that through the calculation of mean and standard deviation of X, a necessary bandwidth is defined and the final PDF of the data is calculated based on a Gaussian kernel (line 3). Line 4 calculates the skewness of X based on its mean, mode (most frequent

Algorithm 5 IoTSAX - Dynamic SAX for IoT

```
1: function IoTSAX($X, \bar{X}, \mathbb{A}, kMax, kMin$)                ▷ $X$: window data points
2:     if $endOfWindow$ then
3:         $PDF \leftarrow KDE(X)$
4:         $skewness \leftarrow |mean - mode|/sd$
5:         $k \leftarrow round(kMax - ((kMax - kMin) * skewness))$
6:         for $i \leftarrow 0, k - 1$ do
7:             $\beta \leftarrow (i + 1) * totalAreaUnderPDF/k$
8:             $breakpoints \leftarrow PDF.quantileFunction(\beta)$
9:         end for
10:        for $i \leftarrow 0, sizeOf\bar{X}$ do
11:            for $j \leftarrow 0, sizeOfBreakpoints$ do
12:                if $breakpoints_j > \bar{X}_i$ then
13:                    $S_i \leftarrow \mathbb{A}_j$
14:                end if
15:            end for
16:        end for
17:    end if
18:    return $S$                                     ▷ Symbolic representation of $X$
19: end function
```

value), and standard deviation. The skewness shows the asymmetry of the PDF compared to its mean value. The higher the skewness, the lower the dynamic alphabet size, which is calculated in line 5 based also on the $kMax$ and $kMin$. All skewness values higher than 1 are considered equal to 1. The skewness may change for different window data points resulting in different alphabet sizes. Lines 6-9 define the breakpoints based on the logic of the original SAX. Specifically, the β variables define the cumulative probabilities that are required if the area under PDF is split into equiprobable regions of a number equal to the alphabet size. Then, the $breakpoints$ are the values where these probabilities occur. Finally, lines 10-16 also follow the logic of the original SAX, where according to which specific region between breakpoints the PAA coefficient falls within, a letter is selected for the symbolic representation S of X.

An example of the PDF via KDE of Fig. 5.6, compared to the normal distribution, is illustrated in Fig. 5.7. In the example, the histogram of the raw values and the estimated PDF show that it does not follow the Normal Distribution, which would be the assumption in the original SAX (here no normalisation occurs, otherwise the Normal Distribution would have $mean = 0$ as in the original SAX). The distribution is skewed with $skewness = 0.38$ resulting in $k = 7$ with breakpoints that split the area under PDF in k equiprobable regions.

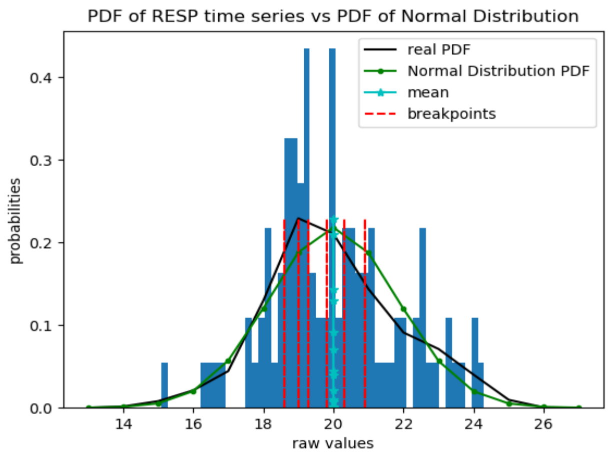


Figure 5.7: An example of the PDF via KDE of Fig. 5.6.

Approximate Rule-based Reasoning

The reasoning is the final step in the time series analysis in order to create the quad. It consists of two main parts, which is the interpretation of the symbolic representation and the approximate reasoning rules. More details are given in Algorithm 6.

Algorithm 6 Quad - Reasoning

```
1: function QUAD($\bar{X}, S$)                                ▷ $\bar{X}$: window PAACoefficients
2:     $subject \leftarrow entityOfWindow$
3:     $property \leftarrow attributeOfWindow$
4:     $object \leftarrow mean(\bar{X})$
5:     for $i \leftarrow 1, sizeOf\bar{X}$ do
6:         $meaningOfDistance \leftarrow distance(S_i, S_{i-1})$
7:     end for
8:     $interpretation \leftarrow shapeOfS(meaningOfDistance)$
9:     $inference \leftarrow approximateRules(property, object)$
10:    $quad \leftarrow < subject >< property >< object >< interpretation, inference >$
11:    return $quad$                                          ▷ quad of event
12: end function
```

Algorithm 6 depicts all the steps that create the final quad that is sent as a notification to the subscriber. In line 1, the Quad function gets as input the PAA coefficients of the window $\bar{X}$ and the symbolic representation S from the IoTSAX function. In lines 2 - 4, the main triple is created that consists of the subject and the property of the window, as well as the object, which is the mean value of the window's PAA coefficients. In lines 5 - 7, the distance of subsequent letters in the symbolic representation is calculated, and meaning is defined. For example, for $k = 3$, if the distance is -1, then, the meaning is that there is a slight increase, if it is -2, then, there is a sharp increase, etc. The values that $meaningOfDistance$ can take are {slightIncrease, increase, sharpIncrease, slightDecrease, decrease, sharpDecrease}. In line 8, an interpretation of the shape of the symbolic representation is extracted based on the meaning of distances. According to the popularity of the meanings, an initial interpretation may have one of the following values {"overall increasing", "overall slightly increasing", "overall decreasing", "overall slightly decreasing", "overall neutral"}. Then, based on whether they have sharp increases/decreases, their sequence and their frequency, the interpretation may additionally have some of the following values {"one sharp increase/decrease followed by one sharp decrease/increase" or "reoccurring sharp increases/decreases followed by sharp decreases/increases", "one sharp increase" or "reoccurring sharp increases", "one sharp decrease" or "reoccurring sharp decreases"}. Line 9 refers to the reasoning result based on approximate manual rules. Specifically, a property contains a set of rules that define the context to be inferred depending on the property's value, which in this case is the object. These rules are approximate; therefore, if a conceptually-related property to the one existing in the rules were to be used, then, the reasoner would be able to find the relevance and use the appropriate rule. Finally, in line 10, a quad is created that apart from the triple, it also contains the contextual information.

Rule-based reasoning is a simple, time-efficient, and low-memory way to infer high-level abstractions of sensor data. Nevertheless, one of their disadvantages is that they cannot be adapted to heterogeneous and complicated sensors. This problem can be addressed by approximate reasoning rules via word embeddings, which represent words into vectors in a vector space. Words that are more conceptually-related, including semantically opposite words like antonyms, exist closer to the vector space.

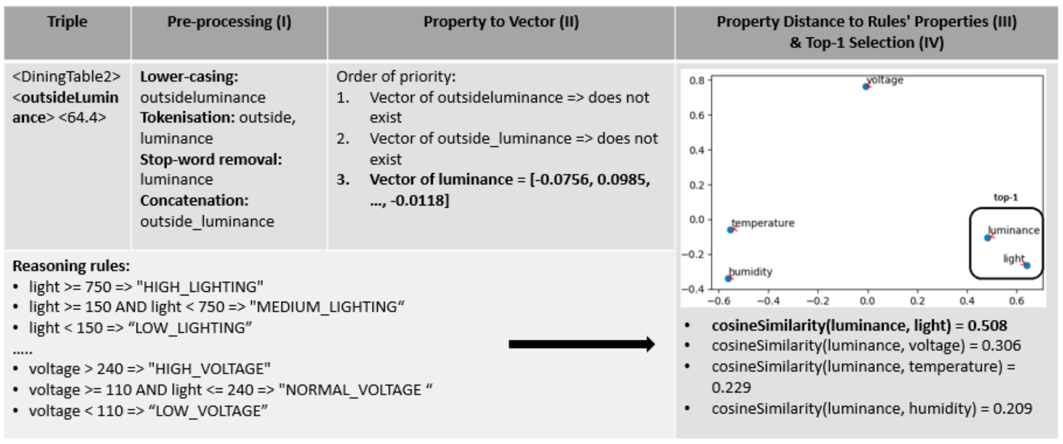
The stages of finding the most relevant property existing in the rules are given in Algorithm 7. In line 1, the Approximation of Reasoning Rules function gets as input the triple tr, the reasoning rules $reasoningRules$, the word embedding model $word2VecModel$, a POS Tagging model $posTagger$, a lemmatisation model $lemmatiser$, and stopwords $stopwords$. In lines 2 - 3, the attribute of the triple as well as all the attributes of the reasoning rules are extracted. In lines 4 and 7, the corresponding attributes are pre-processed by lower-casing, tokenising, removing stop-words, lemmatisation as well as POS tagging (depending on the

Algorithm 7 Approximation of Reasoning Rules

```
1: function Approximation of Reasoning Rules($tr, reasoningRules, word2VecModel, posTagger, lemmatiser, stopwords$)
   ▷ $tr$: triple, $word2VecModel$: embedding model
2:     $property \leftarrow attributeOfTriple(tr)$
3:     $rProperties \leftarrow attributeOfReasoningRules(reasoningRules)$
4:     $propertiesIdx \leftarrow preprocessProperty(property, posTagger, lemmatiser, stopwords)$
5:     $word2VecIdx \leftarrow createWord2VecIdx(property, word2VecModel, propertiesIdx)$
6:     for $rProperty \leftarrow rProperties$ do
7:         $propertiesIdx \leftarrow preprocessProperty(rProperty, posTagger, lemmatiser, stopwords)$
8:         $word2VecIdx \leftarrow createWord2VecIdx(rProperty, word2VecModel, propertiesIdx)$
9:     end for
10:    $propertyVector \leftarrow word2VecIdx.get(property)$
11:    $rPropertiesVectors \leftarrow word2VecIdx.get(rProperties)$
12:    $scoredProperties \leftarrow cosineSimilarity(propertyVector, rPropertiesVectors)$
13:    $sortedScoredProperties \leftarrow sort(scoredProperties)$
14:    $top1rProperty \leftarrow top1(sortedScoredProperties)$
15:    $approximateReasoningRulesIdx \leftarrow top1rProperty$
16:    return $approximateReasoningRulesIdx$    ▷ index that maps a triple's property to the rules' property with the
   highest similarity
17: end function
```

word embedding model used), and concatenating each token with an underscore. All words ranging from original to tokens to concatenated ones are stored into an index ($propertiesIdx$). In lines 5 and 8, an index is created ($word2VecIdx$) that contains the equivalent word embedding vectors of the words contained in the $propertiesIdx$. Priority is given to the original word, then, the concatenated one and, finally, if the previous do not exist, then, an average is taken by all of the tokens' vectors. Lines 10 - 11 extract the word embedding vectors of the attributes based on the $word2VecIdx$. A distance (cosine similarity) is calculated between the vector of the triple's attribute and the ones of all the reasoning rules' attributes, in Line 12. Line 13 sorts the distances from highest to lowest. Finally, in line 14, the reasoning rules' property that has the highest score is picked as the most relevant property to the triple's one, and it is stored in an index ($approximateReasoningRulesIdx$) in line 15 for further use.

An example is given in Fig. 5.8, where only a subset of all the reasoning rules and their properties is given for visualisation purposes. In the example, rigid reasoning rules would not relate the property "outsideLuminance" to "light"; therefore, no reasoning result would occur based on the semantically-coupled reasoning rules.



Triple	Pre-processing (I)	Property to Vector (II)	Property Distance to Rules' Properties (III) & Top-1 Selection (IV)
<DiningTable2> **<outsideLuminance>** <64.4>	**Lower-casing:** outsideluminance **Tokenisation:** outside, luminance **Stop-word removal:** luminance **Concatenation:** outside_luminance	Order of priority: 1. Vector of outsideluminance => does not exist 2. Vector of outside_luminance => does not exist **3. Vector of luminance = [-0.0756, 0.0985, ..., -0.0118]**	
Reasoning rules: • light >= 750 => "HIGH_LIGHTING" • light >= 150 AND light < 750 => "MEDIUM_LIGHTING" • light < 150 => "LOW_LIGHTING" ..... • voltage > 240 => "HIGH_VOLTAGE" • voltage >= 110 AND light <= 240 => "NORMAL_VOLTAGE" • voltage < 110 => "LOW_VOLTAGE"			• **cosineSimilarity(luminance, light) = 0.508** • cosineSimilarity(luminance, voltage) = 0.306 • cosineSimilarity(luminance, temperature) = 0.229 • cosineSimilarity(luminance, humidity) = 0.209

Figure 5.8: An example of the stages of approximate reasoning rules.

5.5 Evaluation

This section provides the evaluation of the approach that consists of the methodology followed and the datasets used as well as the metrics and the final results.

5.5.1 Datasets and Methodology

A combination of different real-world datasets has been used that relate to the applications of Healthcare and Smart Cities, analysed below:

MIMIC II Database[1]: This is a public dataset that contains heart-related readings (e.g. heart rate, central venous pressure etc.) of patients that developed or were at risk of developing an acute hypotensive episode. All readings were pre-processed into literals with properties the measurements {centralVenousPressure etc.}, objects the original values, and subjects the people's name. Missing and final zero values deriving from equipments' disconnection or noise were deleted. The annotations, which contained alerts when a reading was lower or higher than normal, were used as a reference for the creation of manual rules. A representative subset of all the available patients was selected from all four groups since the data contains redundancy.

Intel Lab Data[2]: This is a public dataset generated by 54 sensors deployed in the Intel Berkeley Research lab between February 28th and April 5th, 2004. The sensor readings involve environmental aspects (e.g. temperature, humidity

[1]`https://archive.physionet.org/challenge/2009/training-set.shtml`
[2]`http://db.csail.mit.edu/labdata/labdata.html`

etc.). A representative subset of all the available sensors was selected since the data contains redundancy. Specifically, as in Truong et al.[202], different 24-hour sets of data were used for each property by eleven sensors, five of which are nearby in the Lecture Hall area (sensors 6-10) and the other six in the Dining Room area (sensors 31-36). By selecting nearby sensors, an emphasis was given on the relatedness among the different sensor values. The original data was pre-processed into literals with properties {temperature, humidity etc.}, objects the original values, and subjects that were named according to the sensor the data was generated from as follows: 6: LectureHallSideExit, 7: LectureHallAisle, 8: LectureHallLeftFrontRow, 9: LectureHallLeftBackRow, 10: LectureHallRearExit, 31: DiningTable4, 32: DiningTable3, 33: RightDiningHall, 34: DiningTable2, 35: LeftDiningHall and 36: DiningTable1.

CityPulse[3]: This is a public dataset that contains readings regarding weather (e.g. dew point, wind speed etc.), pollution (e.g. ozone, carbon monoxide etc.), and road traffic (e.g. vehicle count). Different 24-hour sets of the above data were used for each property. The original data was pre-processed into literals with properties {sulfurDioxide, humidity etc.}, objects the original values, and subjects the place/building name.

UCI Electric Consumption[4]: This is a public dataset that contains readings regarding electric power consumption in one household (e.g. global active power, voltage etc.). Different 24-hour sets of the above data were used for each property. The original data was pre-processed into literals with properties {globalActivePower, globalIntensity etc.}, objects the original values, and subjects that were named according to the sub-metering sensor the data was generated from as follows: 0: Kitchen, 1: LaundryRoom, 2: ElectricEquipment and the rest Property.

In order to observe the consensus among the results of different Abstractive Publish/Subscribe Summarisation Systems, 10 entities were examined, where each corresponds to a separate system. It should be noted that noise (e.g. white noise) in the original data was ignored as out of the scope of the work. The reader is directed to Vaseghi [203] on advanced signal processing techniques. Also, all the original attributes' semantics generated by the sensors were examined. The manual rules

[3]`http://iot.ee.surrey.ac.uk:8080/datasets.html`

[4]`https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption`

were mostly created based on the M3 framework's rules[5] and the annotations in the MIMIC II Database, as aforementioned. An overview of the datasets and their characteristics is given in Table 5.3.

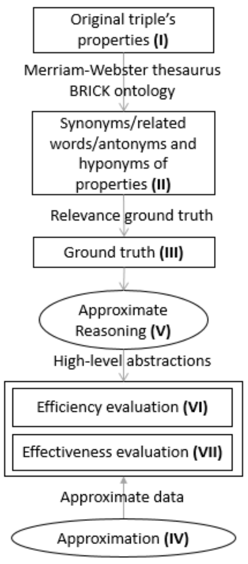
Table 5.3: Dataset Characteristics

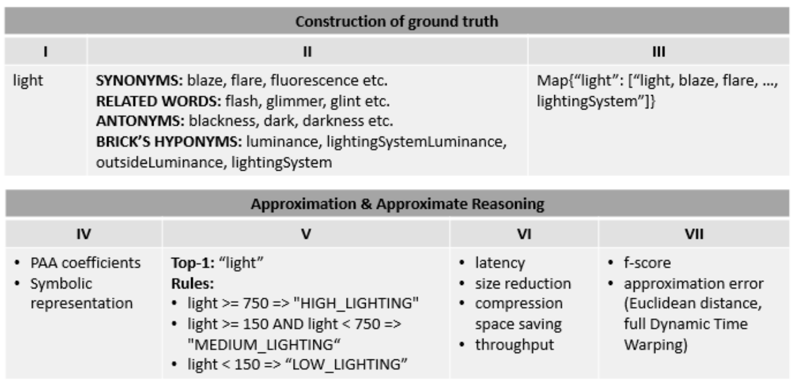
Dataset	Domain	Sensors	Number of Sensors Used	Number of Events per Entity (Average)	Original Schema	Pre-processing
MIMIC II Database	Healthcare	heart readings	11	8.8K	attribute-value	literals, missing values/final zeros deleted
Intel Lab Data	Smart Cities	temperature, voltage, light, humidity	44	262.9K	attribute-value	literals, missing light values replaced with nearby values, spatially nearby sensors selected
CityPulse	Smart Cities	weather, pollution, road traffic	13	187	attribute-value	literals
UCI Electric Consumption	Smart Cities	household electric readings	7	200.2K	attribute-value	literals

The evaluation methodology is illustrated in Fig. 5.9. It examines both the efficiency and effectiveness of the approach. The efficiency involves metrics like latency, throughput, etc. after the approximation and reasoning take place. The effectiveness involves metrics like F-score and approximation error. The F-score is based on a relevance ground truth (partly inspired by Hasan et al. [39] ground truth construction). In order to show the heterogeneity involved in IoT environments, a ground truth set is constructed by choosing each original data property (e.g. light) and storing its synonyms (e.g. blaze), related words (e.g. flash), and antonyms (e.g. blackness) deriving from the Merriam-Webster[6] thesaurus as well as its hyponyms (children) observed in the BRICK ontology[7] (e.g. luminance). In the case of polysemy, the words that applied to the application in use were chosen (e.g. "pressure" as body measurement in the medical case and as environmental one in the Smart Cities case). An example of the evaluation methodology is given in Fig. 5.9 for a triple with property "light".

The streaming rate by the publishers and subscribers follows a uniform distribution as in the real case. Each publisher generates a stream related to a measurement (e.g. temperature) of an entity. There is only one subscriber that generates subscriptions based on the entities. All runs took place in a laptop with Intel(R) Core(TM) i7-6600U CPU@2.60GHz 2.80GHz, and 16GB of RAM. Regarding the implementation, for the Resource Description Framework (RDF)

[5] `https://github.com/gyrard/M3Framework/tree/master/war/RULES`
[6] `https://www.merriam-webster.com/thesaurus`
[7] `https://brickschema.org/#home`


Original triple's properties (I)
Merriam-Webster thesaurus
BRICK ontology
Synonyms/related words/antonyms and hyponyms of properties (II)
Relevance ground truth
Ground truth (III)
Approximate Reasoning (V)
High-level abstractions
Efficiency evaluation (VI)
Effectiveness evaluation (VII)
Approximate data
Approximation (IV)



Construction of ground truth		
I	II	III
light	**SYNONYMS:** blaze, flare, fluorescence etc. **RELATED WORDS:** flash, glimmer, glint etc. **ANTONYMS:** blackness, dark, darkness etc. **BRICK'S HYPONYMS:** luminance, lightingSystemLuminance, outsideLuminance, lightingSystem	Map{"light": ["light, blaze, flare, ..., lightingSystem"]}

Approximation & Approximate Reasoning			
IV	V	VI	VII
• PAA coefficients • Symbolic representation	**Top-1:** "light" **Rules:** • light >= 750 => "HIGH_LIGHTING" • light >= 150 AND light < 750 => "MEDIUM_LIGHTING" • light < 150 => "LOW_LIGHTING"	• latency • size reduction • compression space saving • throughput	• f-score • approximation error (Euclidean distance, full Dynamic Time Warping)

Figure 5.9: Evaluation methodology and an example for a triple with property "light".

models, Apache Jena[8] was used, for the embedding model deeplearning4j[9] was used, for the KDE the corresponding class by Smile[10] was modified, for tagging and lemmatisation Apache OpenNLP[11] was used.

5.5.2 Metrics

Several metrics have been used to evaluate the effectiveness and efficiency of the approach. The effectiveness evaluation focuses on the correctness, whereas the efficiency one focuses on the performance.

Correctness

Correctness consists of the F-score and approximation error metrics.

F-score The $F-score$ metric calculates the effectiveness of the approximate reasoner. According to Table 5.4, each property in the ground truth is linked to an original triple's property (Fig. 5.9); therefore, if the reasoner finds this relevance, then, it is considered as True Positive (TP), otherwise, it is a False Positive (FP)

[8]https://jena.apache.org/
[9]https://deeplearning4j.org/
[10]https://haifengl.github.io/smile/nlp.html
[11]https://opennlp.apache.org/

Table 5.4: Approximate reasoner evaluation components

Ground Truth / Approach	conceptually relevant	not conceptually relevant
conceptually relevant	TP	FP
not conceptually relevant	FN	-

as well as a False Negative (FN), since the original property was found irrelevant but was indeed relevant according to the ground truth. For example, in the case of Fig. 5.8 the "outsideLuminance" property is correctly found most related to the property "light", therefore, the F-score = 1.0, otherwise F-score = 0.0. Specifically, the $F-score$ is defined as:

$$precision = \frac{TP}{TP+FP} \quad \text{and} \quad recall = \frac{TP}{TP+FN} \qquad 5.1$$

$$F-score = 2 \times \frac{precision \times recall}{precision + recall} \qquad 5.2$$

Approximation Error Based on the study of Ding et al. [193] the most important time series distance measures are the Euclidean Distance (ED) and the full Dynamic Time Warping (DTW). The authors also found that the two measurements complement each other as for big datasets the accuracy of ED is higher, whereas for small datasets the DTW's accuracy is higher. Therefore, these two measures are both used to define the distance between the original time series and the approximated one via PAA. These are defined as:

$$ED = \sqrt{\sum_{i=1}^{N}(x_i - \bar{x}_i)^2} \qquad 5.3$$

$$DTW[i][j] = \mid x_{i-1} - \bar{x}_{j-1} \mid + min(min(DTW[i-1][j], DTW[i][j-1]), DTW[i-1][j-1]) \qquad 5.4$$

Performance

The performance consists of the end-to-end latency, size reduction, compression space-saving, and throughput metrics. The end-to-end latency and throughput metrics have been defined in Chapter 4, whereas the rest are analysed below.

Size Reduction This metric is the reduction in the number of messages that the subscriber receives. It is defined as:

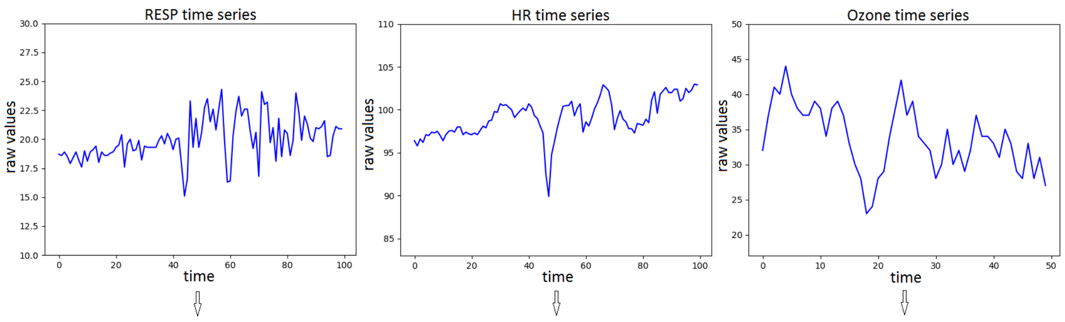
$$size\ reduction = \#forwarded\ messages \tag{5.5}$$

Compression Space-Saving This metric is the percentage of the reduction in data space occurring via approximation and symbolic representation. It is defined as:

$$compression\ space-saving = (1 - \frac{symbolic\ representation\ length}{original\ length}) * 100 \tag{5.6}$$

5.5.3 Results

The results of the IoTSAX approach are analysed below. An example of the abstractive summaries produced is illustrated in Fig. 5.10.



<Patient> <respirationRate> <20.026> <overall slightly increasing with reoccuring sharp increases/decreases followed by sharp decreases/increases, ABNORMAL>

<Patient> <heartRate> <99.386> <overall increasing, NORMAL>

<Property> <ozone> <32.826> <overall slightly decreasing with one sharp increase, MAXIMUM_DAILY_PERMITTED_OZONE>

Figure 5.10: Quad-based notification examples for RESP, HR, and ozone time series, respectively.

Data Approximation

The correctness and performance of data approximation are analysed for the following approaches: 1) the typical Pub/Sub approach (no fusion no abstraction), where all events regarding an entity are sent as a notification to the subscriber with no approximation or reasoning involved, 2) SAX approximation with reasoning (fusion SAX abstraction), where according to the original authors [187] and other studies [195], the use of $n = 5$ to 8 and $k = 5$ to 8 generally yields the best results, and 3) the proposed IoTSAX approximation with reasoning (fusion IoTSAX abstraction), where different threshold values are used as well as $kMax = 10$ and $kMin = 3$.

The results are shown below for 10 entities for each use case. For Healthcare's use case, a combination of 6 to 11 publishers is used depending on the availability of the data for each entity. For the Smart Cities use case, 64 publishers are used for each entity. Each publisher is related to a measurement. The window sizes differ from 100 to 50 for Healthcare and Smart Cities, respectively. The duration of each experiment ranges from 15 to 30 minutes depending on the availability of the data.

End-to-End Latency and Throughput In Table 5.5 and Table 5.6 the end-to-end latency and throughput results are illustrated, respectively. No fusion no abstraction has the best latency as no approximation or reasoning as well as waiting for a window to be populated are involved when sending notifications. Surprisingly, this has no effect on the throughput showing that a fast approach that generates an overload of notifications can have a negative effect on the performance of a system. The abstraction approaches can achieve from 2 to 3 times more the latency of no fusion no abstraction one as further processing and population of windows are involved. The Healthcare's use case has higher latency; therefore, lower throughput compared to the Smart Cities one, because fewer publishers, hence fewer processing windows are created (less parallelism), and the window size involved is bigger showing the effect of both parallelism and window size can pose in the system performance.

In terms of the comparison among the abstraction approaches, there is no significant consensus that indicates which approach or parameter is better. Specifically, for different SAX parameters, the processing is similar time-wise; therefore, it should pose no significant system performance differentiation. This is also depicted in the throughput results showing that indeed SAX and IoTSAX are appropriate approaches to be used in a streaming environment. Nevertheless, it is observed that IoTSAX achieves similar or better latencies compared to SAX. It is believed that the bigger and more complex the computations are, the better the latency will be in the IoTSAX approach as it is incremental compared to SAX. There

Table 5.5: Average end-to-end latency results for both use cases

Use Case	Approach	Parameters	Average end-to-end latency in ms for each entity									
			E0	E1	E2	E3	E4	E5	E6	E7	E8	E9
Healthcare	no fusion no abstraction		**21537**	**11143**	**22108**	**23227**	**36355**	**30171**	**20649**	**23290**	**30186**	**32119**
			Average = 25078.5, Standard Deviation = 6859.736									
	fusion SAX abstraction	n = 5, k = 5 windowSize = 100	61475	62213	62710	63954	65937	65070	63304	68543	61509	67088
			Average = 64180.3, Standard Deviation = 2295.018									
		n = 5, k = 8 windowSize = 100	61005	57264	63554	66322	63882	69938	66484	58642	62605	64939
			Average = 63463.5, Standard Deviation = 3606.05									
		n = 8, k = 5 windowSize = 100	64280	58340	66329	67186	66691	64569	62459	65641	61901	62966
			Average = 64036.2, Standard Deviation = 2564.829									
		n = 8, k = 8 windowSize = 100	70391	58308	68330	64979	61405	66293	63423	65698	67282	64106
			Average = 65021.5, Standard Deviation = 3294.044									
	fusion IoTSAX abstraction	thr = 1 windowSize = 100	69414	58590	65360	62465	66550	64522	62202	67280	63552	62015
			Average = 64195.0, Standard Deviation = 2954.864									
		thr = 4 windowSize = 100	66372	57717	67063	66220	63700	69760	63479	68776	64370	64856
			Average = 65231.3, Standard Deviation = 3185.447									
Smart Cities	no fusion no abstraction		**11684**	**11275**	**11468**	**11156**	**11257**	**10817**	**9976**	**11933**	**10854**	**10174**
			Average = 11059.4, Standard Deviation = 589.385									
	fusion SAX abstraction	n = 5, k = 5 windowSize = 50	43029	36424	39507	40429	40250	41655	40140	38725	42147	40649
			Average = 40295.5, Standard Deviation = 1760.513									
		n = 5, k = 8 windowSize = 50	44582	40787	40020	38790	40477	39157	39618	40081	39714	38403
			Average = 40162.9, Standard Deviation = 1628.99									
		n = 8, k = 5 windowSize = 50	42298	39677	41441	38651	40798	40956	40001	40450	40881	41388
			Average = 40654.1, Standard Deviation = 972.568									
		n = 8, k = 8 windowSize = 50	41452	42174	41953	39614	39451	40812	40594	41071	38856	37174
			Average = 40315.1, Standard Deviation = 1468.351									
	fusion IoTSAX abstraction	thr = 0.5 windowSize = 50	41263	43723	40376	38152	42016	39230	40947	38593	40561	40732
			Average = 40559.3, Standard Deviation = 1557.831									
		thr = 1 windowSize = 50	41643	39676	37395	39771	42256	41271	40454	39791	40475	40907
			Average = 40363.9, Standard Deviation = 1277.564									

Table 5.6: Throughput results for both use cases

Use Case	Approach	Parameters	Throughput in events per second for each entity									
			E0	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**	**E8**	**E9**
Healthcare	no fusion no abstraction		**22.117**	**19.673**	22.052	20.021	13.907	15.980	**21.656**	22.033	13.231	**16.064**
			Average = 18.673, Standard Deviation = 3.357									
	fusion SAX abstraction	n = 5, k = 5 windowSize = 100	22.026	**19.673**	**22.089**	19.917	**14.069**	**16.099**	21.651	22.077	13.231	15.999
			Average = 18.683, Standard Deviation = 3.324									
		n = 5, k = 8 windowSize = 100	21.993	**19.673**	21.911	**20.110**	14.037	15.937	**21.656**	21.864	**13.232**	15.949
			Average = 18.636, Standard Deviation = 3.311									
		n = 8, k = 5 windowSize = 100	22.068	**19.673**	22.006	20.094	13.976	16.013	21.646	22.063	13.231	15.994
			Average = 18.676, Standard Deviation = 3.345									
		n = 8, k = 8 windowSize = 100	22.054	**19.673**	21.944	19.999	13.987	16.096	21.649	22.053	13.231	16.021
			Average = 18.671, Standard Deviation = 3.323									
	fusion IoTSAX abstraction	thr = 1 windowSize = 100	22.066	**19.673**	22.024	19.878	12.015	16.016	21.625	**22.113**	13.231	15.939
			Average = 18.458, Standard Deviation = 3.657									
		thr = 4 windowSize = 100	22.042	**19.673**	**22.089**	19.987	12.044	16.057	21.648	21.956	13.231	15.963
			Average = 18.469, Standard Deviation = 3.643									
Smart Cities	no fusion no abstraction		**53.070**	95.209	97.319	94.058	92.289	88.802	87.993	88.742	88.849	87.766
			Average = 87.41, Standard Deviation = 11.885									
	fusion SAX abstraction	n = 5, k = 5 windowSize = 50	**53.070**	95.070	97.236	94.139	92.171	88.551	87.866	88.700	88.781	**88.147**
			Average = 87.373, Standard Deviation = 11.866									
		n = 5, k = 8 windowSize = 50	**53.070**	95.221	97.267	**94.436**	92.206	88.704	87.806	88.513	88.943	87.904
			Average = 87.407, Standard Deviation = 11.897									
		n = 8, k = 5 windowSize = 50	**53.070**	**95.488**	97.472	94.283	92.159	88.816	87.839	88.740	88.839	88.137
			Average = 87.484, Standard Deviation = 11.925									
		n = 8, k = 8 windowSize = 50	**53.070**	95.109	**97.594**	94.203	92.149	88.802	87.870	88.574	88.762	88.024
			Average = 87.416, Standard Deviation = 11.902									
	fusion IoTSAX abstraction	thr = 0.5 windowSize = 50	**53.070**	95.310	96.963	93.991	92.261	88.613	87.616	88.748	88.826	87.896
			Average = 87.329, Standard Deviation = 11.854									
		thr = 1 windowSize = 50	**53.070**	95.377	97.393	94.083	**92.568**	**88.853**	**87.998**	**88.936**	**89.026**	87.893
			Average = 87.52, Standard Deviation = 11.921									

might also be a possibility that KDE posed an additional computational burden in the case of IoTSAX, hence the similar latencies with SAX in some entities. Finally, it is observed that stricter thresholds (lower thresholds) in IoTSAX show slightly lower latencies as more segmentation takes place.

It is important to note that the system performance was checked if no abstraction takes place, but with fusion, meaning that no approximation or reasoning is involved, but the latencies and throughput are affected by the time each window takes to be populated and no significant differentiation in the results was observed. This shows how powerful SAX and IoTSAX approximation are in a streaming environment.

Size Reduction In Table 5.7, it is seen that when abstraction is used, the number of forwarded messages sent to the subscriber can be reduced from 98% to more than 99%, depending on the window size. The smaller the window size, the lower the reduction. This complies with the latency results that show that although no abstraction approach can be up to 3 times faster than the abstraction ones, it creates an overload of notifications that can overwhelm both the system and the subscriber. It is also observed that more messages are forwarded in the case of Smart Cities that also complies with its lower latency and higher throughput as there is more parallelism and more publishers generating events.

Compression Space-Saving and Approximation Error In Table 5.8 and Table 5.9 the average compression space-saving and the overall approximation error of all time series are illustrated, respectively.

In terms of compression space-saving, it is observed that in the case of SAX the compression is higher for smaller segment numbers (n) as more data is grouped in segments to be approximated. It is also seen that since n is static for all entities in SAX, there is no differentiation in space-saving compared to the dynamic approach of IoTSAX, where the segmentation is based on the fluctuations of the data of each entity. This is highly depicted in the space-saving percentage between Healthcare's use case and Smart Cities one. In Healthcare, SAX performs higher compression compared to IoTSAX as more fluctuations of data exist (patient's measurements may have extreme values); therefore, more segmentation takes place in IoTSAX resulting in smaller space-saving. On the other hand, in the case of Smart Cities, there are not many fluctuations in sensor readings (the

Table 5.7: Size reduction results for both use cases

Use Case	**Approach**	**Parameters**	**Forwarded messages for each entity**									
			E0	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**	**E8**	**E9**
Healthcare	no fusion no abstraction		26345	8737	26476	23911	14415	19190	21980	26340	13960	19091
			Average = 20044.5, Standard Deviation = 5810.345									
	fusion abstraction	windowSize = 50	526.90	174.74	529.52	478.22	288.30	383.80	439.60	526.80	279.20	381.82
			Average = 400.89, Standard Deviation = 116.207									
		windowSize = 100	263.45	87.37	264.76	239.11	144.15	191.90	219.80	263.40	139.60	190.91
			Average = 200.445, Standard Deviation = 58.103									
		windowSize = 150	**175.63**	**58.25**	**176.51**	**159.41**	**96.10**	**127.93**	**146.53**	**175.60**	**93.07**	**127.27**
			Average = 133.63, Standard Deviation = 38.735									
Smart Cities	no fusion no abstraction		44385	60414	62467	62363	58323	61030	58262	57138	57087	60660
			Average = 58212.9, Standard Deviation = 4977.332									
	fusion abstraction	windowSize = 50	887.70	1208.28	1249.34	1247.26	1166.46	1220.60	1165.24	1142.76	1141.74	1213.20
			Average = 1164.258, Standard Deviation = 99.547									
		windowSize = 100	443.85	604.14	624.67	623.63	583.23	610.30	582.62	571.38	570.87	606.60
			Average = 582.129, Standard Deviation = 49.773									
		windowSize = 150	**295.90**	**402.76**	**416.45**	**415.75**	**388.82**	**406.87**	**388.41**	**380.92**	**380.58**	**404.40**
			Average = 388.086, Standard Deviation = 33.182									

Table 5.8: Average compression space-saving percentage results for both use cases

Use Case	**Approach**	**Parameters**	**Average compression space-saving in % for each entity**									
			E0	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**	**E8**	**E9**
Healthcare	fusion SAX abstraction	n = 5 windowSize = 100	**95**	**95**	**95**	**95**	**95**	**95**	**95**	**95**	**95**	**95**
			Average = 95.0, Standard Deviation = 0.0									
		n = 8 windowSize = 100	92	92	92	92	92	92	92	92	92	92
			Average = 92.0, Standard Deviation = 0.0									
	fusion IoTSAX abstraction	thr = 1 windowSize = 100	76.69	71.75	76.66	82.31	86.51	71.99	75.79	80.54	78.09	76.17
			Average = 77.65, Standard Deviation = 4.28									
		thr = 4 windowSize = 100	85.83	82.35	86.78	89.14	92.043	81.32	85.73	89.02	83.97	81.46
			Average = 85.764, Standard Deviation = 3.4									
Smart Cities	fusion SAX abstraction	n = 5 windowSize = 50	90	90	90	90	90	90	90	90	90	90
			Average = 90.0, Standard Deviation = 0.0									
		n = 8 windowSize = 50	84	84	84	84	84	84	84	84	84	84
			Average = 84.0, Standard Deviation = 0.0									
	fusion IoTSAX abstraction	thr = 0.5 windowSize = 50	92.11	93.88	93.74	94.19	93.84	92.90	92.21	92.21	93.96	92.51
			Average = 93.155, Standard Deviation = 0.801									
		thr = 1 windowSize = 50	**93.18**	**94.63**	**94.46**	**94.99**	**94.38**	**93.60**	**92.92**	**92.84**	**94.56**	**93.37**
			Average = 93.893, Standard Deviation = 0.753									

Table 5.9: Approximation error via ED and DTW results for both use cases

Use Case	Approach	Parameters	Overall Euclidean distance for each entity									
			E0	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**	**E8**	**E9**
Healthcare	fusion SAX abstraction	n = 5	21169	5328	31222	25435	4501	19790	16909	19923	16171	21679
		windowSize = 100	Average = 18212.7, Standard Deviation = 7794.009									
		n = 8	19594	4943	28211	24476	4182	17531	15975	18535	14855	19653
		windowSize = 100	Average = 16795.5, Standard Deviation = 7161.622									
	fusion IoTSAX abstraction	thr = 1	**11936**	**2857**	**13098**	**16270**	**2101**	**7648**	**9733**	**11318**	**10231**	10584
		windowSize = 100	**Average = 9577.6, Standard Deviation = 4147.599**									
		thr = 4	12331	3060	13736	16801	2455	8120	10198	11729	10264	**10290**
		windowSize = 100	Average = 9898.4, Standard Deviation = 4211.284									
Smart Cities	fusion SAX abstraction	n = 5	11162	12398	15030	11089	22488	43026	19637	20162	13203	23566
		windowSize = 50	Average = 19176.1, Standard Deviation = 9103.185									
		n = 8	8304	9523	11525	8178	17695	34960	15360	15932	9485	18839
		windowSize = 50	Average = 14980.1, Standard Deviation = 7650.237									
	fusion IoTSAX abstraction	thr = 0.5	**4033**	**5103**	**7845**	**4091**	**9443**	**21318**	**9190**	**9680**	**5910**	**10232**
		windowSize = 50	**Average = 8684.5, Standard Deviation = 4772.718**									
		thr = 1	4220	5297	7870	4158	9488	21450	9332	9830	6100	10361
		windowSize = 50	Average = 8810.6, Standard Deviation = 4766.821									

Use Case	Approach	Parameters	Overall dynamic time warping similarity for each entity									
			E0	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**	**E8**	**E9**
Healthcare	fusion SAX abstraction	n = 5	53794	14772	87985	59539	11636	56912	37657	48055	59416	58855
		windowSize = 100	Average = 48862.1, Standard Deviation = 21492.064									
		n = 8	55783	15717	79014	64279	11233	57405	40690	49766	55262	57226
		windowSize = 100	Average = 48637.5, Standard Deviation = 19900.387									
	fusion IoTSAX abstraction	thr = 1	42433	**10393**	**43449**	**51357**	**7261**	**23296**	**27767**	**34773**	**40598**	**37343**
		windowSize = 100	**Average = 31867.0, Standard Deviation = 13773.761**									
		thr = 4	**42318**	11143	48300	54597	8162	25870	28897	37075	41118	37675
		windowSize = 100	Average = 33515.5, Standard Deviation = 14322.631									
Smart Cities	fusion SAX abstraction	n = 5	44063	50537	55845	47343	69124	137279	64767	72531	51053	78102
		windowSize = 50	Average = 67064.4, Standard Deviation = 25816.11									
		n = 8	30776	36518	43717	32159	56274	117201	47248	55230	33416	68362
		windowSize = 50	Average = 52090.1, Standard Deviation = 24633.056									
	fusion IoTSAX abstraction	thr = 0.5	**14514**	**16460**	**27582**	**13186**	**29181**	**61673**	**26368**	**31497**	**17946**	**35314**
		windowSize = 50	**Average = 27372.1, Standard Deviation = 13547.938**									
		thr = 1	15079	17019	27634	13407	29377	62163	26961	32094	18314	35689
		windowSize = 50	Average = 27773.7, Standard Deviation = 13564.286									

readings may be the same for a window); therefore, less segmentation occurs in IoTSAX compared to SAX that has a static segmentation number. This means that even in redundant sensor readings, SAX will always create a bigger symbolic representation compared to IoTSAX; therefore, saving less space. It should be observed though that in IoTSAX the percentage of space-saving is not only dependent on the data but also the threshold, suggesting that smaller thresholds perform stricter dynamic segments, hence less compression space-saving. Finally, it is seen that the higher the window, the more the effect of the compression in saving space.

In terms of the approximation error, it is observed how well (non-normalised in the case of SAX) PAA coefficients behave compared to the original data. It is seen that ED and DTW sometimes may not have a similar pattern for the different approaches and use cases, which depicts why both measures should be used as each has their strengths and weaknesses. In this work's case, an emphasis is put slightly more on the ED as it depicts a straight line difference between the raw and approximate time series values, whereas DTW observes more the general similarity between the two time series. It is seen that the error results follow similar observations to the compression space-saving percentage ones. Specifically, in SAX, smaller segment numbers (n) result in bigger distances for ED, and the most part for DTW, as higher compression takes place; therefore, more error occurs. In IoTSAX, lower thresholds result in lower approximation error as stricter dynamic segments are created, hence less compression. Generally, both measures agree that IoTSAX outperforms SAX by achieving less approximation error of up to 2 to 3 times. The difference in how much the approximation error is reduced in the case of IoTSAX according to the threshold values, is more evident with data fluctuations and bigger window sizes as it can be seen in Healthcare's use case.

Approximate Reasoning

Different studies have shown that the performance of the embedding models is dependent on the different parameters and the corpus that they have been trained on. Therefore, 11 embedding models are used to check their performance in finding the most conceptually relevant property existing in the rules. These models range from the ones assigning a single vector to each word (e.g. Word2Vec [182], FastText [146]) to the ones giving a vector for each sense/individual meaning of a word (e.g. MSSG [204], DeConf [205], and ConceptNet [134]). Specifically, the pre-trained models are Word2Vec GoogleNews, Vectors, and Phrase[12], Word2Vec Wikipedia No Lemmatisation/Lemmatisation and FastText Wikipedia No Lemmatisation/Lemmatisation[13] [206], MSSG Wikipedia[14], DeConf[15], and ConceptNet[16]. For MSSG (best model is vectors.NP-MSSG.50D.30K) and DeConf only the first sense for each word is taken into account (MSSG Wikipedia First Sense and DeConf First Sense, respectively). MSSG also is examined for each word's global vector regarding all its senses (MSSG Wikipedia Global).

Table 5.10: Average F-score of each property at finding the correct top-1 conceptually similar property existing in the reasoning rules

Approach	**Memory**	**Average F-score of all properties for each use case**	
		Healthcare	**Smart Cities**
WORD2VEC GOOGLENEWS	3.35 GB	0.697	0.740
WORD2VEC VECTORS	54.39 MB	0.677	0.789
WORD2VEC PHRASE	519.81 MB	0.606	0.722
WORD2VEC WIKIPEDIA NO LEMMATISATION	346.60 MB	0.688	0.723
WORD2VEC WIKIPEDIA LEMMATISATION	339.47 MB	0.656	0.723
FASTTEXT WIKIPEDIA NO LEMMATISATION	346.54 MB	**0.750**	0.733
FASTTEXT WIKIPEDIA LEMMATISATION	313.49 MB	0.667	0.751
MSSG WIKIPEDIA GLOBAL	363 MB	0.548	0.710
MSSG WIKIPEDIA FIRST SENSE	363 MB	0.742	0.771
DECONF FIRST SENSE	**236.91 MB**	0.727	**0.850**
CONCEPTNET	1.09 GB (English)	**0.879**	**0.892**

The correctness evaluation is done offline and the F-score results are illustrated in Table 5.10. It is observed that the F-scores are higher in the case of Smart Cities

[12]`https://code.google.com/archive/p/word2vec/`
[13]`http://vectors.nlpl.eu/repository/`
[14]`http://iesl.cs.umass.edu/downloads/vectors/release.tar.gz`
[15]`https://pilehvar.github.io/deconf/`
[16]`https://github.com/commonsense/conceptnet-numberbatch`

compared to the stricter terminologies in the case of Healthcare. Nevertheless, most models behave well with FastText Wikipedia No Lemmatisation and ConceptNet being the best in Healthcare's use case, and DeConf and ConceptNet being the best in the use case of Smart Cities. An unexpected result is the third-best F-score provided by Word2Vec Vectors for Smart Cities, given the fact that it is the simplest and smallest in memory model among all. It is also worth mentioning that lemmatisation models behave worse or similar than no lemmatisation ones and this could be attributed to possible Part-of-Speech tagging errors. The worst results for both cases are given by MSSG Wikipedia Global, making clearer the message that using senses can provide better results. Overall, the superiority of the ConceptNet and DeConf models is observed that suggest that when concept hierarchy is used to learn sense vectors for multi-sense words, then, the words can be better represented depending on the domain and use case. It should also be observed that these models behave very well given the restrictive top-1 similarity between the sense vectors and the targeted properties existing in the reasoning rules. Nevertheless, the memory regarding each model should be taken into account with all models, apart from Word2Vec GoogleNews (3.35 GB) and ConceptNet (English-version 1.09 GB), ranging between the small values of 54.39 MB to 519.81 MB, making DeConf the best model regarding the trade-off between memory and correctness.

5.6 Summary

Existing approaches have tried to resolve requirements regarding usability, user expressibility, data expressiveness, user effectiveness, data effectiveness, and efficiency. Nevertheless, no existing approach covers all requirements well. Time series approximation approaches create succinct and expressive representations (spatio-temporal correlations) either through complex and inefficient ways or through SAX, which is a simple and computationally much less costly symbolic approximation. However, SAX is limited in terms of adaptability to IoT environments due to its fixed/non-adaptive segment number and alphabet size parameters that need to be manually tuned for best results. Also, SAX does not detect sharp fluctuations or extreme/abnormal values, it does not follow the data's true distribution, and it normalises the raw values of the data, losing the original information. Approaches that optimise SAX are either static methodologies that cannot be directly applied to IoT environments, or supervised methodologies, or domain-dependent, or partially optimise the limitations of SAX, without any work addressing sharp fluctuations. Conceptual and contextual awareness approaches create high-level abstractions either through semantic rule engines and languages or machine learning and approximation methodologies. However, they rely on or extend ontologies associated with the aforementioned limitations, some use non-standardised SPARQL-like queries, some are complex and inefficient for IoT environments, and they use domain-specific and representationally-coupled reasoning rules as well as voluminous rules that are inefficient, memory-heavy, and difficult to be updated and maintained. Also, usability is not addressed by these approaches like in the case of Pub/Sub approaches that perform approximate semantic matching of medium usability; nevertheless, the latter do not address high-level abstraction and summarisation.

In this chapter, the IoTSAX approach is proposed, which is a novel dynamic abstractive summarisation methodology for heterogeneous numerical entity graph streams. The approach consists of two phases: 1) an enhanced SAX approximation by providing dynamic segment points as well as alphabet size for symbolic representations that follow data fluctuations, and 2) a novel approximate rule-based reasoning that is based on data approximation interpretation and embedding models. The approach creates quad-based abstractive entity summaries (notifications) for representationally-decoupled and abstractive-aware subscriptions. It is evaluated for two real-world use cases; Healthcare and Smart Cities, along with a proposed ground truth, and baselines including typical Pub/Sub and SAX. The results include 2 to 3 times better approximation error compared to SAX, F-score of up to 0.87 for approximate rule-based reasoning with models that use concept hierarchy to learn sense vectors for multi-sense words being superior, 2 to 3 times slower in end-to-end latency compared to typical Pub/Sub, similar throughput ranging from 13.231 to 97.393 events/second, up to 98% message reduction, and compression space-saving percentage ranging from 71.75% to 94.99% depending on data redundancy. IoTSAX also abstracts the users from defining the best parameters (unlike SAX) by only expecting them to define the level of strictness of the compression, depending on an appropriate threshold.

Chapter 6

Dynamic Entity Summarisation of Enriched Data

6.1 Introduction

This chapter is dedicated to the PoSSUM approach, which has been published in Pavlopoulou and Curry [72] (in press).

The PoSSUM approach is a novel dynamic extractive and abstractive diverse summarisation methodology that is based on embedding-based density clustering and contextual-based top-k ranking. It involves heterogeneous datasets of triples with object-type and data-type (numerical) properties or real-time numerical data enriched with information from knowledge bases (e.g. DBpedia) that may contain redundancy in the attributes and/or values due to duplicates or conceptual similarity. Therefore, it addresses both motivational scenarios regarding Healthcare and Smart Cities, described in Chapter 2. The approach applies to domain-agnostic environments that demand scalability and timeliness, are of limited resources, and have no standardisation in data representations (i.e. representationally-coupled). Although its main application is extractive and abstractive entity summarisation, the approach can also be used for transforming triples to semantic vectors, conceptual clustering, triple ranking, and reasoning.

All challenges, analysed in Chapter 2, are addressed with solutions, described in Chapter 3. Specifically, the approach tackles data challenges regarding heterogeneity, redundancy caused by duplication and conceptual similarity, and data enrichment. Data with different representations and data types (numerical and Linked Data) is observed, combined, and grouped in conceptual clusters if they are conceptually and contextually related. A ranking, then, decides which data from each cluster will be included in a user notification to provide an enriched diverse set of information (extractive and abstractive summary) that does not overwhelm the subscribers and the processing system. In the meantime, users need only provide representationally-decoupled subscriptions through the proposed subscription model. User challenges are also tackled regarding high-level interpretation of the numerical sensor data by proposing an approximation approach along with reasoning to provide meaningful inferences to the users, user-friendly data representation by providing graph-based notifications, and non-technical users' facilitation by demanding only diversity-aware subscriptions that do not include a-priori data, semantic or schematic information, or return partial or abstract user information. Finally, all system challenges regarding scalability, timeliness and resource constraints are addressed since a Publish/Subscribe system (Pub/Sub) is proposed along with windowing policies and summarisation that can cover dynamic sources, data continuity, real-time processing requirements, and possible network overhead. An emphasis (in green) on which components are analysed in this chapter from the overall proposed *Entity-centric Publish/Subscribe Summarisation System* along with their associated research questions is illustrated in Fig. 6.1.

The contributions of this chapter are the following:

- A user-friendly entity-centric subscription model that allows subscribers to express in a simple way whether they need to receive a diverse extractive and abstractive entity summary with top-k diverse filtered information by also providing the desired window traits (type and size).
- PoSSUM, a novel dynamic extractive and abstractive diverse summarisation methodology for heterogeneous numerical and Linked Data entity graph streams, alike, that is based on: 1) a novel embedding and density-based conceptual clustering that is parameter-free, partly incremental, and can be used for general streaming applications, and 2) a contextual-based top-k ranking of the conceptual clusters related to user query importance, informativeness, and diversity that create graph-based summary notifications of entities.
- A novel evaluation methodology including:

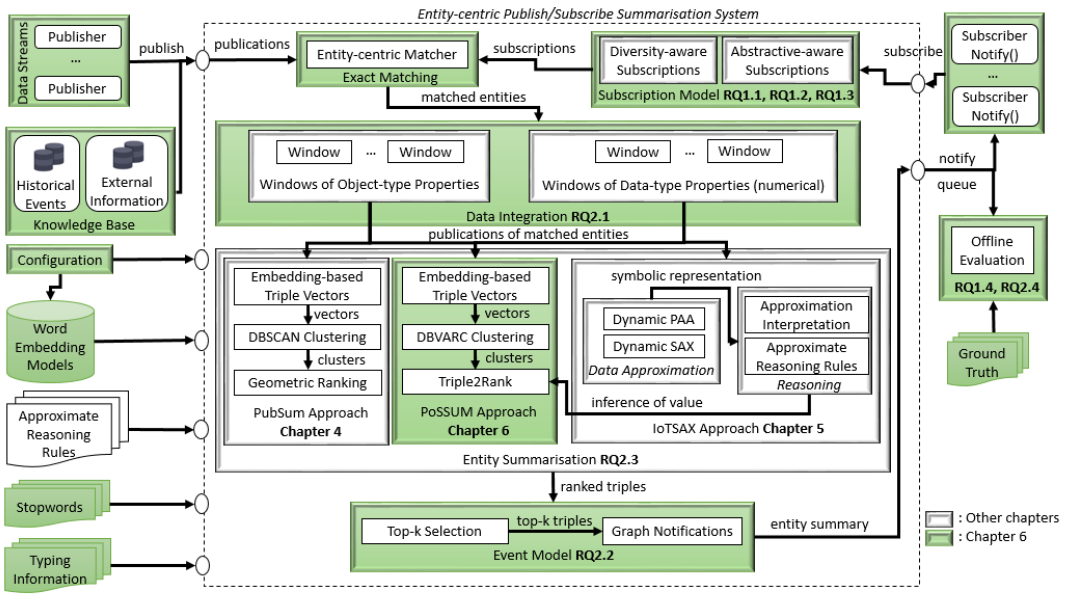


Figure 6.1: The components of the Entity-centric Publish/Subscribe Summarisation System analysed in this chapter.

– The construction of an evaluation dataset based on real-world sensor data related to the domains of Healthcare and Smart Cities as well as Linked Data deriving from DBpedia related to entities like people, things, and places with characteristics involving heterogeneity in data types, semantics, concepts, and contexts.

– The construction of two ground truths (relevance and conceptual) based on semantically extending original datasets with the use of thesauri and ontologies, and observing the contextual coherence, direct word links, and sub-categories in taxonomies.

– The identification of a user-defined ground truth that contains ideal summaries from human judges.

– Identification of FACES [50], a static diverse entity summarisation methodology, typical Pub/Sub, and proposed PubSum approach (Chapter 4) as baselines, and adaptation of FACES to the proposed dynamic Entity-centric Publish/Subscribe Summarisation System (Chapter 4).

– Identification and formulation of a range of evaluation metrics related to the agreement, quality, latency, size reduction, memory footprint, and throughput with novel metrics like diversity consensus, conceptual clustering F-score, and concept-based redundancy-aware F-score.

- An extensive evaluation comparison between the PoSSUM approach and baselines by examining user effectiveness, data effectiveness, and system efficiency. The results include:
  - User effectiveness: data diversity user desire up to 80%, and best summary quality for more than half of the entities by a large margin.
  - Data effectiveness: best conceptual clustering F-score ranging from 0.69 to 0.83 and redundancy-aware F-score of up to 0.95.
  - Efficiency: up to 99% message reduction, half or even less clustering processing time and significantly faster scoring ranging from 1.933ms to 25.733ms and 4.567ms to 18.267ms, respectively, comparable end-to-end latency and throughput ranging from 171,317ms to 310,203ms and 25 to 871.132 events/second, respectively, and occupies a third of the memory.

The rest of the chapter is as follows: In Section 6.2 the necessary background is given related to clustering methodologies, whereas Section 6.3 contains related work and how it maps to the challenges, requirements, and research questions of the thesis. Section 6.4 describes the proposed Entity-centric Publish/Subscribe Summarisation System, where its architecture and details of the PoSSUM approach are provided. Section 6.5 analyses the evaluation methodology as well as the results of the experiments, while Section 6.6 concludes and summarises the chapter.

6.2 Background

This section contains the necessary background for the rest of the chapter. The concept of micro-clustering is analysed along with its strengths regarding streaming data as well as the density-based stream clustering along with its challenges.

6.2.1 The Power of Micro-clustering

Micro-clustering [41, 207] is a clustering methodology that can work efficiently and effectively in the clustering of data streams. Data streams are characterised by different traits compared to static data (e.g. real-time, temporal correlations, heterogeneity, concept-drift etc.); therefore, a more specialised clustering approach would be deemed more appropriate compared to the most popular ones like K-means. Therefore, micro-clustering is split into two phases:

Online (Micro-clustering): It involves on-the-fly clustering as the data is generated that results in micro-clusters. Micro-clusters are tuples that instead of storing the original data points, they maintain their statistical information. For example, this information could involve the sum and sum of the squares of the data values, the sum of the timestamps, and the number of data points. This information can be updated incrementally as new data points are generated and are either merged in already defined micro-clusters or put in a new micro-cluster. The goal is to keep a statistical summary of the data at a sufficiently high level of temporal and spatial granularity.

Offline (Macro-clustering): It involves offline clustering by refining the resulting micro-clusters based on specific criteria or user interests. This results in macro-clusters, which instead of using the original data points, they use their statistical summary in order to efficiently form the final clusters presented to the user.

Micro-clustering is superior to popular clustering methodologies like K-means, because it is specifically designed to cater for data streams' characteristics. This means that the number of clusters is defined automatically based on the temporal observations of the data, the online phase is incremental, leading to efficient processing, the cluster centroids are dynamic according to the formulation of clusters in specific time frames, and along with the offline phase, they contribute to clusters of high quality and user effectiveness based on user criteria.

6.2.2 Density-based Stream Clustering and Its Challenges

Micro-clustering methodologies could be used to perform density-based stream clustering, where density profiles of the data are created. Nevertheless, there is a challenge involved. Specifically, micro-clusters are the initial clusters based on the density estimations calculated in real-time. These may include clusters that eventually will have regions of low density or data points that may lie on the edge of a cluster and are closer to other data points on the edge of other adjacent clusters. This phenomenon begs the question of whether: 1) the data points of low density should be split from their original cluster and form their own one(s), and 2) whether data points that lie on the edge of different clusters, which are densely intra-connected but loosely inter-connected, should be split from their original cluster and merged to a common new one. This is depicted in Fig. 6.2 for two micro-clusters, where the stars are their centroids.

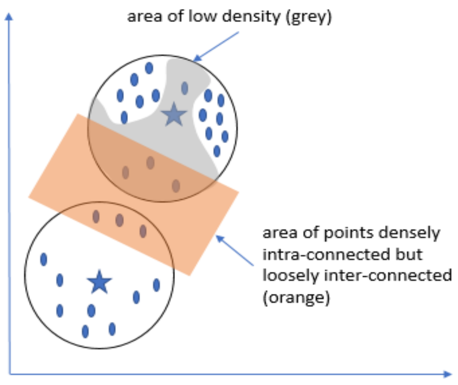

area of low density (grey)
area of points densely intra-connected but loosely inter-connected (orange)

Figure 6.2: An example of the challenge of micro-clustering.

This phenomenon is dependent on the distributions of the data points observed in each micro-cluster as illustrated in Fig. 6.3. These distributions are not known a-priori and assumptions of them following a specific one (e.g. Gaussian distribution around a centroid), are erroneous. In the example, the distributions of the data points in the micro-clusters of Fig. 6.3 (left), and the distributions of each micro-cluster based on the points' distance to its corresponding centroid (right), are depicted. The area in orange constitutes the peak (most dense/popular area) of a distribution, followed by yellow, green and, finally, blue that corresponds to the lowest distribution. On the left, it is observed that the data points do not follow an a-priori distribution, whereas, on the right, it is seen that the further away points are from the peak of a distribution, the higher their variance. Metrics, like variance and distance measures, can observe the deviation or disperse of data points from the centroid of a micro-cluster. The points that are characterised by high variance should be examined again on whether they correctly belong to a cluster during the next phase of micro-clustering, which refines the micro-clusters.

In this chapter's work, a novel embedding and density-based conceptual clustering that is parameter-free is proposed, which is based on the aforementioned concept, that is finding initial density profiles of data streams, and refining them for elements that are densely intra-connected and loosely inter-connected based on a variance-like concept.

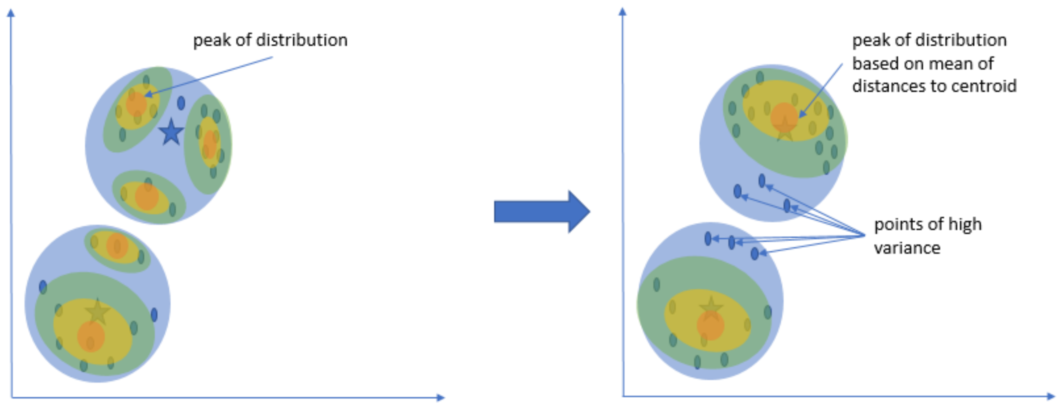


Figure 6.3: An example of how distributions can detect data points of high variance within micro-clusters.

6.3 Related Work

This section contains the related work, which is split into the following categories:

- **Diverse Entity Summarisation**: It involves summarisation approaches of static Resource Description Framework (RDF) graphs based on user importance, diversity, relevance, and popularity.
- **Clustering**: It involves either conceptual clustering or stream clustering approaches that group elements according to a concept or on-the-fly, respectively.
- **Diversity in Pub/Sub**: It involves Pub/Sub systems that produce top-k diverse notifications.
- **Approximate Semantic Matching in Pub/Sub**: It involves Pub/Sub systems that introduce semantic decoupling in the system.
- **Semantic Engines in Pub/Sub**: It involves Pub/Sub systems that semantically enrich the events.
- **Other Approaches**: It includes graph summarisation, graph approximation, subscription summarisation in Pub/Sub, triple ranking, stream approximation, and conceptual and contextual awareness of raw numerical data.

The related work is analysed and mapped according to the aforementioned requirements:

- **R1: Usability**: high usability system independent of representational coupling, query language expertise, system knowledge, bias, and background knowledge.
- **R2: User Expressibility**: users understanding their data needs and expressing them by simple subscriptions of high usability with minimal configuration settings.
- **R3: Data Expressiveness**: domain-agnostic system tackling interoperability and heterogeneity by providing rich notifications that contain conceptual and contextual diversity or high-level abstractions.
- **R4: User Effectiveness**: notifications of high quality according to the users' needs.
- **R5: Data Effectiveness**: redundancy-aware and expressive notifications of high quality according to the wide range of concepts and contexts.
- **R6: Efficiency**: efficient system in terms of memory, processing time, throughput, and scalability.

6.3.1 Diverse Entity Summarisation

This category has been analysed in Chapter 4. It contains approaches that: 1) are offline so they cannot be directly applied to Internet of Things (IoT) environments (R6), 2) are either unsupervised that rely on thesauri and ontologies with a plethora of limitations or supervised, inefficient methodologies that rely on superior embedding models (R3-R5), 3) do not address temporal numerical sensor data (R3), 4) contain manually-tuned parameters (R4, R5), 5) do not address abstractive summaries (R3), and 6) are evaluated only on user effectiveness based on highly subjective metrics (R4). PubSum is the first approach to address dynamic diverse entity summarisation and it is analysed below.

PubSum Approach

Description: This is this thesis' proposed work that addresses dynamic diverse entity summarisation and it is analysed in Chapter 4. It uses a combination of windowing policies, Word2Vec [182] embedding models, the Density-based Spatial Clustering of Applications with Noise (DBSCAN) [141] algorithm for conceptual clustering, and a ranking based on cosine and Euclidean similarity metrics. A representationally-decoupled subscription model is also proposed.

Review: The work addresses challenges related to heterogeneity and redundancy caused by duplication and conceptual similarity (R3) as data with different representations is grouped in conceptual clusters and a ranking decides the final diverse set of information within the summary. At the same time, the users are obliged to provide only representationally-decoupled subscriptions, making the approach highly usable and able to facilitate non-technical users (R1). Also, the final summary is presented as an RDF graph, which is a user-friendly data representation that contains conceptually and contextually rich information (R3). Furthermore, scalability, timeliness and resource constraints are addressed since a Pub/Sub system is proposed along with windowing policies and summarisation (R6). Finally, the approach relies on word embedding models that are more flexible and superior than thesauri and ontologies (R1-R6).

On the other hand, challenges regarding data enrichment and high-level interpretation are out of scope (R3). Also, the conceptual clustering is based on the DBSCAN algorithm, which is a static parameter-tuned approach that occurs in batch for each window (R6). Another issue is that the user effectiveness (R4) of the approach is not so good and the data effectiveness (R5) as well as efficiency (R6) could further be improved. Finally, the approach does not address temporal numerical sensor data, and constructs only extractive summaries without considering high-level abstractions (R3).

6.3.2 Clustering

Clustering is a popular unsupervised learning methodology with several approaches [208] ranging from the popular K-means to density-based algorithms. like DBSCAN, to hierarchical algorithms, like Agglomerative Hierarchical Clustering. Nevertheless, all these approaches are related to static data and they could not be directly applied to streaming data like in the case of IoT. The most related subcategories of clustering approaches to the proposed work are conceptual clustering and stream clustering that are analysed below.

Conceptual Clustering

Description: This type of clustering involves placing instances (attribute-value pairs) into disjoint clusters that each corresponds to a concept. The most influential works are the initial ones like Witt [209] and Cobweb (which FACES is based on) that create conceptual clusters based on the co-occurrence between pairs of features, or how discriminative one category/concept is from others based on common features. More approaches are discussed in Perez et al. [210].

Review: This type of work could address challenges related to heterogeneity and redundancy caused by duplication and conceptual similarity (R3) as data with different representations is grouped in conceptual clusters and the user is expected to realise which concept applies to which cluster. On the other hand, this work is representationally-coupled as it involves attribute-value pairs and the clusters are constructed based on the exact semantics in values among different features (R3-R5). Also, all these approaches are offline with most of them being computationally expensive and memory-heavy so they cannot be directly applied to IoT environments (R6). One of the biggest limitations is that most approaches contain many parameters, including a pre-defined number of clusters, to be manually tuned for best results, which highly affects the data expressiveness (R3) and data effectiveness (R5). Another issue is that some approaches need corpora or knowledge bases to be trained on, or they may rely on user input/feedback or thesauri with aforementioned limitations (R3-R5). Supervised methodologies can prove inefficient, which could be escalated even more by the fact that some approaches are not even incremental (R6). Another characteristic is that usually the user effectiveness (R4) is poorly covered in the evaluation. Finally, usability (R1), user expressibility (R2), rich entity-centric graph data, and summarisation (R3) are out of scope.

Stream Clustering

Description: This type of clustering involves the on-the-fly clustering of streams. The most important approaches are CluStream [211], DenStream [212], ClusTree [213], and DBSTREAM [214]. CluStream creates micro-clusters in an online manner and performs K-means offline for refining the clusters (macro-clusters). DenStream extends DBSCAN to create micro-clusters in a streaming fashion and creates final clusters in an offline manner. It involves two offline stages, one where the original DBSCAN is applied to create an initial set of clusters, and another one, where the final clusters are defined based on density. ClusTree creates a hierarchical tree of micro-clusters inspired by the structures of R-trees based on data distributions and Euclidean distances. Its main characteristic is that it does anytime clustering in different time intervals, and as DenStream, it has a non-optimised initialisation phase. DBSTREAM extends the micro-clustering structure by incorporating the density between the area of two micro-clusters to be used in the offline re-clustering phase. More approaches and a comparison among them are discussed in Carnein et al. [215].

Review: This type of work could address challenges related to scalability, and timeliness and resource constraints in streaming environments (R6). Also, these approaches are independent of models, unsupervised, and could apply to any domain. On the other hand, when it comes to complex IoT environments these approaches pose many limitations. Specifically, the offline initialisation phase and final clustering are computationally inefficient (R6), the approaches mostly apply to numerical data and not rich entity-centric graphs (R3), the assumption by some approaches that the data follows a Gaussian distribution might not be always the case, the creation of only spherical clusters by some approaches (excluding DenStream and DBSTREAM) is rather limited, and a high number of manually-tuned parameters (e.g. threshold values in DenStream and DBSTREAM, the amount of data stored in each tree node in ClusTree etc.) needs to be pre-determined including, by some, the number of clusters (excluding DenStream and DBSTREAM). Finally, usability (R1), user expressibility (R2), data expressiveness (R3), effectiveness (R4, R5), and summarisation (R3) are out of scope.

6.3.3 Diversity in Pub/Sub

This category has been analysed in Chapter 4. It contains approaches that: 1) contain either representationally-coupled subscriptions or abstract keyword-based ones (R1, R2), 2) address redundancy only in the sense of duplication or commonality among attributes/terms (R3), 3) do not apply to rich entity-centric graph data (R3), 4) summarisation is out of scope (R3), but a top-k ranking is performed (R4-R6), and 5) some are poorly evaluated (R4-R6) or evaluated only on user effectiveness (R4).

6.3.4 Approximate Semantic Matching in Pub/Sub

This category has been analysed in Chapter 4. It contains approaches that: 1) although they contain semantically-decoupled subscriptions (R2), they are still in the form of attribute-value pairs that have medium usability (R1), 2) do not address redundancy of any form (R3), 3) most do not apply to rich entity-centric graph data (R3), 4) summarisation is out of scope (R3), 5) some rely on taxonomies, thesauri, or ontologies with a plethora of limitations (R1, R2, R4, R6), and 6) some contain manually-tuned parameters (R4).

6.3.5 Semantic Engines in Pub/Sub

This category has been analysed in Chapter 4. It contains approaches that: 1) contain either representationally-coupled subscriptions (R2) or abstract keyword-based ones (R1), 2) do not address redundancy of any form (R3), 3) summarisation is out of scope (R3), 4) rely on or build ontologies with a plethora of limitations (R1, R2, R4, R6), 5) some rely on supervised learning by domain experts (R4, R6), and 6) do not evaluate effectiveness (R4).

6.3.6 Other Approaches

Other approaches related to graph summarisation, graph approximation, subscription summarisation in Pub/Sub, and triple ranking are analysed in Chapter 4. Approaches related to stream approximation, and conceptual and contextual awareness of raw numerical data are analysed in Chapter 5.

6.3.7 Comparison

In conclusion, no existing work covers all of the requirements well. Comparison among the different categories in relation to the features and the requirements is shown in Table 6.1 and Table 6.2, respectively.

Table 6.1: Features as addressed by related work

Approach	**Query Type**	**Data Type**	**Expressiveness Type**	**Summary Type**	**Model Dependency**	**Unsupervised Learning**	**Domain Agnostic**	**Stream-ing/IoT**	**Lack of Parameters**
Diverse Entity Summarisation									
[50–52, 114, 145, 147–151]	keywords	RDF graphs (object-type or (string) literals)	conceptual & contextual	extractive	ontologies & thesauri or embedding models	dependent on approach	dependent on approach		
Conceptual Clustering									
[142, 209, 210]	N/A	attribute-value pairs	conceptual & contextual	N/A	ontologies or thesauri or user feedback or none	dependent on approach	dependent on approach		dependent on approach
Stream Clustering									
[211–215]	N/A	numerical	N/A	N/A		✓	✓	✓	
Diversity in Pub/Sub									
[54–56]	preferential representational coupled subscriptions or keywords	attribute-value pairs or text	de-duplication & lack of commonality	(top-k ranking)		✓	✓	✓	✓
Approximate Semantic Matching in Pub/Sub									
[39, 57, 58, 162–164]	semantic decoupled or fuzzy subscriptions	RDF graphs or attribute-value pairs	N/A	N/A	distributional semantics and taxonomies or thesauri and ontologies	✓	dependent on approach	✓	dependent on approach
Semantic Engines in Pub/Sub									
[59, 60, 165–167]	representational coupled subscriptions or keywords	RDF graphs	N/A	N/A	ontologies	dependent on approach		✓	✓
PubSum	representational decoupled subscriptions	RDF graphs (object-type)	conceptual & contextual	extractive	embedding models	✓	✓	✓	
PoSSUM	representational decoupled subscriptions	RDF graphs (object-type & data-type sensor data)	conceptual, contextual & high-level interpretations	extractive & abstractive	embedding models	✓	✓	✓	✓

Table 6.2: Requirements as addressed by related work

Approach	**R1 - Usability**	**R2 - User Expressibility**	**R3 - Data Expressiveness**	**R4 - User Effectiveness**	**R5 - Data Effectiveness**	**R6 - Efficiency**
Diverse Entity Summarisation						
FACES [50]	++	N/A	+	++	N/E	N/A
ES-LDAext [51]	++	N/A	++	++	N/E	N/A
DeepLENS [52]	++	N/A	+	++	N/E	N/A
PubSum	++	++	++	-+	+	+
Conceptual Clustering						
[142, 209, 210]	N/A	N/A	+/++	-	-+/+	N/A
Stream Clustering						
[211–215]	N/A	N/A	N/A	N/A	N/A	-+
Diversity in Pub/Sub						
PrefSIENA [54]	-+	-+	-+	+	+	++
Chen et al. [55]	++	-	-+	+	N/E	-+
Hmedeh et al. [56]	++	-	-+	+	-	-+
Approximate Semantic Matching in Pub/Sub						
Hasan et al. [57]	+	-+	N/A	+	N/A	++
Hasan et al. [39]	+	+	N/A	++	N/A	++
Alhakbani et al. [58]	+	+	N/A	++	N/A	++
Semantic Engines in Pub/Sub						
G-TOPSS [59]	-+	-+	N/A	N/E	N/A	++
Esposito et al. [60]	++	-	N/A	N/E	N/A	-+

++ the requirement dimension is well covered
+ the requirement dimension is partially covered with positive results
−+ there is an attempt to address the requirement dimension but the solution is not effective
− the requirement dimension is poorly covered
−− the requirement dimension is very poorly covered
N/A the requirement dimension is not addressed or the focus of the research
N/E the requirement dimension is not evaluated

6.4 Entity-centric Publish/Subscribe Summarisation System

An extension of Pub/Sub is proposed, in this chapter, that can overcome all of the aforementioned issues, limitations, and gaps; the Entity-centric Publish/Subscribe Summarisation System that is based on a novel embedding and density-based conceptual clustering, and a contextual-based top-k ranking (PoSSUM approach) that create graph-based diverse entity summaries (notifications) for diversity-aware subscriptions. In this section, an overview of the architecture of the proposed system as well as details of the PoSSUM approach are described.

6.4.1 Architecture

The architecture of the approach is illustrated in Fig. 6.4. A collection of publishers generate publications concerning different entities with one of them being Usain_Bolt. Specifically, one publisher could generate external information from static sources regarding an entity, whereas other publishers extract information concerning one's measurements regarding respiration rate and heart rate, among others. A subscriber generates a *Diversity-aware Subscription*, where one is interested in an extractive and abstractive summary of the entity Usain_Bolt, that is the top-5 diverse entity information, deriving from the analysis of data taken from count tumbling windows of total size 100 (only count tumbling windows are supported for this implementation). All publications and subscriptions enter the *Entity-centric Publish/Subscribe Summarisation System*, where they are stored and managed. In the example, the triples are the following: $t_{rr}1$: <Usain_Bolt> <respirationRate> "18.700", $t_{rr}2$: <Usain_Bolt> <respirationRate> "18.600", $t_{hr}1$: <Usain_Bolt> <heartRate> "96.400", $t_{hr}2$: <Usain_Bolt> <heartRate> "95.800", $t_{e1}1$: <Usain_Bolt> <birthPlace> <Trelawny_Parish,_Jamaica>, $t_{e1}2$: <Usain_Bolt> <birthPlace> <Trelawny_Parish>. The available triple information is presented without any schematic prefixes for visualisation purposes.

The stored publications and subscriptions are examined by the *Entity-centric Matcher*. If there is a match between the entity requested by the subscribers and the entity to which publications refer, like in this example Usain_Bolt, then, a match occurs. After a match, two kinds of listeners are connected to the system, one for the publishers and one for the subscribers in order to observe whenever

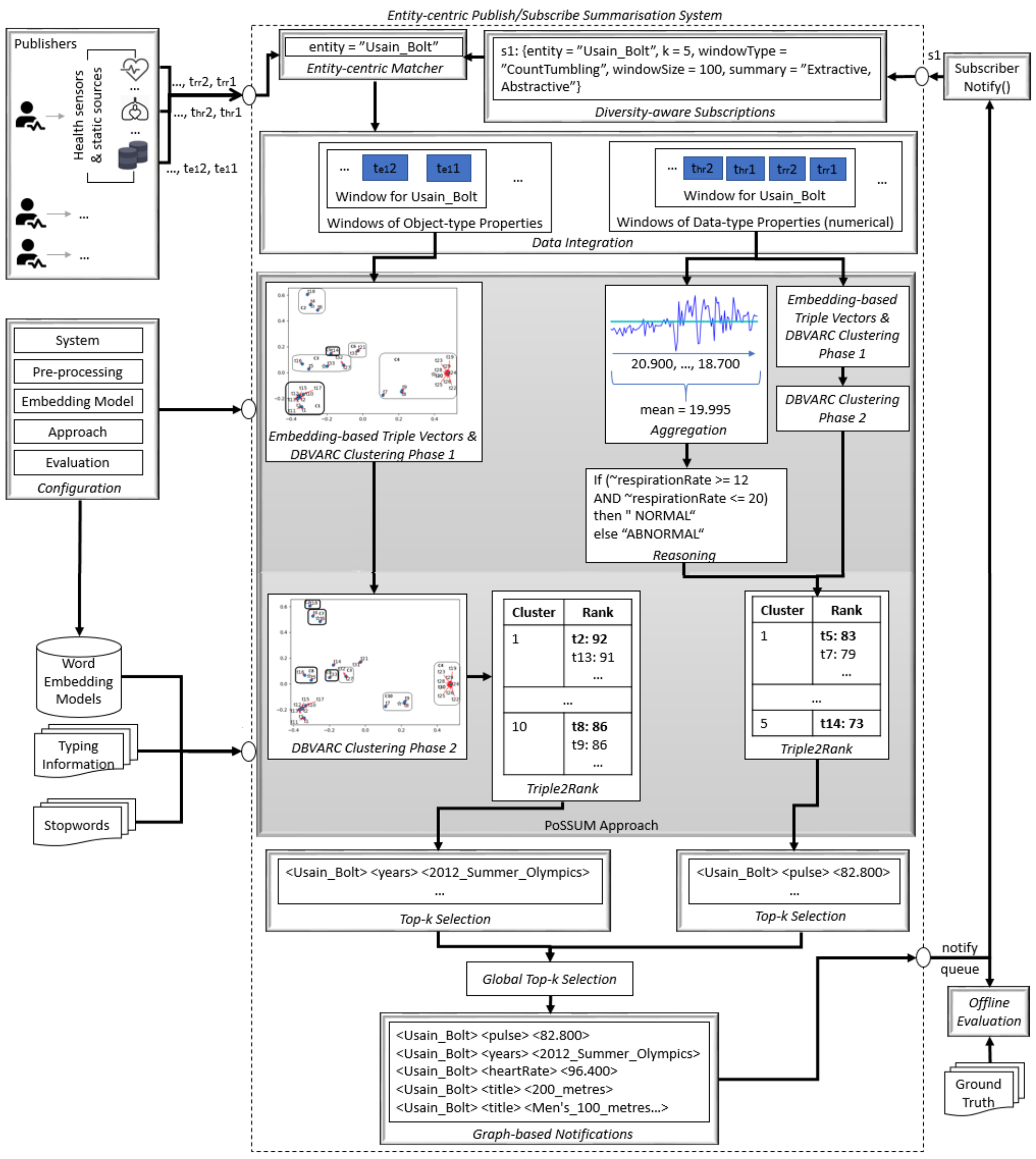

Entity-centric Publish/Subscribe Summarisation System
Publishers
Health sensors & static sources
..., trr2, trr1
..., thr2, thr1
..., te12, te11
entity = "Usain_Bolt"
Entity-centric Matcher
s1: {entity = "Usain_Bolt", k = 5, windowType = "CountTumbling", windowSize = 100, summary = "Extractive, Abstractive"}
Diversity-aware Subscriptions
s1
Subscriber Notify()
Window for Usain_Bolt
Windows of Object-type Properties
Windows of Data-type Properties (numerical)
Data Integration
Embedding-based Triple Vectors & DBVARC Clustering Phase 1
20.900, ..., 18.700
mean = 19.995
Aggregation
Embedding-based Triple Vectors & DBVARC Clustering Phase 1
DBVARC Clustering Phase 2
If (~respirationRate >= 12 AND ~respirationRate <= 20) then " NORMAL" else "ABNORMAL"
Reasoning
System
Pre-processing
Embedding Model
Approach
Evaluation
Configuration
Word Embedding Models
Typing Information
Stopwords
DBVARC Clustering Phase 2
Cluster | Rank
1 | t2: 92, t13: 91
10 | t8: 86, t9: 86
Triple2Rank
Cluster | Rank
1 | t5: 83, t7: 79
5 | t14: 73
Triple2Rank
PoSSUM Approach
<Usain_Bolt> <years> <2012_Summer_Olympics>
Top-k Selection
<Usain_Bolt> <pulse> <82.800>
Top-k Selection
Global Top-k Selection
<Usain_Bolt> <pulse> <82.800>
<Usain_Bolt> <years> <2012_Summer_Olympics>
<Usain_Bolt> <heartRate> <96.400>
<Usain_Bolt> <title> <200_metres>
<Usain_Bolt> <title> <Men's_100_metres...>
Graph-based Notifications
notify
queue
Offline Evaluation
Ground Truth

Figure 6.4: Architecture of the Entity-centric Publish/Subscribe Summarisation System.

new elements have been generated related to the entity for future processing. Afterwards, two windows are created based on the selected windowing policy and the data type; one for object-type properties, and one for data-type (numerical) properties. The publications of the matched entity Usain_Bolt will be integrated as they are generated into the corresponding window depending on their data type (*Data Integration*). Other windows may be created later on based on the matched entities. As elements get in each window a trigger is activated and they are incrementally processed. First, a counter is calculating all elements currently in each window, then, the elements are processed by the *Embedding-based Triple Vectors* methodology. The Embedding-based Triple Vectors approach is incrementally pre-processing each publication that enters a window by extracting its triple information and transforming it into a vector by using a pre-selected word embedding model. In the example, the vector space of the window triples of the entity Usain_Bolt is depicted after their transformation for the object-type properties case. After a triple has been transformed into a vector, an initial incremental clustering is taking place (*DBVARC Clustering Phase 1*), where the element will create a new cluster or will be merged into an existing one based on its proximity, in the embedding-based semantic space, with previous elements within a window. In the example, 6 initial clusters are depicted for the object-type properties window. In the case of the window containing the data-type (numerical) properties, an additional incremental process is taking place. Specifically, the object or value of the triple (publication's payload) is extracted and *Aggregation* takes place, which incrementally calculates an average value of the elements so far, within the window, that relate to the same property. This process is shown for the respiration rate property, in the example. This incremental process ends once each window of the two reaches its full capacity, that is their total counter is equal to the subscription's windowSize, which is 100. Afterwards, *DBVARC Clustering Phase 2* takes place, which involves refining the initial clusters derived from DBVARC Clustering Phase 1, and resulting in the final conceptual clusters. In the example, the 6 initial clusters turned into 10, in total, after the refinement process. In the case of the window containing the data-type (numerical) properties, an additional *Reasoning* process is taking place, where a high-level inference is extracted by the reasoning rules based on what the aggregated value of the attributes/predicates really means. In the example, it is observed that the values so far regarding respiration rate led to the inference of NORMAL. Then, the conceptual clusters are ranked by *Triple2Rank* based on the importance and diversity of the triples in each cluster. In the example, the most important triple for each cluster is depicted in bold. Once the ranking has taken place, the window is evicted (cleared) according to the windowing policy and awaits the new elements. The notification process is, then, triggered that checks the state of all subscriptions in order to notify all interested parties. The ranked elements of each window undergo *Top-k Selection* depending on the subscription's k. Once all top-k triples are available, a *Global*

Top-k Selection occurs to pick a diverse set of the most important triples from both windows. In the example, the subscriber will be notified with the top-5 most diverse and important triples of the entity Usain_Bolt. The extractive and abstractive summarisation process is, then, completed and the newly constructed timestamped *Graph-based Notification* is sent to the subscriber and to a queue that along with necessary metadata will be stored externally in order to be evaluated offline by *Offline Evaluation*. This whole process ends once a pre-defined time duration has been reached. Once the run is completed, the Offline Evaluation is responsible for calculating all the necessary metrics. It should be noted that a more sophisticated abstractive entity summarisation approach could be applied in the system as described in Chapter 5 and presented in the overall system in Chapter 3.

The last component to be analysed is the *Configuration* that is responsible for holding static values of parameters that concern a wide range of components. Specifically, it contains parameters regarding the system (e.g. duration of streaming, windowing policies, and publisher and subscriber policies), pre-processing (e.g. stopwords removal, Part of Speech (POS) tagging, and typing information), word embedding model (e.g. type of model and corpus of model), approach (e.g. type of algorithm/approach like PoSSUM or baselines, PubSum's parameters like DBSCAN parameters as well as distance measure, and FACES-adapted parameters like Cobweb parameters as well as triple store location) and evaluation (e.g. storage of results/notifications, ground truth locations, and metrics to be calculated).

6.4.2 PoSSUM Approach

The main extractive diverse summarisation approach (PoSSUM approach) consists of two phases: 1) conceptual clustering, and 2) contextual-based top-k ranking of triples. The first phase is a combination of the Embedding-based Triple Vectors process and Density-Based VARiance Clustering (DBVARC), a novel parameter-free and partly-incremental conceptual clustering that is based on density and variance, while the second one consists of Triple2Rank, a novel methodology for measuring the importance of a triple within and among all conceptual clusters.

Embedding-based Triple Vectors

The first step of the approach is to turn a triple into an embedding-based vector. The same incremental process is followed as described in Algorithm 8, in Chapter 4. The only difference is that when pre-processing takes place on the property and object of a triple, then, the words are not only concatenated with an underscore but with a dash too. This is dependent on the word embedding model used, which in the proposed approach is ConceptNet[1], and several combinations of words with a dash were observed.

Algorithm 8 Embedding-based Triple Vectors

```
1: function Embedding-based Triple Vectors($tr, word2VecModel, windowElements, stopwords, objectTypes$)
2:     if $\neg endOfWindow$ then
3:         $processedTr \leftarrow schemaExtraction(tr)$
4:         $property \leftarrow attributeOfTriple(processedTr)$
5:         $object \leftarrow valueOfTriple(processedTr)$
6:         if $object\ is\ numerical$ then
7:             $duplicate \leftarrow getDuplicates(property)$
8:             if $duplicate$ then
9:                 $object \leftarrow Aggregator(object)$
10:            end if
11:            $same\ process\ only\ for\ property$
12:        else
13:            $same\ process$
14:        end if
15:        return $statementVectors$          ▷ dynamic list of timestamped triple vectors of window
16:    end if
17: end function
```

Since the data, in this chapter's work, is split into triples with data-type (numerical) properties and object-type properties, the Embedding-based Triple Vectors process is slightly different for each type, and explained in Algorithm 8. In line 6, if a triple's object is numerical, then, its property is checked for duplication among the elements of the window (line 7). If a duplicate is found, then, the object is aggregated incrementally, where its mean value is calculated based on previous window values (lines 8-10). The same process is followed only for the property (line 11). If, on the other hand, a triple has an object-type property, then, the same process is followed for its property and object, alike (lines 12-13).

[1]`https://github.com/commonsense/conceptnet-numberbatch`

DBVARC: Density-Based VARiance Clustering

In this chapter, DBVARC is proposed based on the notion that triples are partitioned based on similarity, resulting in clusters with elements that are similar to their cluster's elements and dissimilar to the other clusters' elements. In the case of DBVARC, this is translated to finding the densely-connected regions in the semantic space derived from the Embedding-based Triple Vectors step.

As aforementioned in Chapter 4, density-based algorithms are more suitable for evolving data streams due to their following characteristics: 1) no pre-defined number of clusters parameter, and 2) no restrictions on the size and the shape of the clusters. Nevertheless, there is no density-based algorithm, even for streaming data, that is completely parameter-free, meaning that tuning needs to take place, which is not possible in streaming environments. DBVARC does not demand any pre-defined parameters and is partly incremental, making it suitable for streaming environments. The algorithm is split into two phases; an incremental phase 1, where initial clusters are created, and a batch phase 2, where clusters are examined for further partitioning if they contain elements that are densely intra-connected and loosely inter-connected. More details are given in Algorithm 9 and Algorithm 10.

Algorithm 9 is the incremental phase 1 of the DBVARC, where each triple is put in an initial cluster. In line 1, the DBVARC function gets as input the $statementVector$, which is the timestamped triple vector derived from the Embedding-based Triple Vectors step. The function aims to create or update micro-clusters ($microClusters$), which are tuples that not only contain the elements of a cluster but also their statistics involving the linear sum of the elements' vectors and the cluster's centroid. Specifically, in line 4, the first micro-cluster is created. Its linear sum of the elements' vectors is calculated in line 14, its centroid is the average of all the elements' vectors in line 15 and its elements contain the vectors in line 16. The created micro-cluster is added to the list of $microClusters$ in line 17. If a new micro-cluster involves only one vector as in the case of line 4, then, the tuple's elements are the vector itself. In line 5, each new $statementVector$ is examined for its closest micro-cluster. Specifically, in line 7, cosine similarity is calculated between the new vector and the centroids of the micro-clusters. The micro-cluster with the highest similarity is picked as the closest one. Then, the highest similarity is compared to a static threshold (line 8). If it is greater than the threshold, the new vector will be merged/added to the closest micro-cluster (line

Algorithm 9 DBVARC Phase 1 - Density-Based VARiance Clustering (Initial Clusters)

```
1: function DBVARC($statementVector$)    ▷ $statementVector$: timestamped triple vector of window
2:     if $\neg endOfWindow$ then    ▷ phase 1 - initial clusters
3:         if $microClusters \leftarrow \emptyset$ then
4:             $createNewCluster(statementVector)$
5:         else
6:             for $microCluster \leftarrow microClusters$ do
7:                 $closestCluster \leftarrow maxCosineSimilarity(microCluster.centroid, statementVector.vector)$
8:             end for
9:             if $maxCosineSimilarity > 0.5$ then
10:                 $updateCluster(closestCluster, statementVector, addition)$
11:             else
12:                 $createNewCluster(statementVector)$
13:             end if
14:         end if
15:     end if
16:     return $microClusters$    ▷ initial set of clusters
17: end function
18: function createNewCluster($statementVectors$)
19:     $sumOfElements \leftarrow \sum statementVectors.vector$
20:     $centroid \leftarrow \frac{sumOfElements}{sizeOfStatementVectors}$
21:     $clusterElements \leftarrow statementVectors$
22:     $microClusters \leftarrow MicroClusterTuple(centroid, sumOfElements, clusterElements)$
23: end function
24: function updateCluster($cluster, statementVector, status$)
25:     if $status \leftarrow addition$ then
26:         $sumOfElements \leftarrow cluster.sumOfElements + statementVector.vector$
27:         $clusterElements \leftarrow cluster.clusterElements.add(statementVector)$
28:     end if
29:     if $status \leftarrow deletion$ then
30:         $sumOfElements \leftarrow cluster.sumOfElements - statementVector.vector$
31:         $clusterElements \leftarrow cluster.clusterElements.remove(statementVector)$
32:     end if
33:     $centroid \leftarrow \frac{sumOfElements}{sizeOfClusterElements}$
34:     $microClusters \leftarrow MicroClusterTuple(centroid, sumOfElements, clusterElements)$
35: end function
```

9) leading to the update of the latter (lines 18-26), otherwise a new micro-cluster will be created that contains the new vector (line 11). All micro-clusters are stored in the list $microClusters$ for the duration of a window and the list is returned in line 12 for further use.

Algorithm 10 is the batch phase 2 of the DBVARC, where the existing clusters are refined. In line 1, the DBVARC function gets as input the $microClusters$ and $statementVectors$ of the previous steps. The function aims to check the variance of the elements assigned in clusters and if it exceeds a threshold, then, the elements are either merged with the next closest cluster or are split into their own cluster. Specifically, in line 5, each element of a micro-cluster is checked for its variance as calculated in lines 13-18. In line 14, the mean of the cosine similarities between the cluster centroid and the cluster's elements' vectors is used to define a threshold (line 4). If the cosine similarity of a cluster element is lower than the threshold (line 16), then, the element has high variance and is stored in $elements$ (line 17) for further examination. In line 8, each element with high variance is examined for its neighbouring clusters defined by lines 19-21, where all cosine similarities between the element's vector and the clusters' centroids are sorted in descending order and the 30% neighbouring clusters are extracted (line 21) since the other clusters would be much further away to be even considered as neighbours. Line 9 defines whether the element will be merged to one of the neighbouring clusters or split into a new one. This decision is made in lines 22-29, where the closest triple in a neighbouring cluster is defined based on its cosine similarity with the element's vector (line 24). The cluster that this triple belongs to is the closest cluster to the element. In this way, not only the distance between cluster elements in regards to their centroid is examined but also to elements that could lie on the edge of a cluster. If the cosine similarity with the closest triple is higher than a static threshold (line 25), then, the element is merged to its closest cluster (line 26) leading to the updates of the old cluster and the cluster to be merged (lines 30-32). On the other hand, if the cosine similarity is lower than a static threshold (line 27), then, the element will be split and it is stored in $triplesToBeSplit$ for further use (line 28). In line 11, the elements belonging to a micro-cluster that is soon to be split, are examined. Specifically, a new cluster with all of the elements is created in line 34. Once again, the variance of the elements is examined in line 36, and according to a threshold (line 35), the elements will either remain in the new cluster or the elements with high variance will form their own cluster (lines 38-39). All new or updated micro-

Algorithm 10 DBVARC Phase 2 - Density-Based VARiance Clustering (Refinement of Clusters)

```
1: function DBVARC($microClusters, statementVectors$)
2:     if $endOfWindow$ then                                  ▷ phase 2 - refinement of clusters
3:         for $microCluster \leftarrow microClusters$ do
4:             $threshold \leftarrow round(mean) - 0.1$
5:             $elements \leftarrow highVarianceElements(microCluster, statementVectors, threshold)$
6:         end for
7:         for $microCluster \leftarrow microClusters$ do
8:             for $statementVector \leftarrow elements$ do
9:                 $sortedClusters \leftarrow neighbourClusters(microClusters, statementVector)$
10:                $triplesToBeSplit \leftarrow mergeOrSplit(statementVector, sortedClusters, microCluster)$
11:            end for
12:            if $\neg triplesToBeSplit \leftarrow \emptyset$ then
13:                $split(microCluster, triplesToBeSplit)$
14:            end if
15:        end for
16:    end if
17:    return $microClusters$                                  ▷ final set of clusters
18: end function
19: function highVarianceElements($microCluster, statementVectors, threshold$)
20:    $mean \leftarrow meanCosineSimilarity(microCluster.centroid, statementVectors.vector)$
21:    for $clusterElement \leftarrow microCluster.clusterElements$ do
22:        if $cosineSimilarity < threshold$ then
23:            $elements \leftarrow clusterElement$
24:        end if
25:    end for
26:    return $elements$
27: end function
28: function neighbourClusters($microClusters, statementVector$)
29:    $sortedClusters \leftarrow sort(cosineSimilarity(microClusters.centroid, statementVector.vector))$
30:    return 30% $of\ sortedClusters$
31: end function
32: function mergeOrSplit($statementVector, sortedClusters, oldCluster$)
33:    for $microCluster \leftarrow sortedClusters$ do
34:        $closestCluster \leftarrow maxCosineSimilarity(microCluster.vectors, statementVector.vector)$
35:    end for
36:    if $maxCosineSimilarity > 0.5$ then
37:        $merge(closestCluster, oldCluster, statementVector)$
38:    else
39:        $triplesToBeSplit \leftarrow statementVector$
40:    end if
41:    return $triplesToBeSplit$
42: end function
```

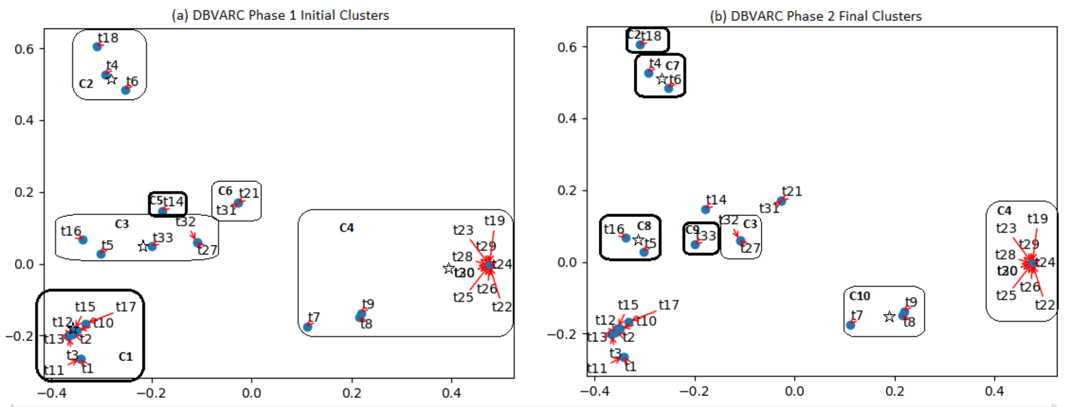
```
43: function MERGE($closestCluster, oldCluster, statementVector$)
44:     $updateCluster(oldCluster, statementVector, deletion)$
45:     $updateCluster(closestCluster, statementVector, addition)$
46: end function
47: function SPLIT($microCluster, triplesToBeSplit$)
48:     $createNewCluster(triplesToBeSplit)$
49:     $threshold \leftarrow round(mean - 0.02, 2)$
50:     $elements \leftarrow highVarianceElements(microCluster, triplesToBeSplit, threshold)$
51:     if $\neg elements \leftarrow \emptyset$ then
52:         $updateCluster(oldCluster, elements, deletion)$
53:         $createNewCluster(elements)$
54:     end if
55: end function
```

clusters are stored in the list $microClusters$, and the list is returned in line 12 for further use.

An example of two phases of DBVARC for the entity Usain_Bolt is given in Fig. 6.5. The squares represent a cluster (C1-C10) and the stars are their centroids. The bold squares are pure clusters (all elements are clustered correctly) based on the ground truth. It is observed that in phase 1 two clusters are found correctly (C1, C5), whereas in phase 2 the refinement of three clusters (C2, C3, C4) led to the creation of four more pure clusters (C2, C7, C8, C9). In clusters C2, C3 and C4, some elements are loosely inter-connected, therefore, they either form a separate common cluster (C7, C10) or they are split into more clusters (C8, C9). The process is further detailed in Fig. 6.6. In the example, after the initial clusters from DBVARC phase 1 (C1-C6), it is shown how clusters with size > 1 or mean < 1.0 are examined for elements with high variance. These elements are deleted from the existing clusters (updated C2, C3, C4), their neighbour clusters are found, and their closest element from each neighbour cluster decides whether they will be merged to this cluster or split into their own. The existing elements are all split into their own clusters (C7, C8, C10) apart from t33 that presented high variance to the new cluster, therefore it formed its own cluster (C9).

Triple2Rank

In this chapter, Triple2Rank is proposed, which is the final step of the approach, where each triple is ranked according to its importance. The ranking is based on the same concept behind taxonomies, where a hierarchical structure of words is created starting from words with basic-level concepts (parents) that are gradually



t1: <birthPlace, Trelawny_Parish,_Jamaica>	t12: <placeOfBirth, Trelawny_Parish>	t23: <title, Best_International_Athlete_ESPY...>
t2: <birthPlace, Trelawny_Parish>	t13: <placeOfBirth, Jamaica>	t24: <title, IAAF_World_Athlete_of_the_Year>
t3: <birthPlace, Jamaica>	t14: <wasDerivedFrom, http://en.wikipedia...>	t25: <title, Track_&_Field_News_Athlete_of_the_Year>
t4: <depiction, http://upload.wikimedia.org/...>	t15: <residence, Kingston,_Jamaica>	t26: <title, Jamaica_Sportsman_of_the_year>
t5: <event, Sprint_(running)>	t16: <sport, Track_and_field>	t27: <title, 100_metres>
t6: <hasPhotoCollection, http://wifo5-03...>	t17: <stateOfOrigin, Jamaica>	t28: <title, L'Équipe_Champion_of_Champions>
t7: <honorificPrefix, The_Honourable>	t18: <thumbnail, http://upload.wikimedia.org/.>	t29: <title, Best_Track_and_Field_Athlete_ESPY...>
t8: <honorificPrefix, Order_of_Distinction>	t19: <title, BBC_Overseas_Sports_Personality_of_the_Year>	t30: <title, Jamaica_Sportsperson_of_the_year>
t9: <honorificSuffix, Order_of_Jamaica>	t20: <title, Laureus_World_Sports_Award_for_Sportsman...>	t31: <title, Men's_100_metres_world_record_progression>
t10: <nationality, Jamaica>	t21: <title, Men's_200_metres_world_record_progression>	t32: <title, 200_metres>
t11: <placeOfBirth, Trelawny_Parish,_Jamaica>	t22: <title, Track_&_Field_Athlete_of_the_Year>	t33: <years, 2012_Summer_Olympics>

Figure 6.5: An example of the two phases of DBVARC for the entity Usain_Bolt in DBpedia.

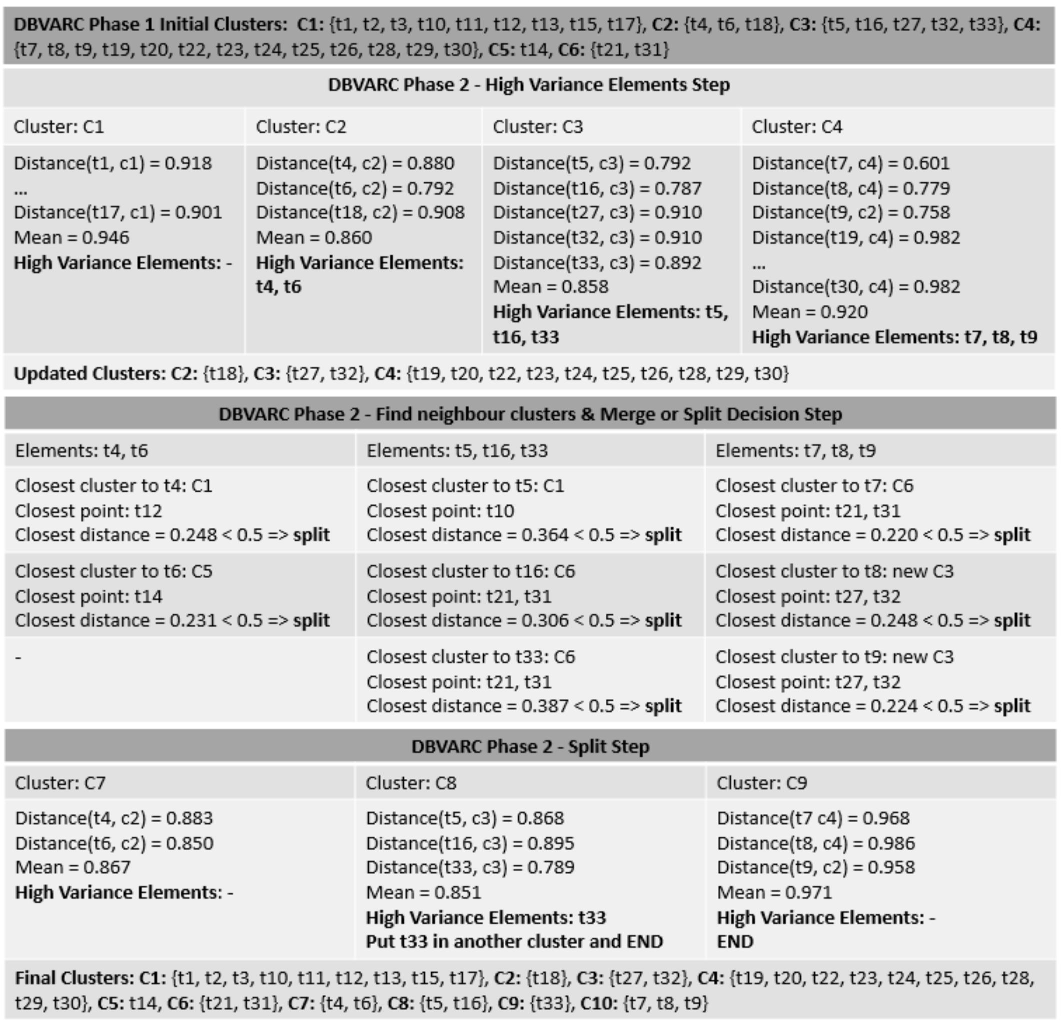
split into words of refined-level concepts (children) that could be subsumed by their parents. The ranking is based on the following rules:

- Basic-level concepts are characterised by shorter and more polysemous words [216]. Therefore, the fewer tokens a word contains and the shorter it is, the higher it should be in the hierarchy.
- Values that are not only more popular (frequent) but also more informative (rare) than others, should be higher in the hierarchy.
- Values that belong to an abnormal/extreme low or high range are more important than others, and should be higher in the hierarchy.

Algorithm 11 is a ranking methodology that measures the importance of a triple within and among all conceptual clusters of an entity. In line 1, the Triple2Rank function gets as input the *microClusters* of Algorithm 10 and the *Reasoner*, which contains the reasoning rules. The function aims to score the triples of the clusters based on significance, but also to define a selection order that chooses which triples will be selected first for the final summary. Line 2 proceeds with the scoring of each triple that is explained in lines 8-14. For each triple of a cluster, a separate scoring is calculated for the property (lines 15-16) and the object (lines 17-22). Since the property is an actual word, a penalty is

Algorithm 11 Triple2Rank

```
1: function Triple2Rank($microClusters, Reasoner$)    ▷ $Reasoner$: contains reasoning rules
2:     $scoredStatementsOfClusters \leftarrow score(microClusters)$
3:     $averageScorePerCluster \leftarrow averageScore(scoredStatementsOfClusters)$
4:     $orderedClusters \leftarrow sort\ based\ on\ averageScore\ or\ sizeOfCluster\ if\ averageScore\ is\ tied$
5:     while $\neg scoredStatementsOfClusters \leftarrow \emptyset$ do
6:         $scoredStatements \leftarrow startSelectionRound(orderedClusters, scoredStatementsOfClusters)$
7:     end while
8:     return $scoredStatements$    ▷ ordered timestamped scored triples of window
9: end function
10: function score($microClusters$)
11:     for $microCluster \leftarrow microClusters$ do
12:         for $triple \leftarrow microCluster.clusterElements$ do
13:             $finalScore \leftarrow propertyScore(property) + objectScore(object)$
14:             $scoredStatementsOfCluster \leftarrow ScoredStatement(timestamp, triple, finalScore)$
15:         end for
16:         $scoredStatementsOfClusters \leftarrow sort(scoredStatementsOfCluster)$
17:     end for
18:     return $scoredStatementsOfClusters$
19: end function
20: function propertyScore($property$)
21:     return $100 + (1 - numberOfTokensOfProperty) + (1 - lengthOfProperty)$
22: end function
23: function objectScore($object$)
24:     if $object\ is\ numerical$ then
25:         $object \leftarrow Reasoner(object)$
26:         return $score\ based\ on\ value\ range\ of\ reasoning\ rules$
27:     else
28:         return $frequencyOfObjectInCluster * \log \frac{numOfClusters}{numOfClustersThatContainObject}$
29:     end if
30: end function
31: function startSelectionRound($orderedClusters, scoredStatementsOfClusters$)
32:     for $cluster \leftarrow orderedClusters$ do
33:         for $scoredStatement \leftarrow scoredStatementsOfCluster$ do
34:             if $object\ is\ from\ Reasoner$ then
35:                 if $object\ not\ selected\ or\ end\ is\ reached$ then
36:                     $scoredStatements \leftarrow scoredStatement$
37:                     $scoredStatementsOfCluster.remove(scoredStatement)$
38:                 end if
39:             else
40:                 if $property\ not\ selected\ or\ end\ is\ reached$ then
41:                     $scoredStatements \leftarrow scoredStatement$
42:                     $scoredStatementsOfCluster.remove(scoredStatement)$
43:                 end if
44:             end if
45:         end for
46:     end for
47:     return $scoredStatements$
48: end function
```



DBVARC Phase 1 Initial Clusters: C1: {t1, t2, t3, t10, t11, t12, t13, t15, t17}, **C2:** {t4, t6, t18}, **C3:** {t5, t16, t27, t32, t33}, **C4:** {t7, t8, t9, t19, t20, t22, t23, t24, t25, t26, t28, t29, t30}, **C5:** t14, **C6:** {t21, t31}

DBVARC Phase 2 - High Variance Elements Step

Cluster: C1	Cluster: C2	Cluster: C3	Cluster: C4
Distance(t1, c1) = 0.918 ... Distance(t17, c1) = 0.901 Mean = 0.946 **High Variance Elements: -**	Distance(t4, c2) = 0.880 Distance(t6, c2) = 0.792 Distance(t18, c2) = 0.908 Mean = 0.860 **High Variance Elements: t4, t6**	Distance(t5, c3) = 0.792 Distance(t16, c3) = 0.787 Distance(t27, c3) = 0.910 Distance(t32, c3) = 0.910 Distance(t33, c3) = 0.892 Mean = 0.858 **High Variance Elements: t5, t16, t33**	Distance(t7, c4) = 0.601 Distance(t8, c4) = 0.779 Distance(t9, c2) = 0.758 Distance(t19, c4) = 0.982 ... Distance(t30, c4) = 0.982 Mean = 0.920 **High Variance Elements: t7, t8, t9**

Updated Clusters: C2: {t18}, **C3:** {t27, t32}, **C4:** {t19, t20, t22, t23, t24, t25, t26, t28, t29, t30}

DBVARC Phase 2 - Find neighbour clusters & Merge or Split Decision Step

Elements: t4, t6	Elements: t5, t16, t33	Elements: t7, t8, t9
Closest cluster to t4: C1 Closest point: t12 Closest distance = 0.248 < 0.5 => **split**	Closest cluster to t5: C1 Closest point: t10 Closest distance = 0.364 < 0.5 => **split**	Closest cluster to t7: C6 Closest point: t21, t31 Closest distance = 0.220 < 0.5 => **split**
Closest cluster to t6: C5 Closest point: t14 Closest distance = 0.231 < 0.5 => **split**	Closest cluster to t16: C6 Closest point: t21, t31 Closest distance = 0.306 < 0.5 => **split**	Closest cluster to t8: new C3 Closest point: t27, t32 Closest distance = 0.248 < 0.5 => **split**
-	Closest cluster to t33: C6 Closest point: t21, t31 Closest distance = 0.387 < 0.5 => **split**	Closest cluster to t9: new C3 Closest point: t27, t32 Closest distance = 0.224 < 0.5 => **split**

DBVARC Phase 2 - Split Step

Cluster: C7	Cluster: C8	Cluster: C9
Distance(t4, c2) = 0.883 Distance(t6, c2) = 0.850 Mean = 0.867 **High Variance Elements: -**	Distance(t5, c3) = 0.868 Distance(t16, c3) = 0.895 Distance(t33, c3) = 0.789 Mean = 0.851 **High Variance Elements: t33** **Put t33 in another cluster and END**	Distance(t7 c4) = 0.968 Distance(t8, c4) = 0.986 Distance(t9, c2) = 0.958 Mean = 0.971 **High Variance Elements: -** **END**

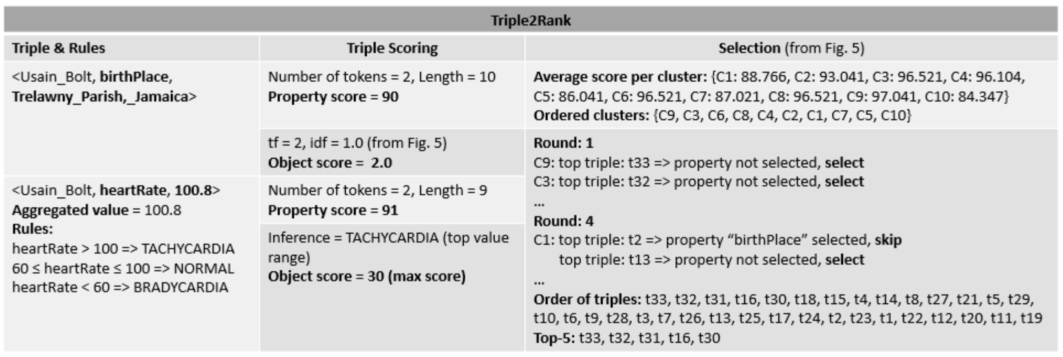
Final Clusters: C1: {t1, t2, t3, t10, t11, t12, t13, t15, t17}, **C2:** {t18}, **C3:** {t27, t32}, **C4:** {t19, t20, t22, t23, t24, t25, t26, t28, t29, t30}, **C5:** t14, **C6:** {t21, t31}, **C7:** {t4, t6}, **C8:** {t5, t16}, **C9:** {t33}, **C10:** {t7, t8, t9}

Figure 6.6: The process of the Usain_Bolt example in Fig. 6.5.

given to it according to the number of tokens that it contains and its length. The final property score is given in line 16, meaning that the more tokens a word contains or the longer it is, the higher the penalty. On the other hand, the object is either an entity or a number. If the object is numerical (aggregated value), then, the Reasoner is called and its high-level inference is deducted that replaces the numerical value (lines 18-19). The score given is based on the value range from which the inference was taken (line 20). If the object is an entity, the score given is based on a cluster-based frequency–inverse document frequency (tf-idf). In this modified tf-idf, the cluster the object belongs to, is the document, and all clusters are the corpus. Therefore, the tf is the number of the object's occurrence within the cluster, and the idf is based on the uniqueness of the object among all clusters (line

22). The triple score is the combination of its property's and object's score in line 11. A $scoredStatement$ is then created in line 12 that contains the timestamped scored triple based on the publication's original timestamp and it is stored in the list $scoredStatementsOfCluster$. The list is sorted in descending order in line 13 and it is returned in line 14 for further use. This list is, then, used for defining the average score of the triples within each cluster (line 3). The average scores are used in line 4 for defining the order in which each cluster will be selected for choosing the best every time triple for the final summary. Priority is given to clusters with a higher average score, but if the score is tied, then, the bigger cluster will be selected first. In lines 5-6, the selection round is taking place, described in lines 23-34. This is a continuous process until all triples have been selected. In this process, a cluster is visited based on its order (line 24), and the highest-scored $scoredStatement$ within the cluster is selected (line 25). If the object of this triple contains a high-level inference (line 26), then, if it has not already been selected in another round, or if all objects of a cluster have already been selected once in other rounds (line 27), the $scoredStatement$ is chosen as the next to be selected for the final summary in line 28, and it is removed from the collection of $scoredStatements$ in line 29, otherwise the next highest-scored $scoredStatement$ within the cluster is selected and so on. On the other hand, if the object is an entity, then, the same selection applies, but for the properties instead of objects (lines 30-33). This provides more diversity in the final summary. The final order of the selected $scoredStatements$ is given in line 7 and is used for top-k selection.

An example of Triple2Rank is given in Fig. 6.7. In the example, the triple scoring is shown for both cases, where the object is an entity Trelawny_Parish,_Jamaica or an aggregated number "100.8". In the case of the numerical object, the reasoning result based on the rules is TACHYCARDIA, which belongs to the top value range of the rules [min-60, 60-100, 100-max], therefore, the maximum score is given to the object. The triple selection is also presented for the triples of Fig. 6.5. C3, C6, and C8 have tied average scores, but they also have the same cluster sizes; therefore, the order among the three clusters is not important. In terms of selection rounds, in Round 4, the birthPlace property has already been chosen before (t3) from C1, therefore, even if t2 is the top-scored triple in the cluster, the next one (t13) will be examined and so on.



Triple2Rank		
Triple & Rules	**Triple Scoring**	**Selection** (from Fig. 5)
<Usain_Bolt, **birthPlace, Trelawny_Parish,_Jamaica**>	Number of tokens = 2, Length = 10 **Property score = 90**	**Average score per cluster:** {C1: 88.766, C2: 93.041, C3: 96.521, C4: 96.104, C5: 86.041, C6: 96.521, C7: 87.021, C8: 96.521, C9: 97.041, C10: 84.347} **Ordered clusters:** {C9, C3, C6, C8, C4, C2, C1, C7, C5, C10}
	tf = 2, idf = 1.0 (from Fig. 5) **Object score = 2.0**	**Round: 1** C9: top triple: t33 => property not selected, **select** C3: top triple: t32 => property not selected, **select** ... **Round: 4** C1: top triple: t2 => property “birthPlace” selected, **skip** top triple: t13 => property not selected, **select** ... **Order of triples:** t33, t32, t31, t16, t30, t18, t15, t4, t14, t8, t27, t21, t5, t29, t10, t6, t9, t28, t3, t7, t26, t13, t25, t17, t24, t2, t23, t1, t22, t12, t20, t11, t19 **Top-5:** t33, t32, t31, t16, t30
<Usain_Bolt, **heartRate, 100.8**> **Aggregated value** = 100.8 **Rules:** heartRate > 100 => TACHYCARDIA 60 ≤ heartRate ≤ 100 => NORMAL heartRate < 60 => BRADYCARDIA	Number of tokens = 2, Length = 9 **Property score = 91**	
	Inference = TACHYCARDIA (top value range) **Object score = 30 (max score)**	

Figure 6.7: An example of Triple2Rank.

6.5 Evaluation

This section provides the evaluation of the approach that consists of the methodology followed and the datasets used as well as the metrics and the final results.

6.5.1 Datasets and Methodology

A combination of different real-world datasets has been used that relate to the applications of Healthcare and Smart Cities.

FACES[2]: As aforementioned in Chapter 4, this is a dataset that contains entities from DBpedia. The focus is given on entities that refer to people (e.g. Albert_Einstein) for the Healthcare application and to places or buildings (e.g. Rice_University) for the Smart Cities application. No literals were considered, and the data was pre-processed by keeping only the last part after a "/" or "#" in URIs so that the data makes more sense from the user perspective. If duplicates occurred (ontology - property properties), then, they were deleted. The same typing information of the objects was used as in Chapter 4.

MIMIC II Database[3]: As aforementioned in Chapter 5, this is a dataset that contains heart-related readings (e.g. heart rate, central venous pressure etc.) of patients that developed or were at risk of developing an acute hypotensive episode. Apart from the sensor readings, this chapter's work uses the annotations and medical history of patients. The goal of the medical history is to record the major problems of the patient at or before admission to the ICU so it can help a doctor decide what led to the development of the current situation. The same data pre-

[2]http://wiki.knoesis.org/index.php/FACES

[3]`https://archive.physionet.org/challenge/2009/training-set.shtml`

processing and selection as in Chapter 5 occurred, with the addition that medical history turned into triples (e.g. <Patient, medicalRecord, ...>).

Intel Lab Data[4]: As aforementioned in Chapter 5, this is a dataset that contains sensor readings involving environmental aspects (e.g. temperature, humidity etc.) of the Intel Berkeley Research lab. The same data pre-processing and selection occurred as in Chapter 5.

CityPulse[5]: As aforementioned in Chapter 5, this is a dataset that contains readings regarding weather (e.g. dew point, wind speed etc.), pollution (e.g. ozone, carbon monoxide etc.), and road traffic (e.g. vehicle count). Apart from the sensor readings, this chapter's work uses the cultural events (e.g. concerts) provided. The same data pre-processing and selection occurred as in Chapter 5, with the addition that the cultural events were translated from Danish to English, and turned into triples (e.g. <Place, culturalEvent, Chamber_concert>). The provided type of the event (object of the triple) was used as its typing information.

UCI Electric Consumption[6]: As aforementioned in Chapter 5, this is a dataset that contains readings regarding electric power consumption in one household (e.g. global active power, voltage etc.). The same data pre-processing and selection occurred as in Chapter 5.

The manual rules were mostly created based on the M3 framework's rules[7] and the annotations of the MIMIC II Database, which contained ranges of attributes' values when an alert occurred. An overview of the datasets and their characteristics is given in Table 6.3.

The evaluation methodology is illustrated in Fig. 6.8 as well as an example for the properties temperature, humidity, dewPoint, and pressure. Specifically, the effectiveness evaluation is based on three types of ground truth; relevance, conceptual, and user-defined. The relevance ground truth, as explained in Chapter 5, was created by choosing each original property (word) (step I in Fig. 6.8) and storing its synonyms, related words, and antonyms deriving from multiple thesauri as well as its hyponyms (children) observed in ontologies (step II). In the case of polysemy, the words that applied to the application in use were chosen (e.g. "pressure" as body measurement in the first use case and as environmental one

[4]`http://db.csail.mit.edu/labdata/labdata.html`

[5]`http://iot.ee.surrey.ac.uk:8080/datasets.html`

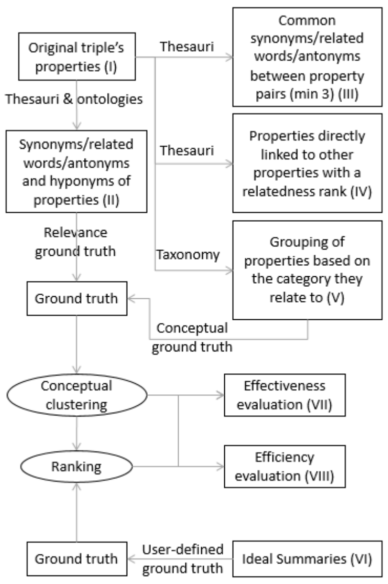
[6]`https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption`

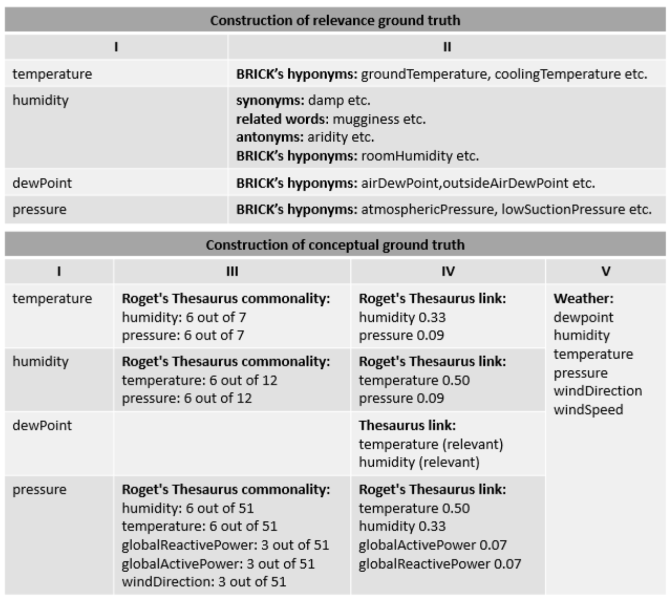
[7]`https://github.com/gyrard/M3Framework/tree/master/war/RULES`

Table 6.3: Datasets used for the Healthcare and Smart Cities use cases

Dataset	**Use Case**	**Data/ Sensors**	**Pre-processing**
FACES	All	DBpedia (people, places, and buildings)	Deletion of data-type properties (textual or numerical) and duplicates.
MIMIC II Database	Healthcare	heart readings, annotations, medical history	The readings and medical history turned to triples (e.g. <Patient, centralVenousPressure, 19.800>). Noise and missing values were deleted. The annotations were used for reasoning rules.
Intel Lab Data	Smart Cities	temperature, voltage, light, humidity	The readings turned to triples (e.g. <DiningTable1, humidity, 40.399>). Missing values replaced with nearby ones.
CityPulse	Smart Cities	weather, pollution, road traffic, cultural events	The readings turned to triples (e.g. <Amsterdam, particulateMatter, 67>). The cultural events were translated from Danish to English, turned to triples (e.g. <Amsterdam, culturalEvent, Chamber_concert>), and their types were used as typing information.
UCI Electric Consumption	Smart Cities	electric readings	The readings turned to triples (e.g. <Kitchen, activeEnergy, 36.000>).

in the second case). If the original word did not exist in the thesauri, then, its separate tokens were considered. The conceptual ground truth proved more challenging as most summarisation approaches use only users to evaluate their results; nevertheless, this type of evaluation is dependent on the users at hand and their experience, resulting in a more subjective view. Therefore, a conceptual ground truth was created that is based on context coherence, that is the number of common words was observed between each original property pair (minimum common words' number was 3) (step III) as well as if words were directly linked to other words with a relatedness rank via thesauri (e.g. Roget's thesaurus) (step IV). Also, it was checked in taxonomies if any words belonged to the same category so that they could be clustered together (step V). In the example, there is a high commonality between temperature, humidity, and pressure; nevertheless, pressure has some commonality with other unrelated to temperature and humidity properties like globalReactivePower, globalActivePower, and windDirection (step III). To define to which conceptual cluster pressure belongs, their direct links among them from thesauri are observed. It is shown that temperature, humidity, and dewPoint have high relevance, and pressure also is linked to them, but not so strongly; nevertheless, the link between temperature, humidity, and dewPoint is higher than the one with globalReactivePower and globalActivePower (step IV); therefore, the final conceptual cluster is {temperature, humidity, dewPoint, pressure}. In both ground truth types, multiple thesauri were used to provide a more reliable gold standard. The user-defined ground truth (step VI) is provided by FACES, and as explained in Chapter 4, it involves the ideal triples selection for specific entities by 15 human judges (for k = 5 and k = 10). This ground truth was used in the evaluation for providing a more complete effectiveness evaluation (step VII). On the other hand, the efficiency evaluation (step VIII) is based on metrics, like latency, throughput, etc. It is taking place after the final ranking has occurred for all triples.


Original triple's properties (I)
Thesauri
Common synonyms/related words/antonyms between property pairs (min 3) (III)
Thesauri & ontologies
Synonyms/related words/antonyms and hyponyms of properties (II)
Thesauri
Properties directly linked to other properties with a relatedness rank (IV)
Relevance ground truth
Taxonomy
Grouping of properties based on the category they relate to (V)
Ground truth
Conceptual ground truth
Conceptual clustering
Effectiveness evaluation (VII)
Ranking
Efficiency evaluation (VIII)
Ground truth
User-defined ground truth
Ideal Summaries (VI)



Construction of relevance ground truth	
I	II
temperature	**BRICK's hyponyms:** groundTemperature, coolingTemperature etc.
humidity	**synonyms:** damp etc. **related words:** mugginess etc. **antonyms:** aridity etc. **BRICK's hyponyms:** roomHumidity etc.
dewPoint	**BRICK's hyponyms:** airDewPoint,outsideAirDewPoint etc.
pressure	**BRICK's hyponyms:** atmosphericPressure, lowSuctionPressure etc.

Construction of conceptual ground truth			
I	III	IV	V
temperature	**Roget's Thesaurus commonality:** humidity: 6 out of 7 pressure: 6 out of 7	**Roget's Thesaurus link:** humidity 0.33 pressure 0.09	**Weather:** dewpoint humidity temperature pressure windDirection windSpeed
humidity	**Roget's Thesaurus commonality:** temperature: 6 out of 12 pressure: 6 out of 12	**Roget's Thesaurus link:** temperature 0.50 pressure 0.09	
dewPoint		**Thesaurus link:** temperature (relevant) humidity (relevant)	
pressure	**Roget's Thesaurus commonality:** humidity: 6 out of 51 temperature: 6 out of 51 globalReactivePower: 3 out of 51 globalActivePower: 3 out of 51 windDirection: 3 out of 51	**Roget's Thesaurus link:** temperature 0.50 humidity 0.33 globalActivePower 0.07 globalReactivePower 0.07	

Figure 6.8: The evaluation methodology and an example for the properties temperature, humidity, dewPoint, and pressure.

The thesauri that have been used for the methodology described above, for both use cases, are the Merriam-Webster[8], Thesaurus[9], and Roget's Thesaurus[10]. The taxonomy IPTC Subject Codes[11] also acted as a reference. For the Healthcare use case, the SNOMED CT taxonomy[12] was used, which is an organised collection of medical terms (e.g. central venous pressure etc.), whereas for the Smart Cities use case, the BRICK ontology[13] was examined for selecting synonyms/related words up to level 0 to 2 deep, which contains words regarding the measurements of buildings (e.g. pressure, humidity, luminance etc.).

The effectiveness evaluation has been done offline, that is the results are examined on the assumption that all available data belongs in a window, whereas the efficiency evaluation is done online. All experiments were run 5 times and the average result was taken. The runs took place in a laptop with Intel(R) Core(TM) i7-6600U CPU@2.60GHz 2.80GHz, and 16GB of RAM. Regarding the implementation, for the RDF models, Apache Jena[14] was used, for the embedding model, deeplearning4j[15] was used. Regarding PubSum approach, Smile[16] was used for DBSCAN. Regarding FACES-adapted, WordNet of extJWNL[17] was used, for tagging, Apache OpenNLP[18] was used, for the proposed version of Cobweb algorithm by FACES, the corresponding class by MOA[19] was modified and for the proposed version of tf-idf by FACES, a TDB Triple Store[20] was used that stores part of the DBpedia ontology.

6.5.2 Metrics

Several metrics have been used to evaluate the effectiveness and efficiency of the approach. The effectiveness evaluation focuses on the correctness, whereas the efficiency one focuses on the performance.

[8]https://www.merriam-webster.com/thesaurus

[9]https://www.thesaurus.com/

[10]http://www.roget.org/

[11]https://docs.aylien.com/newsapi/search-taxonomies/#search-labels-for-iptc-subject-codes

[12]https://browser.ihtsdotools.org/?

[13]https://brickschema.org/#home

[14]https://jena.apache.org/

[15]https://deeplearning4j.org/

[16]https://haifengl.github.io/smile/nlp.html

[17]http://extjwnl.sourceforge.net/

[18]https://opennlp.apache.org/

[19]https://moa.cms.waikato.ac.nz/

[20]https://jena.apache.org/documentation/tdb/

Correctness

Correctness consists of the agreement, diversity consensus, and quality metrics by using the three aforementioned types of ground truth; relevance, conceptual, and user-defined.

Agreement and Diversity Consensus The agreement metric Agr is defined in Chapter 4, whereas the diversity consensus $DivCon$ shows the degree of triple selection by the judges in the user-defined ground truth that belong to different conceptual clusters in the conceptual ground truth. The diversity consensus is defined as:

$$DivCon = \frac{1}{n}\sum_{i=1}^{n}\begin{cases} |\frac{CC(Summ_i^I(e))}{Summ_i^I(e)}|, & \text{if } |Summ_i^I(e)| \leqslant |CC(e)| \\ |\frac{CC(Summ_i^I(e))}{CC(e)}|, & \text{otherwise} \end{cases} \tag{6.1}$$

where n is the number of summaries, $Summ_i^I(e)$ is the i-th ideal summary for an entity e, and $CC(Summ_i^I(e))$ as well as $CC(e)$ are the distinct conceptual clusters belonging to the i-th ideal summary and an entity e, respectively.

Quality The quality refers either to the final summary/notification (after ranking) that takes place or to the conceptual clustering (before ranking). The former involves the QUD and $RS\_F-score$ metrics, whereas the latter involves the $RCC\_F-score$ metric.

QUD is the quality user-defined metric that is based on the commonalities between the approach's summaries and the ideal ones, in the user-defined ground truth. It has already been analysed in Chapter 4; nevertheless, in this chapter's work, no time-dependent adaptation of the original metric is taking place as the effectiveness evaluation is happening offline. Therefore, it is defined as:

$$QUD(S(e)) = \frac{1}{n}\sum_{i=1}^{n}|Summ(e) \cap Summ_i^I(e)| \tag{6.2}$$

where $Summ(e)$ is the approach's summary.

$RS\_F-score$ and $RCC\_F-score$ are the redundancy-aware F-score of the summary and the conceptual clustering, respectively. The $RS\_F-score$ metric has been explained and defined in Chapter 4; nevertheless, in this chapter's work, the metric was adapted to cater for conceptually similar information (not only duplicates) as shown in Table 6.4.

Table 6.4: Redundancy-aware evaluation components

Approach \ Ground Truth	conceptually similar	not conceptually similar
conceptually similar	TP or R^{-}	FP or N^{-}
not conceptually similar	FN or R^{+}	-

According to Table 6.4, all words that were selected by the approach to belonging to the same conceptual cluster and are indeed in the same cluster based on the conceptual ground truth, are considered as True Positives (TP). On the other hand, if they do not belong to the same cluster based on the ground truth, then, they are considered as False Positives (FP). The words that the approach put in different conceptual clusters, but belong to the same cluster based on the ground truth, are considered as False Negatives (FN). Similarly, regarding the summary, the TP could refer to the set of non-delivered redundant triples (R^{-}), the FP to the set of non-delivered non-redundant triples (N^{-}), and FN to the set of delivered redundant triples (R^{+}). The $RCC\_F-score$ has the same formula to the $F-score$ metric of Chapter 5 with the difference that they refer to different ground truths and that the $RCC\_F-score$ is related to clustering; therefore, another procedure should take place. Specifically, each conceptual cluster of the ground truth is checked against each conceptual cluster of the approach's summary and, then, based on the commonality percentage of the attributes, each ground truth cluster is mapped to its most representative one of the approach's summary (the one with the highest commonality percentage). In this way, non-pure clusters are taken into account and their performance is observed based on Table 6.4. The difference between the two metrics is that $RCC\_F-score$ shows the effectiveness of the approach on detecting conceptually similar words, whereas the $RS\_F-score$ shows the loss of important information that took place via ranking and sending only a subset of the whole available information (top-k).

Performance

The performance consists of the processing time of individual processes like clustering or scoring as well as the end-to-end latency, size reduction, memory footprint, and throughput metrics. All these metrics have been defined in Chapter 4 with the difference that in this chapter's work the throughput is examined from two perspectives: 1) the number of triples/events the system receives (throughput in), and 2) the one that it extracts in the form of summaries (throughput out), in a specific amount of time.

6.5.3 Results

The results are analysed for the following approaches: 1) the typical Pub/Sub approach (No top-k), where no summary is taking place and all publications are sent to the subscriber as notifications, 2) FACES-adapted, where according to the original authors [50] the use of $Cobweb-cut-off = 5$ and $Cobweb-path-level = 3$ yields the best results, 3) PubSum approach, analysed in Chapter 4, where GOOGLENEWS[21] Word2Vec model, $\varepsilon = 1$ and $minPts = 1$ for DBSCAN yields the best results as well as the choice of Euclidean distance for ranking, and 4) the proposed PoSSUM approach that uses the pre-trained ConceptNet[22] word embedding model.

Agreement and Diversity Consensus

According to Table 6.5, the average agreement among all entities regarding people, places, and buildings within FACES ideal summaries is $Agr = 1.917$ for k = 5, and $Agr = 4.587$ for k = 10. This means that approximately 2 out of 5, and 5 out of 10 triples were identical among different judges when their top choices for an entity summary were concerned. On the other hand, the diversity consensus is $DivCon = 0.780$ for k = 5, and $DivCon = 0.651$ for k = 10. This shows that almost 80% and 65% of the top-5 and top-10, respectively, ideal information of the judges is diverse, meaning that it belongs to different conceptual clusters.

These results prove that judges might not highly agree on which triples are important (humans' subjective nature), especially when less information is considered (the top-5 agreement is smaller than the top-10 one). Nevertheless, they also prove that judges do agree that the less information they are provided with, the more diversity they require. This is a good indication that a diverse entity summarisation approach is deemed essential in heterogeneous and data

[21]https://code.google.com/archive/p/word2vec/
[22]https://github.com/commonsense/conceptnet-numberbatch

Table 6.5: Average Agreement and Diversity Consensus for each entity

Entity	Agreement		Diversity Consensus	
	k = 5	**k = 10**	**k = 5**	**k = 10**
Bill Gates	1.679	4.107	0.725	0.625
Richard Mentor Johnson	0.821	2.857	0.800	0.758
Albert Einstein	2.048	5.238	0.771	0.714
England	2.464	5.464	0.950	**0.800**
Joe Biden	2.190	4.238	0.793	0.640
Walt Disney	2.286	4.381	0.814	0.671
Rice University	2.214	5.536	0.850	0.738
Barack Obama	1.500	4.000	0.775	0.650
Benjamin Franklin	1.666	4.238	0.571	0.443
Alan Turing	1.666	4.286	0.718	0.586
Edward Teller	1.714	4.607	0.800	0.625
Mike Turner	2.238	5.381	0.829	0.629
Charles Darwin	1.143	3.286	0.743	0.686
Amsterdam	1.714	3.786	0.650	0.519
Vladimir Putin	1.619	3.143	0.771	0.614
Poland	2.214	4.821	0.850	0.716
John Napier	2.429	5.536	0.800	0.638
Czech Republic	2.286	4.786	**1.000**	0.725
Bruce Lee	1.464	4.536	0.658	0.588
Marie Curie	2.286	5.286	0.750	0.625
Henry IV of France	1.607	4.036	0.850	0.671
Seychelles	2.333	5.333	0.886	0.686
J. C. Penney	**2.571**	**7.476**	0.829	0.696
Ann Arbor, Michigan	1.893	4.786	0.650	0.508
Nile	1.476	4.000	0.707	0.603
Yukon River	1.500	5.000	0.669	0.656
Norway	2.429	5.476	0.857	0.743
Texas	2.429	3.667	0.829	0.729
Friedrich Engel (mathematician)	1.571	3.607	0.800	0.639
Usain Bolt	2.048	4.714	0.714	0.607
All (average)	**1.917**	**4.587**	**0.780**	**0.651**

voluminous environments like IoT since it would not only resolve resource constraints of the environment but would also not overwhelm the users with redundant information.

Quality

The quality of the final summary and the conceptual clustering is shown in Table 6.6 and Table 6.7. Table 6.6 shows the overlap between the user-defined ground truth and the approach's summary for each entity. It is shown that on average the best approach is PoSSUM for both top-5 and top-10 summaries. This does not indicate that it always performs the best for all entities since this happens for 17 out of 30 entities in top-5 summaries, and 13 out 30 in top-10 ones. The second-best approach is FACES with the best results for 9 out of 30 entities in top-5 summaries, and 11 out of 30 in top-10 ones, leaving as last PubSum with 5 out of 30, and 7 out of 30, respectively. This shows that PoSSUM behaves the best by a large margin for shorter summaries (top-5) by capturing better the most ideal triples by the judges. In the case of larger summaries (top-10), PoSSUM still performs the best but very closely to FACES in regards to the number of best-performing entities. Nevertheless, especially for top-10 summaries, it is observed that when PoSSUM is better than FACES, the quality may be even twice better. This is not the case when FACES is better, indicating that even though the number of the best entities for PoSSUM and FACES for the top-10 is close, the quality of PoSSUM is much better in total. In general, it is observed that for all approaches the quality gets better for higher k as more triples are selected; therefore, the chances of a summary containing an ideal triple by the judges are higher. Also, the quality numbers indicate the slight discrepancy among judges of what is an ideal summary resulting in numbers that would never achieve the maximum possible results as they are analogous to the agreement Agr metric.

The quality of the final summary is not only dependent on the triple ranking as depicted in Table 6.6, but also on the conceptual clusters that have been formed. This is not depicted in the QUD metric as it may lead to a good result for an entity, but at the same time, the entity's conceptual clustering might perform poorly. Therefore, $RS\_F-score$ and $RCC\_F-score$, shown in Table 6.7, give a more thorough view of the quality of all of the steps of an approach (triple's vector representation, conceptual clustering, and ranking). FACES is only analysed for its dataset, in regards to $RS\_F-score$, since its

Table 6.6: Average QUD for each entity

Entity	**QUD**					
	FACES		**PubSum**		**PoSSUM**	
	k = 5	**k = 10**	**k = 5**	**k = 10**	**k = 5**	**k = 10**
Bill Gates	0.125	0.273	**0.205**	**0.285**	0.110	0.268
Richard Mentor Johnson	0.105	0.175	0.015	0.189	**0.130**	**0.311**
Albert Einstein	0.080	0.254	**0.354**	**0.500**	0.229	0.443
England	0.215	**0.503**	0.150	0.413	**0.250**	0.435
Joe Biden	**0.249**	**0.491**	0.029	0.221	0.111	0.265
Walt Disney	0.051	0.177	**0.114**	**0.231**	0.085	0.194
Rice University	**0.473**	**0.562**	0.326	0.478	0.225	0.443
Barack Obama	**0.220**	0.298	0.190	**0.304**	0.025	0.271
Benjamin Franklin	**0.273**	**0.403**	0.136	0.317	0.121	0.307
Alan Turing	0.126	**0.323**	0.125	0.277	**0.206**	0.226
Edward Teller	0.255	**0.475**	0.225	0.322	**0.375**	0.432
Mike Turner	0.206	0.357	0.200	**0.369**	**0.366**	0.346
Charles Darwin	0.097	0.248	0.137	**0.318**	**0.219**	0.292
Amsterdam	**0.160**	**0.259**	0.095	0.237	0.075	0.150
Vladimir Putin	0.046	0.210	0.057	0.210	**0.063**	**0.218**
Poland	0.030	0.247	0.035	0.143	**0.400**	**0.545**
John Napier	0.295	0.572	0.200	0.478	**0.350**	**0.580**
Czech Republic	0.215	0.358	0.030	0.278	**0.425**	**0.483**
Bruce Lee	0.098	0.240	**0.221**	0.324	0.071	**0.456**
Marie Curie	0.170	0.455	0.310	0.390	**0.325**	**0.488**
Henry IV of France	0.045	0.290	0.130	0.273	**0.235**	**0.310**
Seychelles	0.177	0.386	0.171	0.343	**0.343**	**0.500**
J. C. Penney	0.394	**0.734**	**0.486**	0.600	0.371	0.657
Ann Arbor, Michigan	**0.205**	**0.407**	0.155	0.307	0.100	0.279
Nile	**0.139**	0.291	0.120	0.311	0.133	**0.383**
Yukon River	0.151	0.418	0.160	**0.478**	**0.413**	0.413
Norway	0.200	0.366	0.069	0.220	**0.314**	**0.563**
Texas	**0.137**	**0.333**	0.000	0.191	0.093	0.247
Friedrich Engel (mathematician)	0.130	0.270	0.230	0.347	**0.300**	**0.360**
Usain Bolt	**0.360**	**0.390**	0.149	0.259	0.309	0.357
All (average)	0.181	0.359	0.161	0.320	**0.226**	**0.371**

triple ranking methodology is semantically-coupled; therefore, it could not be applied for other data than DBpedia. The rest of the datasets concern either data-type (numerical) properties (Numerical Healthcare and Numerical Smart Cities) or object-type properties (FACES dataset integrated with medicalRecord triples for the Object-type Healthcare dataset, and with culturalEvents for the Object-type Smart Cities one). All Healthcare (merged) and All Smart Cities (merged) datasets refer to the results when both data-type (numerical) and object-type properties are considered and a merged top-k ranking is taking place for the final notification to the subscriber.

$RCC\_F-score$ depicts the overlap between the conceptual ground truth and the approach's conceptual clusters. It is shown that in almost all datasets PoSSUM is the best approach, indicating that it forms the closest to the original conceptual clusters. The worst results are for the FACES dataset since it contains the highest conceptual diversity; nevertheless, it still achieves $RCC\_F-score = 0.69$, 27% more than that of FACES. PubSum's performance was close to PoSSUM's one, showing that when high conceptual diversity is involved, then, embedding models can prove superior to thesauri/ontologies used in FACES. Similar behaviour of the results is depicted in all object-type datasets, since they contain the FACES one, with the embedding-based approaches performing up to 2 times better than FACES. The biggest difference is depicted in the numerical datasets, with PoSSUM performing almost 2 times better than the other approaches for both use cases. Nevertheless, the numerical datasets have less conceptual diversity; therefore, even a single erroneous conceptual cluster could significantly affect the final $RCC\_F-score$. The advantage of PoSSUM over the other approaches is also depicted in the merged datasets (All) with the Healthcare use case having the best results ($RCC\_F-score = 0.818$), followed by Smart Cities ($RCC\_F-score = 0.720$). The poor results of FACES regarding $RCC\_F-score$, but its good results for QUD, prove that only evaluating a summary's quality based on humans should not be the only criterion for the performance of all steps of an algorithm, especially, when semantic and conceptual diversity are concerned.

$RS\_F-score$ depicts the loss of non-conceptually similar information in a summary due to top-k ranking. This metric is partly affected by the $RCC\_F-score$ as the better the latter is, the more diversity the final summary will have; therefore, the lower the loss of non-conceptually similar information. This is shown for PoSSUM that has the best results for k = 10, and in most cases, for k = 5.

Table 6.7: Average RS_F-score and RCC_F-score

Dataset	**RS_F-score**						**RCC_F-score**		
	FACES		**PubSum**		**PoSSUM**		**FACES**	**PubSum**	**PoSSUM**
	k=5	**k=10**	**k=5**	**k=10**	**k=5**	**k=10**			
FACES	0.796	0.739	0.823	0.838	**0.833**	**0.863**	0.429	0.657	**0.690**
Healthcare									
Numerical Healthcare	N/A	N/A	0.740	**0.533**	**0.900**	**0.533**	0.500	0.444	**0.833**
Object-type Healthcare	N/A	N/A	**0.932**	0.939	0.923	**0.942**	0.434	**0.805**	0.803
All Healthcare (merged)	N/A	N/A	0.899	0.907	**0.917**	**0.919**	0.467	0.625	**0.818**
Smart Cities									
Numerical Smart Cities	N/A	N/A	**0.949**	0.949	**0.949**	**0.956**	0.333	0.439	**0.715**
Object-type Smart Cities	N/A	N/A	0.819	0.831	**0.839**	**0.862**	0.362	0.671	**0.725**
All Smart Cities (merged)	N/A	N/A	**0.912**	0.906	0.910	**0.925**	0.347	0.555	**0.720**

Nevertheless, $RS\_F-score$ is also partly affected by the triple ranking; therefore, PubSum may be slightly better than PoSSUM in the case of Numerical Smart Cities for top-5, affecting in this way the overall result (All Smart Cities (merged)), even though it has worse $RCC\_F-score$ than PoSSUM. Specifically, in this case, PubSum picked the most diverse triples, even from erroneous conceptual clusters, during triple ranking. The worst approach is FACES, supported also by the fact it had the worst $RCC\_F-score$. In general, the lower the k, the lower the $RS\_F-score$ as the information filtering is stricter. The cases where this is not applied are when the number of conceptual clusters is less than k; therefore, more conceptual similarity will exist in the final summary (Numerical Healthcare).

Processing Time, End-to-End Latency and Throughput

The processing time, end-to-end latency and throughput results are shown in Table 6.8 and Table 6.9, respectively. Table 6.8 presents the processing time of the conceptual clustering and the triple ranking. The window processing time is the overall time it takes from the first element that reaches the window to the window's final summary creation. The window time contains the other times, hence it is the longest. The times are shown for a small (windowSize = 50) to a large window (windowSize = 500) to observe how the size affects the performance. It is seen that the fastest approach is PoSSUM for all cases, apart from the case of windowSize = 50 in Smart Cities, where PubSum is slightly overall faster. Nevertheless, PoSSUM has the fastest time in clustering and scoring on all occasions. This means that even though it contains two additional steps (aggregation and reasoning), the semi-incremental power of DBVARC clustering and the simplification of Triple2Rank have contributed to better overall processing time. Specifically, PoSSUM takes half or even less time than PubSum in clustering, and it is slightly better for windowSize = 50, but significantly better for bigger windows than PubSum in scoring. Both approaches are significantly faster than FACES. In general, bigger windows perform more slowly since the window is waiting for longer for its maximum size to be reached, and the triples are more in number when clustering and scoring occurs. It is also observed that usually, clustering takes more time than scoring for PoSSUM, but the case is reversed for PubSum and FACES.

Similar behaviour is depicted in Table 6.9 (no embedding model loading time to the system has been included in the latency as opposed to Chapter 4). It is seen

Table 6.8: Processing time performance (ms) for an entity

Metric	**FACES**		**PubSum**		**PoSSUM**	
	window size 50	**window size 500**	**window size 50**	**window size 500**	**window size 50**	**window size 500**
Healthcare						
Window processing time	63631	413784	24975	257384	**24430**	**249508**
Clustering processing time	118.067	727.267	6.033	32.267	**1.933**	**18.667**
Scoring processing time	483.933	1726.133	6.333	355.733	**5.433**	**15.400**
Smart Cities						
Window processing time	67809	373500	**24267**	254146	24989	**250827**
Clustering processing time	156.633	1378.867	10.533	47.000	**5.533**	**25.733**
Scoring processing time	301.133	1101.733	8.433	290.667	**4.567**	**18.267**

Table 6.9: Average latency (ms) and throughput (events per second) for ten entities

Metric	**No top-k**	**FACES**		**PubSum**		**PoSSUM**	
		window size 50	**window size 500**	**window size 50**	**window size 500**	**window size 50**	**window size 500**
Healthcare							
End-to-end latency	**21265**	1118700	1234488	279675	**308622**	**279101**	310203
Throughput in	**235.109**	183.451	174.943	233.733	231.962	**234.089**	**234.267**
Throughput out		2.083	0.105	35.000	**33.333**	**35.250**	**33.333**
Smart Cities							
End-to-end latency	**19986**	753794	904376	178182	259393	**171317**	**204688**
Throughput in	**913.600**	620.791	602.246	602.251	592.259	**871.132**	**866.294**
Throughput out		1.083	0.051	18.750	16.667	**28.083**	**25.000**

that the best latency and throughput is observed for No top-k since no additional analysis occurs, but when summarisation happens, the results are mostly better for PoSSUM. PubSum is generally worse than PoSSUM, but in the case of Healthcare, it is not significantly different. The worst approach is FACES with very poor results compared to the embedding-based approaches. In general, the bigger the window, the higher the latency, and the lower the throughput. The throughput is analogous to the latency; therefore, the longer the events are processed the fewer events the system can handle. This also explains the reason the throughput in is higher than the throughput out since the system may integrate more and more events in parallel to the summarisations, but the events that are put out in the form of a summary are dependent on the time these summarisations take. The latency and throughput are also dependent on the parallelism, which is shown in the different results between the Healthcare and Smart Cities use cases. Smart Cities has lower latency and higher throughput than Healthcare since more publishers take place. It should also be noted that the end-to-end latency is higher than the window processing time since it takes into account the timestamp of the earliest published event that is included in the summary. This means that the longer the events are generated, the higher the average end-to-end latency will be as the rate the events are generated is higher than the one they get consumed (throughput in is higher than throughput out).

Size Reduction and Memory Footprint

The size reduction and memory footprint results are shown in Table 6.10 and Table 6.11, respectively. According to Table 6.10, the No top-k approach might have the best latency and throughput, but since no summarisation takes place, all messages/events are sent to the subscriber. The summarisation approaches contribute to the reduction of messages by eliminating conceptually similar data. The bigger the window and the lower the k, the higher the reduction of messages. For example, for windowSize = 50, there is 90% message reduction for k = 5, and 80% for k = 10. For windowSize = 500, the reduction is 99% and 98%, respectively. Therefore, the filtering has an impact in reducing significantly the final number of forwarded messages, and instead, sending only an important and representative subset.

It should be noted, though, that the summarisation approaches come with a longer memory footprint as shown in Table 6.11. For example, FACES uses

Table 6.10: Number of forwarded messages for ten entities for the duration of 30 minutes

Metric	**no top-k**	**fusion top-k**			
		window size 50		**window size 500**	
		k = 5	**k = 10**	**k = 5**	**k = 10**
Message reduction	N/A	90%	80%	**99%**	**98%**
Healthcare					
Number of forwarded messages	423196.200	42319.620	84639.240	**4231.962**	**8463.924**
Smart Cities					
Number of forwarded messages	1644480	164448	328896	**16444.800**	**32889.600**

the whole DBpedia for the triple ranking with 11.8GB of memory only for the relevant triples, whereas PubSum uses GOOGLENEWS Word2Vec that holds 3.35GB. PoSSUM holds the lowest memory since it uses ConceptNet, which has 1.09GB of memory for the English version. Another important factor in the comparison among the approaches is that PoSSUM does not demand any pre-defined parameters; therefore, no tuning is necessary making it easier to be applied in streaming environments. On the other hand, both FACES and PubSum are dependent on parameter tuning, which hinders the overall performance of a dynamic system.

Table 6.11: Memory footprint and parameters' number of approaches

Approach	**Model/Ontology**	**Memory Consumption**	**Number of parameters**
FACES	DBpedia (relevant triples)	11.8GB	2
PubSum	Word2Vec GOOGLENEWS	3.35GB	2
PoSSUM	ConceptNet (english)	**1.09GB**	**0**

6.6 Summary

Existing approaches have tried to resolve requirements regarding usability, user expressibility, data expressiveness, user effectiveness, data effectiveness, and efficiency. Nevertheless, no existing approach covers all requirements well. Diverse entity summarisation approaches create sophisticated conceptually and contextually diverse summaries of high usability, but they are static methodologies that cannot be directly applied to IoT environments, may rely on thesauri and ontologies associated with aforementioned limitations, and do not address abstractive summaries or deal with temporal numerical data. Conceptual clustering approaches address conceptual and contextual diversity, but they are bound by representational coupling and are static, computationally expensive, and memory-heavy methodologies with a plethora of manually-tuned parameters. Stream clustering approaches create on-the-fly clusters, but they involve offline initialisation phases and computationally inefficient cluster enhancements, they do not apply to rich entity-centric graphs, and their assumption on data distribution, shape of clusters, and pre-defined parameters is limited. Pub/Sub approaches producing top-k diverse notifications only address duplication or commonality among attributes/terms in relation to redundancy, and they do not apply to rich entity-centric graph data. Pub/Sub approaches performing approximate semantic matching of medium usability do not address redundancy of any form, and summarisation is out of scope. The same applies to Pub/Sub approaches semantically enriching events, with the additional disadvantage that ontologies or domain experts are also used. The PubSum approach (explained in Chapter 4) does not address abstractive summaries or deal with temporal numerical data, it is based on a static parameter-tuned clustering approach and it could be improved in terms of user effectiveness, data effectiveness, and efficiency.

In this chapter, the PoSSUM approach is proposed, which is a novel dynamic extractive and abstractive diverse summarisation methodology for heterogeneous numerical and Linked Data entity graph streams, alike. The approach consists of two phases: 1) a novel embedding and density-based conceptual clustering that is parameter-free, partly incremental, and it can be used for general streaming applications, and 2) a contextual-based top-k ranking of the conceptual clusters related to user query importance, informativeness, and diversity. It also includes approximation along with reasoning to provide high-level interpretation of the numerical sensor data. The approach creates graph-based diverse extractive and abstractive entity summaries (notifications) for representationally-decoupled subscriptions. It is evaluated for two real-world use cases; Healthcare and Smart Cities, along with proposed ground truths and metrics as well as baselines including typical Pub/Sub, PubSum, and FACES, a thesaurus/ontology-based diverse entity summarisation approach. The results include data diversity user desire up to 80%, best summary quality for more than half of the entities by a large margin, best conceptual clustering F-score ranging from 0.69 to 0.83, redundancy-aware F-score of up to 0.95, up to 99% message reduction, half or even less clustering processing time and significantly faster scoring, comparable end-to-end latency and throughput, and consumption of a third of the memory. PoSSUM is parameter-free; therefore, no prior parameter tuning is needed in contrast to the other approaches.

Chapter 7

Conclusions and Future Work

7.1 Thesis Summary

Internet of Things (IoT) is an emerging and very promising technology; however, it involves a number of challenges regarding data, user, and system (Chapter 2). Data challenges involve heterogeneity deriving from data with different representations (schemata and semantics) that cause issues with interoperability, redundancy due to duplication and conceptual similarity that overwhelms users and systems, and enrichment necessitation of raw sensor data with data from external static sources to provide a more complementary contextual information. User challenges involve high-level interpretation necessitation (reasoning) of raw data to provide more meaningful information, need of a user-friendly data representation of the final information, and facilitation of non-technical users by providing representationally-decoupled and simple queries that cover as much complete and diverse information. System challenges involve scalability, timeliness and resource constraints by proposing systems that can handle dynamism, high sensor volume and data velocity, data continuity with low latency, high throughput, and low memory consumption.

Existing event-based systems can ease the system challenges, but fall short in terms of data and user ones (Chapter 2). An emphasis is given in Publish/Subscribe systems (Pub/Sub) due to their best trade-off among system performance capabilities and simplicity, compared to other systems like Complex Event Processing (CEP) or data stream processing models. Existing Pub/Sub schemes have either expressible but non-usable and representationally-coupled subscription models or they provide redundant and irrelevant notifications that may overload systems and users. On the other hand, semantic stream processing approaches provide semantically enriched information at a conceptual or contextual level and deal with interoperability issues with the use of ontologies or linguistic resources. Although these approaches satisfy the data representation, semantic annotation, reasoning, interlinking, and high-level abstraction needs of the users, they suffer from complexity, inefficiency, representational coupling, necessitation of advanced analytics, low usability, and domain dependency.

This thesis explores the aforementioned challenges, limitations, and gaps that lead to the formulation of two research questions related to the requirements of usability, user expressibility, data expressiveness, user and data effectiveness, and system efficiency. These are addressed by proposing a new Pub/Sub scheme; the *Entity-centric Publish/Subscribe Summarisation System* (Chapter 3) that involves user-friendly contextually-aware subscriptions as well as extractive and abstractive summarisation approaches for the publications.

Existing diverse entity summarisation approaches are highly-usable and capable of creating expressive summaries; however, they are static and inefficient, may rely on rigid thesauri and ontologies, and are evaluated subjectively. Existing Pub/Sub systems address notification redundancy only in the sense of duplication or commonality among attributes/terms without exploring entity-centric data. Pub/Sub systems exploring approximate semantic matching contain semantically-decoupled subscriptions, but they are of medium usability and do not address redundancy, summarisation or entity-centric data. Finally, Pub/Sub systems containing semantic engines rely on rigid ontologies to enrich data. This thesis proposes PubSum (Chapter 4) to address the aforementioned gaps, which is a dynamic diverse summarisation methodology for heterogeneous Linked Data streams that is based on an embedding-based density clustering and a geometric-based top-k ranking.

Existing approximation approaches are efficient in resource-constraint environments, but they may assume incorrect data distributions, normalise data, use fixed parameters that affect the effectiveness, and do not observe data fluctuations. Approaches related to high-level abstractions are capable of creating expressive notifications; however, they may be static and inefficient, supervised, applicable to one use case, and complex for real-time systems due to the use of advanced machine learning techniques. Other data expressive approaches, based on semantic rule engines and rule-based reasoning, suffer from rigid ontologies, representational coupling, and complex queries. This thesis proposes IoTSAX (Chapter 5) to address the aforementioned gaps, which is a dynamic abstractive summarisation methodology for heterogeneous numerical entity graph streams based on an enhanced dynamic symbolic approximation and approximate rule-based reasoning.

PubSum (Chapter 4) is a highly usable, effective, and efficient diverse entity summarisation approach; however, data enrichment and high-level abstraction are not addressed, the efficiency and effectiveness could be improved further, and temporal numerical sensor data are not explored. Conceptual clustering and stream clustering approaches cannot be directly used for dynamic entity summarisation as they mostly suffer from complexity, inefficiency, a high number of parameters, and do not apply to entity-centric data. This thesis proposes PoSSUM (Chapter 6) to address the aforementioned gaps, which is a dynamic extractive and abstractive diverse summarisation methodology for heterogeneous numerical and Linked Data entity graph streams. PoSSUM is based on an embedding-based and density-based conceptual clustering, and a contextual-based top-k ranking.

The ability of the PubSum, IoTSAX, and PoSSUM approaches to address the research questions has been extensively evaluated by exploring two real-world use cases; Healthcare and Smart Cities. A range of synthetic and real-world data was analysed with characteristics involving heterogeneity in data types, semantics, concepts, and contexts. Ground truths and metrics were constructed, and along with existing ones, they showed that PubSum, IoTSAX, and PoSSUM achieve effective and efficient entity summaries that are expressive through conceptual and contextual diversity as well as high-level abstractions, while requiring simple and highly usable subscriptions with minimal configuration settings for further user expressibility.

7.2 Contributions

The core contributions of the thesis are the following:

Entity-centric Publish/Subscribe Summarisation System: A new entity-centric Pub/Sub scheme is proposed that combines the advantages of the topic-based and graph-based Pub/Sub schemes. The proposed subscription model is highly usable since only the entity name needs to be known a-priori, while the user expressibility is boosted by minimal and simple parameters at no expense of simplicity or representational decoupling. Hence, no a-priori data awareness is needed or expertise in query languages. The proposed event model is simple and understandable to the users, and it can be adapted based on the notification type, that is extractive or abstractive summary, by providing rich conceptual and contextual information. The system also supports windowing, data fusion, high-level abstraction, approximation, conceptual clustering, and top-k ranking that result in expressive entity summaries of high quality using limited resources, and processing time that do not overwhelm the users or the system with redundancy and information overload. This work has been published in the papers Pavlopoulou and Curry [68–72].

PubSum Approach: The aim of the proposed PubSum approach is to create diverse extractive summaries of heterogeneous Linked Data streams, that is triples with entities as objects. It is achieved by processing three main elements; *Embedding-based Triple Vectors*, *DBSCAN Clustering*, and *Geometric Ranking*. The approach proposes novel ways of transforming triples into vectors and ranking the triples within conceptual clusters since it is based on superior word embedding models and window-based methodologies relying on a combination of similarity metrics and some pre-defined rules. Hence, it behaves better compared to state-of-the-art diverse entity summarisation approaches that rely on rigid ontologies or other linguistic resources and contain static ranking implementations. In the end, a diverse set of information (extractive summary) is presented as a notification to the subscribers that does not overwhelm them or the processing system, and it caters for the users' different interpretations. Overall, the evaluation shows that although it behaves slightly worse in summary quality, it achieves slightly better redundancy-awareness, 6 times better latency, 3 times better memory and up to 33% better throughput, compared to state-of-the-art diverse entity summarisation approaches. Also, although it is up to 6 times slower than typical Pub/Sub systems, it achieves up to 92% message reduction and slightly similar throughput. This work has been published in the papers Pavlopoulou and Curry [68–70].

IoTSAX Approach: The aim of the proposed IoTSAX approach is to create abstractive summaries of numerical graph streams, that is triples with numerical objects. It is achieved by processing two main elements; *Data Approximation* and *Reasoning*. The approximation consists of a novel methodology that enhances the original Symbolic Aggregate Approximation (SAX) algorithm. Specifically, the proposed *Dynamic PAA* defines dynamic segment points that observe and retain the sharp fluctuations in the data due to extreme/abnormal values. This improves the effectiveness of the original SAX that contains fixed and rigid segments. The proposed *Dynamic SAX* defines a dynamic alphabet size that leads to more fine-grain symbolic representations of the approximate data, based on its original skewness, compared to the original SAX that loses the true data distribution and contains a fixed and rigid alphabet size. The reasoning consists of a novel methodology *Approximation Interpretation* that interprets the temporal pattern or shape of the symbolic representation and a novel *Approximate Reasoning Rules* algorithm that imposes representational decoupling capabilities in reasoning rules by introducing semantic relatedness with the use of word embedding models. The latter improves on approaches that either use rigid and domain-dependent rules or non-flexible ontologies. In the end, large collections of numerical values regarding attributes are transformed into abstractive summaries containing the above information. These are presented as a notification to the subscribers that does not overwhelm them or the processing system. Overall, the evaluation shows that it achieves good approximate reasoning effectiveness. Also, it is almost similar in latency and throughput, but 2 to 3 times better in approximation error and up to 13% better in compression space-saving percentage, when data redundancy occurs, compared to the efficient SAX approximation. Furthermore, although it is up to 3 times slower than typical Pub/Sub systems, it achieves up to 98% message reduction and slightly similar throughput. This work has been published in Pavlopoulou and Curry [71] (early access).

PoSSUM Approach: The aim of the proposed PoSSUM approach is to create diverse extractive and abstractive summaries of heterogeneous enriched graph streams, that is triples with entities and numerical values as objects. It is achieved by processing three main elements; *Embedding-based Triple Vectors*, *DBVARC Clustering*, and *Triple2Rank*. The approach proposes a novel Density-Based VARiance Clustering (DBVARC) algorithm that is better in terms of other existing conceptual or stream clustering approaches since it is parameter-free and partly incremental. The algorithm is also based on the notion of refining clusters based on intra-connected and inter-connected density, which is normally not addressed by most approaches. A novel triple ranking methodology is also proposed that is based on the notion of taxonomies to judge the importance of each triple within a conceptual cluster. Data enrichment and simple reasoning capabilities are also supported. In the end, an enriched diverse set of information (extractive and abstractive summary) is presented as a notification to the subscribers that does not overwhelm them or the processing system, and it caters for the users' different interpretations. Overall, the evaluation shows that it achieves the best summary quality for more than 50% of the entities by a large margin, up to 2 times better data effectiveness, up to 63% less processing time, up to 4 times better latency, up to 44% better throughput, and 3 times better memory, compared to state-of-the-art diverse entity summarisation approaches. Also, although it can be up to 92% slower than typical Pub/Sub systems, depending on the window size, it achieves up to 99% message reduction. This work has been published in Pavlopoulou and Curry [72] (in press).

Evaluation Methodology: A novel evaluation methodology is proposed that goes beyond the typical entity summarisation evaluation that examines only user effectiveness based on judges (often small in number) that could be highly subjective. The goal of the diverse entity summarisation is not only to examine the satisfaction of users, but to also observe the coverage of the wide range of different concepts and contexts, the redundancy awareness, and the retaining of the most representative information within a summary. For this purpose, two ground truths (relevance and conceptual) were constructed based on semantically extending original datasets with the use of thesauri and ontologies, and observing the contextual coherence, direct word links, and sub-categories in taxonomies. Also, several novel metrics were proposed like diversity consensus, conceptual clustering F-score, and concept-based redundancy-aware F-score. Another important aspect is the efficiency evaluation in systems like Pub/Sub, which is not addressed by the typical entity summarisation evaluation. The proposed evaluation methodology uses a range of other effectiveness and efficiency metrics on a formulated real-world dataset related to the domains of Healthcare and Smart Cities as well as Linked Data deriving from DBpedia characterised by heterogeneity in data types, semantics, concepts, and contexts. Baselines were also identified and adapted to the proposed Entity-centric Publish/Subscribe Summarisation System. This work has been published or submitted in all of the aforementioned publications.

Literature Review and Gap Analysis: Related work has been analysed and projected against the thesis' requirements related to usability, user expressibility, data expressiveness, user and data effectiveness, and efficiency. A gap in the literature has been detected since no existing Pub/Sub system or approach can equally satisfy the aforementioned requirements. Existing diverse entity summarisation approaches are highly-usable and expressive, but they are static and inefficient, and are evaluated subjectively. Pub/Sub systems do not explore entity-centric data and either address only duplication redundancy or contain semantically-decoupled subscriptions, but of medium usability. Approximation approaches are efficient, but lack effectiveness. Approaches related to high-level abstractions are expressive, but are inefficient or representationally-coupled. Conceptual clustering and stream clustering approaches cannot be directly applied to entity-centric data and suffer from complexity as well as a high number of parameters. Most of the approaches above rely on rigid ontologies or other linguistic resources. This work has been published or submitted in all of the aforementioned publications.

7.3 Conclusions

The main conclusion of the thesis is that the proposed *Entity-centric Publish/Subscribe Summarisation System* containing the PubSum, IoTSAX, and PoSSUM approaches, which create dynamic extractive and abstractive diverse entity summaries for heterogeneous numerical and Linked Data entity graph streams, alike, based on embedding-based density clustering, geometric-based or contextual-based top-k ranking, dynamic symbolic approximation, and approximate rule-based reasoning, can effectively and efficiently address the thesis' research questions. This new Pub/Sub scheme outperforms existing Pub/Sub schemes that are bound by low to medium usability and representationally-coupled subscription models, or they provide redundant and irrelevant notifications that may overload systems and users. The approaches outperform state-of-the-art diverse entity summarisation approaches that are bound by inefficiency, rigid ontologies and linguistic resources, and subjective evaluation as well as efficient symbolic approximation ones that lack dynamism and correctness.

The proposed system and its approaches address the following research questions:

RQ1: Can a Pub/Sub be created that offers usability *(R1)* while addressing users' expressibility *(R2)* effectively *(R4)* and efficiently *(R6)*?

- **RQ1.1**: can a simple abstract representationally-decoupled subscription be defined that relies on expert and non-expert users alike *(R1)*?

- **RQ1.2**: can a subscription be defined that its notification will cover a range of different human interpretations independent of its complexity and with minimal configuration settings *(R2)*?
- **RQ1.3**: can a subscription be defined that its notification will not overwhelm the users with data overload *(R2)*?
- **RQ1.4**: can the satisfaction of users based on the received notifications be evaluated according to: (1) how well they address the users' different needs and interpretations *(R4)*?, (2) how much they reduce the information overload to the users *(R4)* and to the system *(R6)*?

This research question has been addressed by the following:

Subscription Model: The proposed model supports *Diversity-aware subscriptions* and/or *Abstractive-aware subscriptions.* This model only requires a-priori users' knowledge of the entity name they are interested in, linking it to the simplicity, high usability, and representational decoupling traits of the topic-based Pub/Sub scheme (RQ1.1). At the same time, the model expects minimal and simple parameters related to windowing, entity type, and filtering level. These parameters help boost the subscription expressiveness in a simple and understandable way to the users. The notifications cover a range of different human interpretations through the selected windowing and entity type by the users, since these dictate that the data is enriched/integrated by multiple sources, and a conceptual and contextual diverse entity summarisation or high-level abstractions are provided (RQ1.2). The selected filtering level by the users dictates the top-k filtering that will occur in the final notification so that they are not overwhelmed by information overload (RQ1.3).

User Effectiveness: The notifications were evaluated on the quality and diversity consensus based on a ground truth from human judges (RQ1.4). The PubSum approach achieved promising summary quality, whereas the PoSSUM approach achieved up to 80% data diversity desire and the best summary quality for most entities.

Data Effectiveness: The notifications were evaluated on the redundancy-aware F-score based on two proposed ground truths, duplication reduction, approximation error, and reasoning F-score (RQ1.4). The PubSum approach achieved up to 0.95 redundancy-aware F-score, and up to 69.3% duplication reduction, the IoTSAX approach achieved 2 to 3 times better approximation error and up to 0.87 reasoning F-score, whereas the PoSSUM approach achieved up to 0.95 redundancy-aware F-score.

Efficiency: The notifications were evaluated on the message reduction and compression space-saving percentage (RQ1.4). The PubSum approach achieved up to 92% message reduction, the IoTSAX approach achieved up to 98% message reduction and compression space-saving percentage ranging from 71.75% to 94.99%, whereas the PoSSUM approach achieved up to 99% message reduction.

RQ2: Can a Pub/Sub be created that offers expressiveness of heterogeneous data *(R3)* effectively *(R4, R5)* and efficiently *(R6)*?

- **RQ2.1**: can a methodology be defined for integrating data from multiple sources *(R3)*?
- **RQ2.2**: can an appropriate publication structure of integrated data be defined that is also understandable to the users *(R3)*?
- **RQ2.3**: can a methodology be defined for semantically abstracting integrated data and providing rich notifications with conceptual and contextual diversity or high-level abstractions independent of domain *(R3)*?
- **RQ2.4**: can the semantically-abstracted rich notifications be evaluated according to: (1) how well they cover the wide range of different concepts and contexts *(R5)*?, (2) how well they reduce information redundancy without sacrificing important information *(R5)*?, 3) how much they boost the system's performance *(R6)*, 4) how many dependencies are needed (e.g. domain experts, external ontologies, memory-heavy models etc.) *(R4, R5, R6)*?

This research question has been addressed by the following:

Data Integration: The proposed system supports a range of windowing policies on a per-entity basis so that complementary information regarding an entity is provided by integrating data from real-time sources and external static ones (RQ2.1).

Event Model: The proposed model supports Resource Description Framework (RDF) graphs that represent conceptual entities with their associated extractive

(RDF triples) and/or abstractive (RDF quads) properties or background information (RQ2.2).

PubSum Approach: The proposed approach is a dynamic diverse summarisation methodology for heterogeneous Linked Data streams that is based on an embedding-based density clustering and a geometric-based top-k ranking (RQ2.3).

IoTSAX Approach: The proposed approach is a dynamic abstractive summarisation methodology for heterogeneous numerical entity graph streams that is based on an enhanced dynamic symbolic approximation and approximate rule-based reasoning (RQ2.3).

PoSSUM Approach: The proposed approach is a dynamic extractive and abstractive diverse summarisation methodology for heterogeneous numerical and Linked Data entity graph streams, alike, that is based on an embedding-based and density-based conceptual clustering, and a contextual-based top-k ranking (RQ2.3).

Data Effectiveness: The notifications were evaluated on the conceptual clustering F-score and redundancy-aware F-score based on two proposed ground truths, duplication reduction, approximation error, and reasoning F-score (RQ2.4). The PoSSUM approach achieved a conceptual clustering F-score ranging from 0.69 to 0.83. The other metrics are covered above.

Efficiency: The notifications were evaluated on the end-to-end latency, memory, throughput, processing time, message reduction, and compression space-saving percentage (RQ2.4). The PoSSUM approach achieved latency ranging from 29,237ms to 187,395ms and 3 times less memory, and throughput ranging from 833 to 1,005 events/second, the IoTSAX approach achieved latency ranging from 37,395ms to 69,414ms, and throughput ranging from 13.231 to 97.393 events/second, whereas the PoSSUM approach achieved clustering processing time ranging from 1.933ms to 25.733ms, scoring processing time ranging from 4.567ms to 18.267ms, latency ranging from 171,317ms to 310,203ms, throughput ranging from 25 to 871.132 events/second, and a third of the memory. The other metrics are covered above.

7.4 Limitations

The main limitations of this thesis' work are analysed below.

PubSum Approach: The work is based on DBSCAN clustering in order to define dense regions in the vector space that form conceptual clusters. This algorithm is superior to other popular clustering approaches due to the lack of a pre-defined number of clusters parameter, and restrictions on the size and the shape of the clusters. However, it is a static algorithm resulting in the introduction of extra time processing cost, and it needs two manually-tuned parameters that can significantly affect the quality of the formed cluster. Manually tuning parameters are not applicable for IoT environments. Also, the clustering and scoring occur in batch for each window, introducing extra time processing cost. Another issue is that summarisation always introduces loss of important information; therefore, even if the summary quality is high, there will always be some users that will not be fully satisfied. Finally, the work only applies to triples with object-type properties.

IoTSAX Approach: The proposed Dynamic Piecewise Aggregation Approximation (PAA) algorithm defines dynamic segment points that observe and retain the sharp fluctuations in the data due to extreme/abnormal values. However, the effectiveness in observing these fluctuations and efficiency in saving space are dependent on the user that defines through a threshold how strict or relaxed the compression should be, without this meaning that the approach is bound by manually-tuned parameters. Another issue is that the approximate reasoning rules may introduce some false positives since the approximation is based on a restrictive top-1 similarity between the words of embedding models and the ones in the reasoning rules. Finally, each sensor is analysed in isolation from the other sensors without examining correlations, or conceptual and contextual relatedness.

PoSSUM Approach: The work is based on DBVARC clustering in order to define dense regions in the vector space that form conceptual clusters. Although this is a parameter-free algorithm, it may be too strict for sparse elements as it focuses on strict densely-connected regions. This also suggests that the approach is affected by the curse of dimensionality, that is if the dimensionality increases, but the volume of the elements stays the same, then, there may be no points that overlap within a region to form a conceptual cluster. Another issue is that one part of clustering and the scoring occur in batch for each window, introducing extra time processing cost. Finally, the loss of important information due to summarisation, applies to this approach as well.

Word Embedding Models Dependency: The work relies on word embedding models in order to turn triples into vectors and it uses these vectors, in return, to discover conceptually and contextually related triples. Although word embedding models are superior to other linguistic resources, their quality and accuracy are highly dependent on the corpora they have been trained on. Also, some words, phrases, or even contexts might be missing in the models. Another issue is that the models that use concept hierarchy to learn sense vectors for multi-sense words may map a vector to a word that has another sense; therefore, changing its real context. Finally, the way the triples are transformed into vectors by averaging the predicates' vectors with the objects' type ones, may introduce some error regarding the real context of the whole triple.

Typing Information Dependency: The work relies on the fact that triples with object-type properties contain a-priori typing information. Although the work presents a methodology for extracting typing information in the case of missing or erroneous types, there is no support for finding this information in a streaming and sophisticated fashion. In the case no typing information can be found for a triple, then, the triple cannot be transformed into a vector.

7.5 Future work

The main future directions of this thesis' work are analysed below.

Event Matching: The thesis' work focuses on single entity exact matching. An interesting future direction would be to support either multi-entity matching or domain-specific matching, by also adapting the proposed subscription model. In the first case, a Pub/Sub could detect correlations and relatedness among entities, and it could provide a summarised view of the common information among the entities. For example, in Healthcare's use case, multiple patients could be observed that are in close proximity for commonalities in their summarised health status, to detect possible epidemics. Static methodologies have explored multi-entity summarisation; therefore, they would need to be investigated on how to be adapted to Pub/Sub systems. In the second case, a Pub/Sub system could support a plethora of domains and a subscriber could have more targeted information if interested in a specific domain, including its sub-domains. Topic modelling methodologies along with taxonomy-based concepts could be explored for this purpose.

Event Types: The thesis' work focuses on graphs with object-type and data-type (numerical) properties. An interesting future direction would be to explore how summarisation could be achieved when events contain a plethora of other data types. These could range from simpler forms, like text, to more advanced ones, like multi-depth graphs, and images or videos. Existing works have explored video and image processing in CEP, but these works were not designed with entity summarisation in mind. Entity summarisation is a complex matter that in this case could be achieved by exploring methodologies related to graph centrality and static video summarisation.

Density-based Clustering: The thesis' work proposes a parameter-free and partly-incremental DBVARC clustering in order to define dense regions in the vector space that form conceptual clusters. An interesting future direction would be to find a fully incremental approach that is still parameter-free and it does not suffer from the curse of dimensionality. This is a very complex problem that would need to provide mechanisms that are not only affected by the location of the elements in the vector space, and at the same time, they should be efficient. Recent work related to clustering in non-Euclidean spaces could be explored on resolving this issue.

Typing Information: The thesis' work proposes a methodology for extracting the typing information on entities if they have missing or erroneous types. An interesting future direction would be to find a streaming typing information extraction. In this way, approaches concerning entity relatedness or entity linking could be facilitated in terms of real-time capabilities. Existing work in Pub/Sub or CEP that has focused on data enrichment through knowledge bases or the Web could be explored for this purpose.

Advanced Pattern Recognition, Reasoning, and Optimisations: An interesting future direction would be to explore other advanced data analytics methodologies. One example would be finding similar patterns and correlations among different time series, in conjunction with conceptually-similar groupings of measurements generated by publishers. This has not been explored yet in the literature. Another example would be to perform reasoning by more sophisticated methodologies than manual reasoning rules. The main issue with this direction is that the approaches should be both effective and non-complex for IoT environments. Another example would be to explore how to group subscribers/subscriptions based on commonality or subsumption of subscriptions. Another example would be to find an incremental way to perform triple scoring, although this goes hand in hand with a fully incremental clustering. Finally, an example could be exploring decentralised network communications to support entity summarisation in Pub/Sub.

Bibliography

[1] Patrick Th Eugster, Pascal A Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The many faces of publish/subscribe. *ACM Computing Surveys (CSUR)*, 35(2):114–131, 2003.

[2] Mark Weiser. The computer for the 21st century. *ACM SIGMOBILE Mobile Computing and Communications Review*, 3(3):3–11, 1999.

[3] Ammar Rayes and Samer Salam. Internet of things from hype to reality. *Springer*, 2017.

[4] Edward Curry and Amit Sheth. Next-generation smart environments: From system of systems to data ecosystems. *IEEE Intelligent Systems*, 33(3):69–76, 2018.

[5] Edward Curry. *Real-time linked dataspaces: Enabling data ecosystems for intelligent systems.* Springer Nature, 2020.

[6] Aldous Huxley. Brave new world (1932). *London: Vintage*, 131, 1998.

[7] Jeretta Horn Nord, Alex Koohang, and Joanna Paliszkiewicz. The internet of things: Review and theoretical framework. *Expert Systems with Applications*, 133:97–108, 2019.

[8] Lu Tan and Neng Wang. Future internet: The internet of things. In *3rd International Conference on Advanced Computer Theory and Engineering*, volume 5, pages V5–376. IEEE, 2010.

[9] Ioan Szilagyi and Patrice Wira. Ontologies and semantic web for the internet of things-a survey. In *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, pages 6949–6954. IEEE, 2016.

[10] Michael Schukat, David McCaldin, Kejia Wang, Guenter Schreier, Nigel H Lovell, Michael Marschollek, and Stephen J Redmond. Unintended consequences of wearable sensor use in healthcare. *Yearbook of Medical Informatics*, 25(01):73–86, 2016.

[11] Seyed Sajad Mousavi, Michael Schukat, and Enda Howley. Traffic light control using deep policy-gradient and value-function-based reinforcement learning. *IET Intelligent Transport Systems*, 11(7):417–423, 2017.

[12] Fadele Ayotunde Alaba, Mazliza Othman, Ibrahim Abaker Targio Hashem, and Faiz Alotaibi. Internet of things security: A survey. *Journal of Network and Computer Applications*, 88:10–28, 2017.

[13] Shancang Li, Li Da Xu, and Shanshan Zhao. 5g internet of things: A survey. *Journal of Industrial Information Integration*, 10:1–9, 2018.

[14] Emiliano Sisinni, Abusayeed Saifullah, Song Han, Ulf Jennehag, and Mikael Gidlund. Industrial internet of things: Challenges, opportunities, and directions. *IEEE Transactions on Industrial Informatics*, 14(11):4724–4734, 2018.

[15] Li Da Xu, Wu He, and Shancang Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, 2014.

[16] Gianpaolo Cugola and Alessandro Margara. Processing flows of information: From data stream to complex event processing. *ACM Computing Surveys (CSUR)*, 44(3):1–62, 2012.

[17] Andreas Thalhammer. *Linked data entity summarization.* PhD thesis, Karlsruhe Institute of Technology, 2017.

[18] Yezheng Liu, Fei Du, Jianshan Sun, Yuanchun Jiang, Jianmin He, Tingting Zhu, and Chunhua Sun. A crowdsourcing-based topic model for service matchmaking in internet of things. *Future Generation Computer Systems*, 87:186–197, 2018.

[19] Yongrui Qin, Quan Z Sheng, Nickolas JG Falkner, Schahram Dustdar, Hua Wang, and Athanasios V Vasilakos. When things matter: A survey on data-centric internet of things. *Journal of Network and Computer Applications*, 64: 137–153, 2016.

[20] André Freitas, Juliano Efson Sales, Siegfried Handschuh, and Edward Curry. How hard is this query? measuring the semantic complexity of schema-agnostic queries. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 294–304, 2015.

[21] Michael Schukat, Pablo Cortijo Castilla, Hugh Melvin, and Fei Hu. Trust and trust models for the iot. In *Security and Privacy in Internet of Things (IoTs): Models, Algorithms, and Implementations*. CRC Press, 2016.

[22] Thanh Tran, Tobias Mathäß, and Peter Haase. Usability of keyword-driven schema-agnostic search. In *Extended Semantic Web Conference*, pages 349–364. Springer, 2010.

[23] André Freitas, Edward Curry, João Gabriel Oliveira, and Sean O'Riain. Querying heterogeneous datasets on the linked data web: challenges, approaches, and trends. *IEEE Internet Computing*, 16(1):24–33, 2011.

[24] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th International Conference on World Wide Web*, pages 22–32. ACM, 2005.

[25] Mahdi Bakhshi, Mohammadali Nematbakhsh, Mehran Mohsenzadeh, and Amir Masoud Rahmani. Data-driven construction of sparql queries by approximate question graph alignment in question answering over knowledge graphs. *Expert Systems with Applications*, 146:113205, 2020.

[26] Santosh Pattar, Rajkumar Buyya, Kuppanna Rajuk Venugopal, SS Iyengar, and LM Patnaik. Searching for the iot resources: fundamentals, requirements, comprehensive review, and future directions. *IEEE Communications Surveys & Tutorials*, 20(3):2101–2132, 2018.

[27] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[28] Aris M. Ouksel and Amit Sheth. Semantic interoperability in global information systems. *ACM SIGMOD Record Special Interest Group on Management of Data*, 28(1):5–12, 1999.

[29] Şefki Kolozali, Maria Bermudez-Edo, Nazli FarajiDavar, Payam Barnaghi, Feng Gao, Muhammad Intizar Ali, Alessandra Mileo, Marten Fischer, Thorben Iggena, Daniel Kuemper, et al. Observing the pulse of a city: A smart city framework for real-time discovery, federation, and aggregation of data streams. *IEEE Internet of Things Journal*, 6(2):2651–2668, 2018.

[30] George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, 1987.

[31] Friedrich Wilhelm Nietzsche. *The will to power*. Vintage, 1968.

[32] George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81, 1956.

[33] Ejaz Ahmed and Mubashir Husain Rehmani. Mobile edge computing: opportunities, solutions, and challenges. *Future Generation Computer Systems*, 70:59–63, 2017.

[34] Mansoor Nasir, Khan Muhammad, Jaime Lloret, Arun Kumar Sangaiah, and Muhammad Sajjad. Fog computing enabled cost-effective distributed summarization of surveillance videos for smart cities. *Journal of Parallel and Distributed Computing*, 126:161–170, 2019.

[35] Muhammad Salek Ali, Massimo Vecchio, Miguel Pincheira, Koustabh Dolui, Fabio Antonelli, and Mubashir Husain Rehmani. Applications of blockchains in the internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 21(2):1676–1717, 2018.

[36] Jasenka Dizdarević, Francisco Carpio, Admela Jukan, and Xavi Masip-Bruin. A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration. *ACM Computing Surveys (CSUR)*, 51(6):1–29, 2019.

[37] Payam Barnaghi, Wei Wang, Cory Henson, and Kerry Taylor. Semantics for the internet of things: early progress and back to the future. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(1):1–21, 2012.

[38] Daniel Puschmann, Payam Barnaghi, and Rahim Tafazolli. Adaptive clustering for dynamic iot data streams. *IEEE Internet of Things Journal*, 4(1):64–74, 2016.

[39] Souleiman Hasan and Edward Curry. Approximate semantic matching of events for the internet of things. *ACM Transactions on Internet Technology (TOIT)*, 14(1):2, 2014.

[40] K Prasanna Lakshmi and CRK Reddy. A survey on different trends in data streams. In *2010 International Conference on Networking and Information Technology*, pages 451–455. IEEE, 2010.

[41] Charu C Aggarwal. *Data streams: models and algorithms*, volume 31. Springer Science & Business Media, 2007.

[42] Daniel Puschmann, Payam Barnaghi, and Rahim Tafazolli. Using lda to uncover the underlying structures and relations in smart city data streams. *IEEE Systems Journal*, 12(2):1755–1766, 2017.

[43] Frieder Ganz, Payam Barnaghi, and Francois Carrez. Information abstraction for heterogeneous real world internet data. *IEEE Sensors Journal*, 13(10):3793–3805, 2013.

[44] Hoonseok Park and Jae-Yoon Jung. Sax-arm: Deviant event pattern discovery from multivariate time series using symbolic aggregate approximation and association rule mining. *Expert Systems with Applications*, 141:112950, 2020.

[45] Chaw Thet Zan and Hayato Yamana. Dynamic sax parameter estimation for time series. *International Journal of Web Information Systems*, 2017.

[46] Andreas Harth, Katja Hose, Marcel Karnstedt, Axel Polleres, Kai-Uwe Sattler, and Jürgen Umbrich. Data summaries for on-demand queries over linked data. In *Proceedings of the 19th International Conference on World Wide Web*, pages 411–420. ACM, 2010.

[47] Norberto Fernández, Jesús Arias, Luis Sánchez, Damaris Fuentes-Lorenzo, and Óscar Corcho. Rdsz: an approach for lossless rdf stream compression. In *European Semantic Web Conference*, pages 52–67. Springer, 2014.

[48] Charu C Aggarwal and S Yu Philip. A survey of synopsis construction in data streams. In *Data Streams: Models and Algorithm*, pages 169–207. Springer, 2007.

[49] Neha Gupta and Indrjeet Rajput. Stream data mining: a survey. *International Journal of Engineering Research and Applications*, 3:1113–1118, 2013.

[50] Kalpa Gunaratna, Krishnaparasad Thirunarayan, and Amit Sheth. Faces: diversity-aware entity summarization using incremental hierarchical conceptual clustering. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[51] Seyedamin Pouriyeh, Mehdi Allahyari, Krys Kochut, Gong Cheng, and Hamid Reza Arabnia. Combining word embedding and knowledge-based topic modeling for entity summarization. In *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, pages 252–255. IEEE, 2018.

[52] Qingxia Liu, Gong Cheng, and Yuzhong Qu. Deeplens: Deep learning for entity summarization. In *Proceedings of the Workshop on Deep Learning for Knowledge Graphs*, 2020.

[53] Qingxia Liu, Gong Cheng, Kalpa Gunaratna, and Yuzhong Qu. Entity summarization: State of the art and future challenges. *Journal of Web Semantics*, page 100647, 2021.

[54] Marina Drosou, Kostas Stefanidis, and Evaggelia Pitoura. Preference-aware publish/subscribe delivery with diversity. In *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, page 6. ACM, 2009.

[55] Lisi Chen and Gao Cong. Diversity-aware top-k publish/subscribe for text stream. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 347–362. ACM, 2015.

[56] Zeinab Hmedeh, Cedric du Mouza, and Nicolas Travers. Tdv-based filter for novelty and diversity in a real-time pub/sub system. In *Proceedings of the 19th International Database Engineering & Applications Symposium*, pages 136–145, 2015.

[57] Souleiman Hasan, Sean O'Riain, and Edward Curry. Approximate semantic matching of heterogeneous events. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, pages 252–263. ACM, 2012.

[58] Noura Alhakbani, Mohammad Mehedi Hassan, Mourad Ykhlef, and Giancarlo Fortino. An efficient event matching system for semantic smart data in the internet of things (iot) environment. *Future Generation Computer Systems*, 95:163–174, 2019.

[59] Milenko Petrovic, Haifeng Liu, and Hans-Arno Jacobsen. G-topss: Fast filtering of graph-based metadata. In *Proceedings of the 14th International Conference on World Wide Web*, pages 539–547, 2005.

[60] Christian Esposito, Massimo Ficco, Francesco Palmieri, and Aniello Castiglione. A knowledge-based platform for big data analytics based on publish/subscribe services and stream processing. *Knowledge-Based Systems*, 79:3–17, 2015.

[61] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE Communications Surveys & Tutorials*, 16(1):414–454, 2013.

[62] Charbel El Kaed, Imran Khan, Andre Van Den Berg, Hicham Hossayni, and Christophe Saint-Marcel. Sre: semantic rules engine for the industrial internet-of-things gateways. *IEEE Transactions on Industrial Informatics*, 14 (2):715–724, 2017.

[63] Amelie Gyrard, Christian Bonnet, Karima Boudaoud, and Martin Serrano. Lov4iot: A second life for ontology-based domain knowledge to build semantic web of things applications. In *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 254–261. IEEE, 2016.

[64] Edmond Jajaga and Lule Ahmedi. C-swrl: Swrl for reasoning over stream data. In *2017 IEEE 11th International Conference on Semantic Computing (ICSC)*, pages 395–400. IEEE, 2017.

[65] Frieder Ganz, Payam Barnaghi, and Francois Carrez. Automated semantic knowledge acquisition from sensor data. *IEEE Systems Journal*, 10(3):1214–1225, 2014.

[66] Wang Wei and Payam Barnaghi. Semantic annotation and reasoning for sensor data. In *European Conference on Smart Sensing and Context*, pages 66–76. Springer, 2009.

[67] Pierre-Joseph Proudhon. *What is property?: An inquiry into the principle of right and of government*, volume 1. BR Tucker, 1876.

[68] Niki Pavlopoulou and Edward Curry. Towards a window-based diverse entity summarisation engine in publish/subscribe systems. In *Proceedings of EYRE 19: 2nd International Workshop on Entity Retrieval.* ACM, 2019.

[69] Niki Pavlopoulou and Edward Curry. Using embeddings for dynamic diverse summarisation in heterogeneous graph streams. In *2019 First International Conference on Graph Computing (GC)*, pages 5–12. IEEE, 2019.

[70] Niki Pavlopoulou and Edward Curry. Dynamic diverse summarisation in heterogeneous graph streams: a comparison between thesaurus/ontology-based and embeddings-based approaches. *International Journal of Graph Computing*, 1(1):23–39, 2020.

[71] Niki Pavlopoulou and Edward Curry. Iotsax: A dynamic abstractive entity summarisation approach with approximation and embedding-based reasoning rules in publish/subscribe systems. *IEEE Internet of Things Journal*, 2021.

[72] Niki Pavlopoulou and Edward Curry. Possum: An entity-centric publish/subscribe system for diverse summarisation in internet of things. *ACM Transactions on Internet Technology*, 2021.

[73] Soulakshmee Devi Nagowah, Hatem Ben Sta, and Baby Ashwin Gobin-Rahimbux. An overview of semantic interoperability ontologies and frameworks for iot. In *2018 Sixth International Conference on Enterprise Systems (ES)*, pages 82–89. IEEE, 2018.

[74] Dennis Pfisterer, Kay Romer, Daniel Bimschas, Oliver Kleine, Richard Mietz, Cuong Truong, Henning Hasemann, Alexander Kröller, Max Pagel, Manfred

Hauswirth, et al. Spitfire: toward a semantic web of things. *IEEE Communications Magazine*, 49(11):40–48, 2011.

[75] Stephanie B Baker, Wei Xiang, and Ian Atkinson. Internet of things for smart healthcare: Technologies, challenges, and opportunities. *IEEE Access*, 5:26521–26544, 2017.

[76] Rustem Dautov, Salvatore Distefano, and Rajkumaar Buyya. Hierarchical data fusion for smart healthcare. *Journal of Big Data*, 6(1):1–23, 2019.

[77] Yazdan Ahmad Qadri, Ali Nauman, Yousaf Bin Zikria, Athanasios V Vasilakos, and Sung Won Kim. The future of healthcare internet of things: a survey of emerging technologies. *IEEE Communications Surveys & Tutorials*, 22(2):1121–1167, 2020.

[78] Charu C Aggarwal, Naveen Ashish, and Amit Sheth. The internet of things: A survey from the data-centric perspective. In *Managing and Mining Sensor Data*, pages 383–428. Springer, 2013.

[79] Marcello Ienca, Agata Ferretti, Samia Hurst, Milo Puhan, Christian Lovis, and Effy Vayena. Considerations for ethics review of big data health research: A scoping review. *PloS One*, 13(10):e0204937, 2018.

[80] Azana Hafizah Mohd Aman, Wan Haslina Hassan, Shilan Sameen, Zainab Senan Attarbashi, Mojtaba Alizadeh, and Liza Abdul Latiff. Iomt amid covid-19 pandemic: Application, architecture, technology, and security. *Journal of Network and Computer Applications*, page 102886, 2020.

[81] Rasheed Hussain and Sherali Zeadally. Autonomous cars: Research results, issues, and future challenges. *IEEE Communications Surveys & Tutorials*, 21 (2):1275–1313, 2018.

[82] Council of cities, 2021. URL `https://oascities.org/national-networks/`.

[83] Sajana Samarasinghe. Top 10 smart cities worldwide, 2021. URL `https://businesschief.com/top10/top-10-smart-cities-worldwide`.

[84] Edward Curry, Viktoriya Degeler, Eoghan Clifford, Daniel Coakley, Andrea Costa, Schalk-Jan Van Andel, Nick van de Giesen, and Christos

Kouroupetroglou. Linked water data for water information management. In *11th International Conference on Hydroinformatics*, 2014.

[85] Dan Puiu, Payam Barnaghi, Ralf Tönjes, Daniel Kümper, Muhammad Intizar Ali, Alessandra Mileo, Josiane Xavier Parreira, Marten Fischer, Sefki Kolozali, Nazli Farajidavar, et al. Citypulse: Large scale data analytics framework for smart cities. *IEEE Access*, 4:1086–1108, 2016.

[86] Bin Cheng, Salvatore Longo, Flavio Cirillo, Martin Bauer, and Ernoe Kovacs. Building a big data platform for smart cities: Experience and lessons from santander. In *2015 IEEE International Congress on Big Data*, pages 592–599. IEEE, 2015.

[87] John Soldatos, Nikos Kefalakis, Manfred Hauswirth, Martin Serrano, Jean-Paul Calbimonte, Mehdi Riahi, Karl Aberer, Prem Prakash Jayaraman, Arkady Zaslavsky, Ivana Podnar Žarko, et al. Openiot: Open source internet-of-things in the cloud. In *Interoperability and Open-source Solutions for the Internet of Things*, pages 13–25. Springer, 2015.

[88] Roland Stühmer, Yiannis Verginadis, Iyad Alshabani, Thomas Morsellino, and Antonio Aversa. Play: Semantics-based event marketplace. In *Working Conference on Virtual Enterprises*, pages 699–707. Springer, 2013.

[89] Shobharani Pacha, Suresh Ramalingam Murugan, and R Sethukarasi. Semantic annotation of summarized sensor data stream for effective query processing. *The Journal of Supercomputing*, pages 1–23, 2017.

[90] Antonio F Skarmeta, José Santa, Juan A Martínez, Josiane X Parreira, Payam Barnaghi, Shirin Enshaeifar, Michail J Beliatis, Mirko A Presser, Thorben Iggena, Marten Fischer, et al. Iotcrawler: Browsing the internet of things. In *2018 Global Internet of Things Summit (GIoTS)*, pages 1–6. IEEE, 2018.

[91] Joao Gama. A survey on learning from data streams: current and future trends. *Progress in Artificial Intelligence*, 1(1):45–55, 2012.

[92] Hongyan Liu, Yuan Lin, and Jiawei Han. Methods for mining frequent items in data streams: an overview. *Knowledge and Information Systems*, 26(1):1–30, 2011.

[93] César Cañas, Eduardo Pacheco, Bettina Kemme, Jörg Kienzle, and Hans-Arno Jacobsen. Graps: A graph publish/subscribe middleware. In *Proceedings of the 16th Annual Middleware Conference*, pages 1–12, 2015.

[94] Shobharani Pacha, Suresh Ramalingam Murugan, and R Sethukarasi. Semantic annotation of summarized sensor data stream for effective query processing. *The Journal of Supercomputing*, 76(6):4017–4039, 2020.

[95] Lefteris Zervakis, Christos Tryfonopoulos, Vinay Setty, Stephan Seufert, and Spiros Skiadopoulos. Towards publish/subscribe functionality on graphs. In *2nd International Workshop on Preservation of Evolving Big Data*. CEUR-WS.org, 2016.

[96] Yongrui Qin, Lina Yao, and Quan Z Sheng. Approximate semantic matching over linked data streams. In *International Conference on Database and Expert Systems Applications*, pages 37–51. Springer, 2016.

[97] Eli Fidler, Hans-Arno Jacobsen, Guoli Li, and Serge Mankovski. The padres distributed publish/subscribe system. In *8th International Conference on Feature Interactions in Telecommunications and Software Systems*, pages 28–30, 2005.

[98] Roger S Barga, Jonathan Goldstein, Mohamed Ali, and Mingsheng Hong. Consistent streaming through time: A vision for event stream processing. In *3rd Biennial Conference on Innovative Data Systems Research*, 2007.

[99] Alan J Demers, Johannes Gehrke, Biswanath Panda, Mirek Riedewald, Varun Sharma, Walker M White, et al. Cayuga: A general purpose event monitoring system. In *3rd Biennial Conference on Innovative Data Systems Research*, volume 7, pages 412–422, 2007.

[100] Asaf Adi and Opher Etzion. Amit-the situation manager. *The International Journal on Very Large Data Bases*, 13(2):177–203, 2004.

[101] Eugene Wu, Yanlei Diao, and Shariq Rizvi. High-performance complex event processing over streams. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, pages 407–418, 2006.

[102] Yanlei Diao, Neil Immerman, and Daniel Gyllstrom. Sase+: An agile language for kleene closure over event streams. *UMass Technical Report*, 2007.

[103] Haopeng Zhang, Yanlei Diao, and Neil Immerman. On complexity and optimization of expensive queries in complex event processing. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, pages 217–228, 2014.

[104] Gianpaolo Cugola and Alessandro Margara. Tesla: a formally defined event specification language. In *Proceedings of the Fourth ACM International Conference on Distributed Event-Based Systems*, pages 50–61, 2010.

[105] Darko Anicic, Paul Fodor, Sebastian Rudolph, and Nenad Stojanovic. Ep-sparql: a unified language for event processing and stream reasoning. In *Proceedings of the 20th International Conference on World Wide Web*, pages 635–644, 2011.

[106] Alexander Artikis, Anastasios Skarlatidis, François Portet, and Georgios Paliouras. Logic-based event recognition. *The Knowledge Engineering Review*, 27(4):469–506, 2012.

[107] Alexander Artikis, Alessandro Margara, Martin Ugarte, Stijn Vansummeren, and Matthias Weidlich. Complex event recognition languages: Tutorial. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, pages 7–10, 2017.

[108] Arvind Arasu, Shivnath Babu, and Jennifer Widom. The cql continuous query language: semantic foundations and query execution. *The VLDB Journal*, 15(2):121–142, 2006.

[109] Yijian Bai, Hetal Thakkar, Haixun Wang, Chang Luo, and Carlo Zaniolo. A data stream language and system designed for power and extensibility. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, pages 337–346, 2006.

[110] Andre Bolles, Marco Grawunder, and Jonas Jacobi. Streaming sparql-extending sparql to process data streams. In *European Semantic Web Conference*, pages 448–462. Springer, 2008.

[111] Davide Francesco Barbieri, Daniele Braga, Stefano Ceri, and Michael Grossniklaus. An execution environment for c-sparql queries. In *Proceedings of the 13th International Conference on Extending Database Technology*, pages 441–452, 2010.

[112] Jean-Paul Calbimonte, Oscar Corcho, and Alasdair JG Gray. Enabling ontology-based access to streaming data sources. In *International Semantic Web Conference*, pages 96–111. Springer, 2010.

[113] Danh Le-Phuoc, Minh Dao-Tran, Josiane Xavier Parreira, and Manfred Hauswirth. A native and adaptive approach for unified processing of linked streams and linked data. In *International Semantic Web Conference*, pages 370–388. Springer, 2011.

[114] Marcin Sydow, Mariusz Pikuła, and Ralf Schenkel. Diversum: Towards diversified summarisation of entities in knowledge graphs. In *Data Engineering Workshops (ICDEW), 2010 IEEE 26th International Conference*, pages 221–226. IEEE, 2010.

[115] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):34–43, 2001.

[116] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data: The story so far. In *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, pages 205–227. IGI global, 2011.

[117] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250, 2008.

[118] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. Dbpedia-a crystallization point for the web of data. *Journal of Web Semantics*, 7(3):154–165, 2009.

[119] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, pages 697–706, 2007.

[120] Edward Curry, James O'Donnell, Edward Corry, Souleiman Hasan, Marcus Keane, and Seán O'Riain. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics*, 27(2):206–219, 2013.

[121] Stefan Decker, Sergey Melnik, Frank Van Harmelen, Dieter Fensel, Michel Klein, Jeen Broekstra, Michael Erdmann, and Ian Horrocks. The semantic web: The roles of xml and rdf. *IEEE Internet Computing*, 4(5):63–73, 2000.

[122] Eugenio Rubio-Drosdov, Daniel Díaz-Sánchez, Florina Almenárez, Patricia Arias-Cabarcos, and Andrés Marín. Seamless human-device interaction in the internet of things. *IEEE Transactions on Consumer Electronics*, 63(4):490–498, 2017.

[123] Muhammad Ahsan Raza, Rahmah Mokhtar, Noraziah Ahmad, Maruf Pasha, and Urooj Pasha. A taxonomy and survey of semantic approaches for query expansion. *IEEE Access*, 7:17823–17833, 2019.

[124] Šejla Čebirić, François Goasdoué, Haridimos Kondylakis, Dimitris Kotzinos, Ioana Manolescu, Georgia Troullinou, and Mussab Zneika. Summarizing semantic graphs: a survey. *The International Journal on Very Large Data Bases*, 28(3):295–327, 2019.

[125] Hiteshwar Kumar Azad and Akshay Deepak. Query expansion techniques for information retrieval: a survey. *Information Processing & Management*, 56(5):1698–1735, 2019.

[126] Ahlem Rhayem, Mohamed Ben Ahmed Mhiri, and Faiez Gargouri. Semantic web technologies for the internet of things: Systematic literature review. *Internet of Things*, 11:100206, 2020.

[127] Michael Compton, Payam Barnaghi, Luis Bermudez, Raul Garcia-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, et al. The ssn ontology of the w3c semantic sensor network incubator group. *Journal of Web Semantics*, 17:25–32, 2012.

[128] Daniel Philip Puschmann. *Extracting information from heterogeneous internet of things data streams.* PhD thesis, University of Surrey, 2018.

[129] Jagdev Bhogal, Andrew MacFarlane, and Peter Smith. A review of ontology based query expansion. *Information Processing & Management*, 43(4):866–886, 2007.

[130] Melinda McDaniel and Veda C Storey. Evaluating domain ontologies: clarification, classification, and challenges. *ACM Computing Surveys (CSUR)*, 52(4):1–44, 2019.

[131] Susan Jones, Mike Gatford, Steve Robertson, Micheline Hancock-Beaulieu, Judith Secker, and Steve Walker. Interactive thesaurus navigation: intelligence rules ok? *Journal of the American Society for Information Science*, 46(1):52–59, 1995.

[132] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey. *arXiv preprint arXiv:1901.09069*, 2019.

[133] Bin Wang, Angela Wang, Fenxiao Chen, Yuncheng Wang, and C-C Jay Kuo. Evaluating word embedding models: Methods and experimental results. *APSIPA Transactions on Signal and Information Processing*, 8, 2019.

[134] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *31st AAAI Conference on Artificial Intelligence*, 2017.

[135] Jose Camacho-Collados and Mohammad Taher Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research*, 63:743–788, 2018.

[136] Arijit Khan, Sourav S Bhowmick, and Francesco Bonchi. Summarizing static and dynamic big graphs. *Proceedings of the Very Large Data Bases Endowment*, 10(12):1981–1984, 2017.

[137] Haridimos Kondylakis, Dimitris Kotzinos, and Ioana Manolescu. Rdf graph summarization: principles, techniques and applications. In *22nd International Conference on Extending Database Technology*, pages 433–436, 2019.

[138] Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. Graph summarization methods and applications: A survey. *ACM Computing Surveys (CSUR)*, 51(3):1–34, 2018.

[139] Frieder Ganz, Daniel Puschmann, Payam Barnaghi, and Francois Carrez. A practical evaluation of information processing and abstraction techniques for the internet of things. *IEEE Internet of Things Journal*, 2(4):340–354, 2015.

[140] Paris Carbone, Asterios Katsifodimos, Stephan Ewen, Volker Markl, Seif Haridi, and Kostas Tzoumas. Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 36(4), 2015.

[141] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, volume 96, pages 226–231, 1996.

[142] Douglas H Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2(2):139–172, 1987.

[143] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[144] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022, 2003.

[145] Dongjun Wei, Yaxin Liu, Fuqing Zhu, Liangjun Zang, Wei Zhou, Jizhong Han, and Songlin Hu. Esa: entity summarization with attention. In *Proceedings of EYRE 19: 2nd International Workshop on Entity Retrieval.* ACM, 2019.

[146] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 2017.

[147] Marcin Sydow, Mariusz Pikuła, and Ralf Schenkel. The notion of diversity in graphical entity summarisation on semantic knowledge graphs. *Journal of Intelligent Information Systems*, 41(2):109–149, 2013.

[148] Kalpa Gunaratna, Krishnaprasad Thirunarayan, Amit Sheth, and Gong Cheng. Gleaning types for literals in rdf triples with application to entity

summarization. In *International Semantic Web Conference*, pages 85–100. Springer, 2016.

[149] Danyun Xu, Liang Zheng, and Yuzhong Qu. Cd at ensec 2016: Generating characteristic and diverse entity summaries. In *Proceedings of the 2nd International Workshop on Summarizing and Presenting Entities and Ontologies*, 2016.

[150] Dongjun Wei, Shiyuan Gao, Yaxin Liu, Zhibing Liu, and Longtao Hang. Mpsum: entity summarization with predicate-based matching. In *Proceedings of EYRE 19: 2nd International Workshop on Entity Retrieval*, 2020.

[151] Kalpa Gunaratna, Amir Hossein Yazdavar, Krishnaprasad Thirunarayan, Amit Sheth, and Gong Cheng. Relatedness-based multi-entity summarization. In *Proceedings of the 36th International Joint Conference on Artificial Intelligence*, page 1060, 2017.

[152] David Pisinger. The quadratic knapsack problem—a survey. *Discrete Applied Mathematics*, 155(5):623–648, 2007.

[153] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in Neural Information Processing Systems*, 26, 2013.

[154] Gong Cheng, Thanh Tran, and Yuzhong Qu. Relin: relatedness and informativeness-based centrality for entity summarization. In *International Semantic Web Conference*, pages 114–129. Springer, 2011.

[155] Andreas Thalhammer and Achim Rettinger. Browsing dbpedia entities with summaries. In *European Semantic Web Conference*, pages 511–515. Springer, 2014.

[156] Andreas Thalhammer, Nelia Lasierra, and Achim Rettinger. Linksum: using link analysis to summarize entity data. In *International Conference on Web Engineering*, pages 244–261. Springer, 2016.

[157] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. Dynamic factual summaries for entity cards. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 773–782, 2017.

[158] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[159] Jörg Waitelonis and Harald Sack. Towards exploratory video search using linked data. *Multimedia Tools and Applications*, 59(2):645–672, 2012.

[160] Mohammad Sadoghi and Hans-Arno Jacobsen. Relevance matters: Capitalizing on less (top-k matching in publish/subscribe). In *2012 IEEE 28th International Conference on Data Engineering*, pages 786–797. IEEE, 2012.

[161] William Culhane, KR Jayaram, and Patrick Eugster. Fast, expressive top-k matching. In *Proceedings of the 15th International Middleware Conference*, pages 73–84, 2014.

[162] Haifeng Liu and H-Arno Jacobsen. A-topss—a publish/subscribe system supporting approximate matching. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 1107–1110. Elsevier, 2002.

[163] Milenko Petrovic, Ioana Burcea, and Hans-Arno Jacobsen. S-topss: Semantic toronto publish/subscribe system. In *Proceedings of the 29th Annual International Conference on Very Large Data Bases*, pages 1101–1104. Elsevier, 2003.

[164] Weiwei Zhang, Jiangang Ma, and Dan Ye. Fomatch: A fuzzy ontology-based semantic matching algorithm of publish/subscribe systems. In *2008 International Conference on Computational Intelligence for Modelling Control & Automation*, pages 111–117. IEEE, 2008.

[165] Jinling Wang, Beihong Jin, and Jing Li. An ontology-based publish/subscribe system. In *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*, pages 232–253. Springer, 2004.

[166] Liangzhao Zeng and Hui Lei. A semantic publish/subscribe system. In *IEEE International Conference on E-Commerce Technology for Dynamic E-Business*, pages 32–39. IEEE, 2004.

[167] Qunzhi Zhou, Yogesh Simmhan, and Viktor Prasanna. Scepter: Semantic complex event processing over end-to-end data flows. *Technical Report,*

Computer Science Department, University of Southern California, pages 12–926, 2012.

[168] Alex Wun, Milenko Petrovi, and Hans-Arno Jacobsen. A system for semantic data fusion in sensor networks. In *Proceedings of the 2007 Inaugural International Conference on Distributed Event-based Systems*, pages 75–79. ACM, 2007.

[169] Kia Teymourian, Malte Rohde, and Adrian Paschke. Fusion of background knowledge and streams of events. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, pages 302–313, 2012.

[170] Souleiman Hasan, Sean O'Riain, and Edward Curry. Towards unified and native enrichment in event processing systems. In *Proceedings of the 7th ACM International Conference on Distributed Event-based Systems*, pages 171–182, 2013.

[171] Piyush Yadav, Dibya Prakash Das, and Edward Curry. State summarization of video streams for spatiotemporal query matching in complex event processing. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 81–88. IEEE, 2019.

[172] Piyush Yadav, Dhaval Salwala, Dibya Prakash Das, and Edward Curry. Knowledge graph driven approach to represent video streams for spatiotemporal event pattern matching in complex event processing. *International Journal of Semantic Computing*, 14(03):423–455, 2020.

[173] Andrew McGregor. Graph stream algorithms: a survey. *ACM SIGMOD Record Special Interest Group on Management of Data*, 43(1):9–20, 2014.

[174] Peter Triantafillou and Andreas Economides. Subscription summarization: A new paradigm for efficient publish/subscribe systems. In *24th International Conference on Distributed Computing Systems*, pages 562–571. IEEE, 2004.

[175] Eiko Yoneki and Jean Bacon. Distributed multicast grouping for publish/subscribe over mobile ad hoc networks. In *IEEE Wireless Communications and Networking Conference, 2005*, volume 4, pages 2293–2299. IEEE, 2005.

[176] Zbigniew Jerzak and Christof Fetzer. Bloom filter based routing for content-based publish/subscribe. In *Proceedings of the 2nd International Conference on Distributed Event-based Systems*, pages 71–81, 2008.

[177] Yi-min Wang, Lili Qiu, Chad E Verbowski, Demetrios Achlioptas, Gautam Das, and Per-Ake Larson. Summary-based routing for content-based event distribution networks, April 3 2007. US Patent 7,200,675.

[178] Kemafor Anyanwu, Angela Maduko, and Amit Sheth. Semrank: ranking complex relationship search results on the semantic web. In *Proceedings of the 14th International Conference on World Wide Web*, pages 117–127, 2005.

[179] Thomas Franz, Antje Schultz, Sergej Sizov, and Steffen Staab. Triplerank: Ranking semantic web data by tensor decomposition. In *International Semantic Web Conference*, pages 213–228. Springer, 2009.

[180] Antonio J Roa-Valverde and Miguel-Angel Sicilia. A survey of approaches for ranking on the web of data. *Information Retrieval*, 17(4):295–325, 2014.

[181] Petar Ristoski and Heiko Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *International Semantic Web Conference*, pages 498–514. Springer, 2016.

[182] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[183] Hailong Jin, Lei Hou, Juanzi Li, and Tiansi Dong. Attributed and predictive entity embedding for fine-grained entity typing in knowledge bases. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 282–292, 2018.

[184] Marieke van Erp and Piek Vossen. Entity typing using distributional semantics and dbpedia. In *International Semantic Web Conference*, pages 102–118. Springer, 2016.

[185] Zoi Kaoudi, Iris Miliaraki, and Manolis Koubarakis. Rdfs reasoning and query answering on top of dhts. In *International Semantic Web Conference*, pages 499–516. Springer, 2008.

[186] Yi Zhang, Jamie Callan, and Thomas Minka. Novelty and redundancy detection in adaptive filtering. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 81–88. ACM, 2002.

[187] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 2–11, 2003.

[188] Christos Faloutsos, Mudumbai Ranganathan, and Yannis Manolopoulos. Fast subsequence matching in time-series databases. *ACM SIGMOD Record Special Interest Group on Management of Data*, 23(2):419–429, 1994.

[189] Kin-Pong Chan and Ada Wai-Chee Fu. Efficient time series matching by wavelets. In *Proceedings 15th International Conference on Data Engineering*, pages 126–133. IEEE, 1999.

[190] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.

[191] Marc Moonen and Bart De Moor. *SVD and Signal Processing, III: Algorithms, Architectures and Applications*. Elsevier, 1995.

[192] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3(3):263–286, 2001.

[193] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the Very Large Data Base Endowment*, 1(2):1542–1552, 2008.

[194] Şefki Kolozali, Daniel Puschmann, Maria Bermudez-Edo, and Payam Barnaghi. On the effect of adaptive and nonadaptive analysis of time-series sensory data. *IEEE Internet of Things Journal*, 3(6):1084–1098, 2016.

[195] Justin Borg and Joseph G Vella. Mining massive time series data: With dimensionality reduction techniques. In *International Conference on Advances in Computing and Data Sciences*, pages 496–506. Springer, 2020.

[196] James A Reggia and Yun Peng. Modeling diagnostic reasoning: a summary of parsimonious covering theory. *Computer Methods and Programs in Biomedicine*, 25(2):125–134, 1987.

[197] Lawrence Rabiner and Biinghwang Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, 3(1):4–16, 1986.

[198] Christoph Doblander, Tanuj Ghinaiya, Kaiwen Zhang, and Hans-Arno Jacobsen. Shared dictionary compression in publish/subscribe systems. In *Proceedings of the 10th ACM International Conference on Distributed and Event-based Systems*, pages 117–124, 2016.

[199] Martin Serrano and Amelie Gyrard. A review of tools for iot semantics and data streaming analytics. *Building Blocks for IoT Analytics*, 6:139–163, 2016.

[200] Mahdi Fahmideh and Didar Zowghi. An exploration of iot platform development. *Information Systems*, 87:101409, 2020.

[201] Richard A Davis, Keh-Shin Lii, and Dimitris N Politis. Remarks on some nonparametric estimates of a density function. In *Selected Works of Murray Rosenblatt*, pages 95–100. Springer, 2011.

[202] Cuong Truong, Kay Römer, and Kai Chen. Sensor similarity search in the web of things. In *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–6. IEEE, 2012.

[203] Saeed V Vaseghi. *Advanced digital signal processing and noise reduction.* John Wiley & Sons, 2008.

[204] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Conference on Empirical Methods in Natural Language Processing*, 2015.

[205] Mohammad Taher Pilehvar and Nigel Collier. De-conflated semantic representations. In *Conference on Empirical Methods in Natural Language Processing*, 2016.

[206] Andrei Kutuzov, Murhaf Fares, Stephan Oepen, and Erik Velldal. Word vectors, reuse, and replicability: Towards a community repository of large-text resources. In *Proceedings of the 58th Conference on Simulation and Modelling*, pages 271–276. Linköping University Electronic Press, 2017.

[207] Charu C Aggarwal. A survey of stream clustering algorithms. In *Data Clustering*, pages 231–258. Chapman and Hall/CRC, 2018.

[208] Margaret H Dunham. *Data mining: Introductory and advanced topics.* Pearson Education India, 2006.

[209] Stephen Jose Hanson and Malcolm Bauer. Conceptual clustering, categorization, and polymorphy. *Machine Learning*, 3(4):343–372, 1989.

[210] Airel Pérez-Suárez, José F Martínez-Trinidad, and Jesús A Carrasco-Ochoa. A review of conceptual clustering algorithms. *Artificial Intelligence Review*, 52(2):1267–1296, 2019.

[211] Charu C Aggarwal, S Yu Philip, Jiawei Han, and Jianyong Wang. A framework for clustering evolving data streams. In *Proceedings 2003 VLDB 29th International Conference on Very Large Data Bases*, pages 81–92. Morgan Kaufmann, 2003.

[212] Feng Cao, Martin Estert, Weining Qian, and Aoying Zhou. Density-based clustering over an evolving data stream with noise. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 328–339. SIAM, 2006.

[213] Philipp Kranen, Ira Assent, Corinna Baldauf, and Thomas Seidl. The clustree: indexing micro-clusters for anytime stream mining. *Knowledge and Information Systems*, 29(2):249–272, 2011.

[214] Michael Hahsler and Matthew Bolaños. Clustering data streams based on shared density between micro-clusters. *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1449–1461, 2016.

[215] Matthias Carnein, Dennis Assenmacher, and Heike Trautmann. An empirical comparison of stream clustering algorithms. In *Proceedings of the Computing Frontiers Conference*, pages 361–366, 2017.

[216] James W Tanaka and Marjorie Taylor. Object categories and expertise: Is the basic level in the eye of the beholder? *Cognitive Psychology*, 23(3):457–482, 1991.