Ollscoil na Gaillimhe
University of Galway

Doctoral Thesis

Unsupervised Deep Representation Learning for Low-Resourced Languages and Applications

Koustava Goswami
B.Tech., M.Sc. in Computer Science

External Examiner
Dr. Marcos Zampieri

Supervisor
Dr. John P McCrae
Dr. Theodorus Fransen

Internal Examiner
Dr. Peter Paul Buitelaar

Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of ***Doctor of Philosophy***

Data Science Institute
School of Computer Science
College of Science and Engineering
University of Galway

DECLARATION

I declare that this thesis, titled "*Unsupervised Deep Representation Learning for Low-Resourced Languages and Applications*", is composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification.

Galway,

Koustava Goswami

“To those who love representation learning, to you who are reading this...”

ACKNOWLEDGEMENTS

This thesis marks the end of a beautiful three years PhD journey at Galway, a very memorable and exciting phase of my life. This journey has not only allowed me to achieve one of the highest degrees in my life but also helped me to gain wonderful research experiences in both academic and industrial settings. This would not have been possible without the guidance and support of many people.

First, I would like to thank my wonderful PhD supervisors Dr John McCrae and Dr Theodorus Fransen for all their patience and sensible advice. They have encouraged me, provided me with valuable and constructive feedback and supported me in every aspect. John, thank you for selecting me and giving me the freedom to work in the domains I am interested in. Without your prompt help and assistance every time, this journey would have been very difficult for me. Theodorus, thank you for helping me out with all the writing every time. Your guidance in different linguistic contexts helped me to put my ideas into words.

I am grateful to my parents Dr Madan Gopal Goswami and Mrs Swati Goswami, my brother Aritra Goswami, my in-laws Mr Arup Kumar Banerjee and Mrs Jayati Banerjee, my brother-in-law Ritwik Banerjee and my whole family for all their supports and encouragements. Without them, this long journey would not have been possible.

A special thanks to my lovely wife Maitri for being my best friend, and my companion over the years. Her encouragement and patience during the hectic days and nights of my work defy exaggeration. It is impossible to imagine this journey without her unending support and love.

During this journey, I was fortunate enough to work with some wonderful mentors who have helped me to grow as a researcher. I am grateful to Dr Sourav Dutta from Huawei Research Ireland and Dr Haytham Assem from Amazon Alexa for providing me with the opportunity to work in industry research settings in the early stage of my PhD. Sourav, thank you for all the help and guidance you provided from time to time. I would also like to thank Dr Heike Adel and Dr Jun Araki from Bosch Research for allowing me to work on some exciting and new research topics under their guidance.

I would also like to thank my PhD graduate research committee members Dr Paul Buitelaar, Prof. Michael G Madden and Dr Mihael Arcan for all their valuable feedback, which has helped me to polish my research work.

I am also fortunate enough to work with some of the great minds in my institute. Specifically, I want to thank two of my friends from my lab Rajdeep and Priya for having such wonderful discussions on different research works which fortunately helped me to publish our joint works. I would also like to thank my friends Shardul, Bharathi, Atul, Omnia, Piyush, Tarek, Adrian, Bernardo, and Oksana here at our institute for helping me out in different scenarios.

Lastly, I would like to acknowledge financial support from two funding bodies that enabled me to undertake my PhD research; the Irish Research Council through grant IRCLA/2017/129 (CARDAMOM-Comparative Deep Models of Language for Minority and Historical Languages) and Science Foundation Ireland through grant SFI/12/RC/2289_P2 (Insight_2).

ABSTRACT

Artificial intelligence and Natural Language Processing (NLP) are becoming integral parts of modern technologies and amenities starting from the adaptation of Amazon Alexa to automatic chatbots in different industries. Though earlier NLP algorithms were developed mainly for tasks in the English language, researchers have taken a step to adapt these to different languages around the globe. Despite many advancements in the NLP domain, these algorithms have quite poor performance in low-resource scenarios. The state-of-the-art multilingual language models suffer due to a lack of high-quality annotated training datasets for low-resource languages and applications. Therefore there is a need of designing efficient automatic NLP systems for low-resource languages and applications to achieve state-of-the-art performances on the downstream NLP tasks.

In this thesis, we introduce novel state-of-the-art deep models which capture global and contextual semantic representations of sentences in a document. We focus on building unsupervised deep models to efficiently exploit the existing unlabelled datasets for feature extraction. Our contribution also includes designing state-of-the-art unsupervised sentence embedding models capable of performing a wide range of cross-lingual tasks for low-resource scenarios. We raise several research questions at the start of the thesis and we provide answers supported by state-of-the-art experimental results.

While training deep representation models for low-resource languages, one of the key challenges was to recognize the closely-related languages in a code-mixed corpus having short texts in the NLP pipeline. Existing language identification models support a limited set of languages thus we have come up with a novel idea of building an unsupervised language identification system by capturing the global sentence representations of the languages. Our experimental results highlight the efficiency of the framework across different language families including a challenging task, dialect identification. We then turn our interest to explore the capacity of building a cross-lingual contextual sentence embedding framework to be trained on non-parallel sentences. To this extent, we designed an unsupervised sentence embedding framework trained using a dual-encoder architecture in a multitask setup. During training, we also inject the word-level semantic knowledge to preserve the relative semantic distance between input sentences. We propose to train this model using a new machine-learning framework called the anchor-learner framework. During evaluations, on a few benchmark experiments, the unsupervised model has outperformed some of the state-of-the-art supervised sentence embedding models. While we can capture better cross-lingual contextual sentence representations, there is still a big performance gap for the natural language understanding tasks in comparison with the large language models. By our novel, unsupervised projection methodology of embedding space, the sentence embedding models learn the semantic understanding of a big language model with fewer structural parameters. This produces rich semantic sentence embedding, which in a low-resource domain and applications, produces better results for the classification of NLP tasks. Finally, we worked on improving the sentence embeddings for closely-related languages by injecting the knowledge of cognates. One of the hard tasks is to identify cognates. For low-resource languages it is very hard to get annotated cognate pairs, thus we proposed a novel unsupervised cognate detection framework which exploits the grammatical and structural knowledge of the words of a language to be transferred to closely-related languages of the same language family. The introduction of these cognates to sentence embedding frameworks during training enhanced the efficacy of the models in cross-lingual tasks.

Our contribution also includes the introduction of several unsupervised loss functions that can be applied to different NLP domains and tasks. The methods and frameworks that are developed in this thesis can be implemented to a wide range of low-resource languages and applications with minimum training datasets required.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Natural Language Processing (NLP) has transformed the field of Artificial Intelligence (AI) with the ability to understand and communicate in human languages. Language understanding is a very complicated task as there are more than 7000 languages (Abney & Bird, 2010; Hauksdóttir, 2014) in the world and each language has its own semantic and syntactic structures. With the introduction of deep learning using high quality annotated training datasets immense developments happened in the field of NLP starting from enhancing the efficiency of Part-Of-Speech (POS) taggers (Kanakaraddi & Nandyal, 2018) to building well-established chatbot systems (Caldarini et al., 2022). Intelligent NLP systems are also playing key roles while it comes down to the development of under-resourced languages in the modern technology driven world (Cunliffe et al., 2022).

Designing and developing high-quality NLP systems requires lots of high-quality annotated training datasets or corpora along with high-end computing resources. To design the most accurate automatic NLP systems for different complex languages, linguistic expertise is needed (Meurers, 2021). For example, while building automatic machine translation systems for morphologically rich languages, grammatical knowledge of the source and target languages has become a must skill to have (Bisk & Tran, 2018). State-of-the-art large language models are often seen to produce enhanced performances on different NLP tasks. Reimers and Gurevych, 2020 proposed a state-of-the-art supervised sentence embedding model capable of identifying efficient parallel sentences between two corpora. State-of-the-art multilingual language model *XLM-R* (Conneau et al., 2020) has high efficacy for different downstream token classification tasks including named-entity recognition. These systems need very high-quality massive corpora to be trained. This highlights the fact that NLP systems are dependent on good quality and amount of training datasets.

Accumulation of high quality training resources to train NLP systems is very expensive and time consuming. Annotation requires releasing refined annotator guidelines, choosing high knowledgeable annotators of the specific fields, and managing them to produce and test resources within specific times (Tauchmann, 2021). Recently there are tools to do crowd sourcing annotations like Amazon Mechanical Turk (MTurk) (Crowston, 2012), but these are quite expensive to be used for a small scale organisations. On the other hand, since it is very expensive and challenging to get high quality annotated texts to train a NLP systems, the state-of-the-art high performing NLP systems are very much restricted to "high-resource" languages (e.g., English, Spanish, Chinese etc.) or scenarios (e.g., movie domain, internal product review domain, classification tasks on big training datasets including news articles etc.). As a result, designing NLP systems for "under-resource languages" or "low-resource applications" is quite challenging.

Most of the state-of-the-art NLP systems for downstream NLP applications are supervised models which means they rely heavily on high-quality annotated training data or massive training corpus. Thus these systems are not quite efficient when it comes down to "low-resource" NLP systems. Inspite of the existence of 7000 languages in the world, only a small fraction of these languages (roughly 20) are considered as "high-resource" languages (Baumann & Pierrehumbert, 2014). Recently researchers have tried to extend the usability of NLP systems to be adopted widely by the "low-resource" community (Hedderich et al., 2021). For example Cardenas et al., 2019 proposed an unsupervised parts-of-speech tagging system which

achieved high performance for different low-resource languages. Kakwani et al., 2020 designed *IndicNLP-Suite* trained on *ALBERT* language model to achieve comparable results for different downstream NLP tasks for Indian languages. Despite the fact that many languages like Bengali, Assamese, Bhojpuri, Vietnamese are widely spoken by millions of people on the web, digitization of these languages is quite low which demands the need of better adaptability of NLP systems around the globe. Knowledge based systems such as knowledge graphs (either domain specific or language specific) (Abu-Salih, 2021; Sharma et al., 2019), wordnets (Bosch & Griesel, 2018; Kanojia et al., 2018; Rahit et al., 2018), dictionaries (Mati et al., 2021) are becoming very rich resources for building low-resource NLP applications. It is established that getting information or lexical content about a language is comparatively easy using dictionaries and thus it can be used as a resource to train NLP models (Duan et al., 2020; Mikolov et al., 2013b). Many researchers are trying to build rich resources by creating dictionaries or wordnets for these low-resource languages (Bosch & Griesel, 2018; Kanojia et al., 2018). But creating these link-based resources can be very tedious work and most often it needs domain or linguistic expertise. One the other hand, one of the prime examples of open source resource creation is Wiktionary[1]. It is a widely used open source resource. Recently researchers used the Wiktionary resources to train models for different NLP applications (Segonne et al., 2019). These resources of Wiktionary help to construct unlabelled corpora. For most of the low-resourced languages or domains, unlabelled corpora can be collected using some web-based scraping tools. Many such resources can be found in social media posts or in reviews posted by customers (Chakravarthi et al., 2020b). Around the globe, official documents of countries can also be found documented in their own languages which are great multilingual resources to train distributional models (Koehn, 2005). The sentence structures of these documents convey lots of structural and semantic information. Thus injecting the sentence representations of the document can be helpful while designing NLP systems for the languages or resources where annotated data is not found.

1.1 MOTIVATION AND AIM

1.1.1 Motivation behind Deep Representation Learning

Capturing an efficient document representation includes understanding the underlying semantic information about the language or the domain. Recently document representation has been explored for different NLP applications starting from information retrieval (H. Tang et al., 2021) to designing question-answering (Zayats et al., 2021) systems. Traditionally documents are represented using *one-hot* word vectors present in the vocabulary set (Xu & Rudnicky, 2000). But this methodology has major disadvantages. It does not consider the word order in a sentence or paragraph to be encoded and suffers from data sparsity and high dimensionality. Moreover, the semantic relatedness between the words or sentences is not preserved. Later, deep model based distributed representation learning learns (refer to Section 2.3.2) the word level semantics, thus encoding the documents becomes more meaningfully. Deep learning systems overcome the issue of curse of dimensionality by mapping the objects to low-dimensional vectors as dense representations of the inputs. The learning of these systems is *hierarchical* in nature (Z. Liu et al., 2020) and the non-linear mapping through the deep layers captures the structural and semantic representation of the inputs. Mostly, modern deep representation based systems are designed to capture the context of the word based on the surroundings as vector based systems (Turney & Pantel, 2010). These are known as *embeddings*. Popular neural word embedding techniques (Mikolov et al., 2013b; Pennington et al., 2014) preserve the

1 https://hi.wiktionary.org/wiki/Hindi – The Hindi Wiktionary

semantic relation and information about entities (Goldberg, 2017). These systems have been shown to capture the word analogy on different hierarchy providing word level semantics for different NLP applications (Mikolov et al., 2013c). Later, transformer (Vaswani et al., 2017a) based large language models like *BERT* (Devlin et al., 2019) were developed that learn the positional and contextual representation of the words and capture the syntactic regularities as well as semantic relatedness. Researchers have tried to encode paragraphs of a document via deep neural network systems (D. Wang et al., 2020; Q. Xie et al., 2021), but the word-level encoding process of these paragraphs are less efficient for complex NLP applications. In fact, Prasad et al., 2019 studied the ability of these language models to preserve the syntactic regularities on the sentence level rather than word level. Reimers and Gurevych, 2020 also highlighted that the sentence representations of these language models are more efficient for mapping the cross-lingual documents in the same vector space. But, this system does not perform well for low-resourced languages. Conneau et al., 2020 highlighted the absence of efficient training datasets for most of the low-resource languages and thus an unsupervised or semi-supervised training methodology has to be adopted. Thus the open question is what are the best possible ways to design unsupervised frameworks? In this thesis I dive deeper into the design of unsupervised deep neural systems, which can efficiently capture the sentence representations of the documents in low-resourced languages.

1.1.2 Aim of the Thesis

I aim to build novel deep learning frameworks for low-resource languages or other low-resource NLP applications. As we discussed, recently NLP has progressed quite rapidly and quite efficient systems are designed, though these systems are mainly designed for high-resourced languages like English, Spanish, Chinese etc. For these languages there exist efficient processing tools for the downstream tasks like part-of-speech tagging, machine translation, named-entity recognition, word-segmentation, natural language generation, word sense disambiguation, and natural language understanding tasks. These tools are being developed based on better adaptability of syntactic or semantic features of the individual languages. Most of these tools are designed based on annotated data or with the help of linguists. When it comes to low-resourced languages for different language families, it is not practical to hire linguists for every task. On the other hand, even for high-resource languages, short text classification across diverse domains and applications in low-resourced scenarios still have quite poor performance using base language models. Large language models like *XLM-R-large* (Conneau et al., 2020) currently provide state-of-the-art results on several benchmark tasks, but involve an architecture with a staggering 550 million parameters. Adapting such large multi-lingual language models to classification tasks requires appropriate fine-tuning. The requirement of sufficient annotated training data to properly tune such models further complicates their deployment in the production pipeline. Thus, our decision of designing efficient multilingual unsupervised deep neural systems can be the solution to tackle these problems by exploiting the existing unlabeled data and produce better downstream tasks.

In this thesis, we aim to work on two main tasks for cross-lingual representation learning

- **Unsupervised Cross-lingual Sentence Representation Learning:** For most of the downstream cross-lingual tasks like parallel sentence matching tasks, understanding the contextual and semantic representations of the cross-lingual sentences is important. Existing cross-lingual sentence embeddings (Hirota et al., 2020; Reimers & Gurevych, 2020) are trained to map contextual embeddings in the shared vector space based on parallel sentences, which is a hard task for low-resource languages. Thus to tackle this problem, we aim to design a novel unsupervised sentence embedding framework

trained with no parallel training dataset. Our goal is to capture the semantic relatedness between the source and target sentence pairs by utilizing the semantic knowledge of the contextual words as a knowledge injector. Being fully unsupervised the applicability of the sentence embedding framework increases for diverse low-resource languages of different language families. Though the cross-lingual sentence embedding frameworks achieve state-of-the-result in different cross-lingual tasks, there is a big difference in performance between large language models and the sentence embedding methods for natural language understanding tasks. Thus, state-of-the-art methodologies include fine-tuning the large language models on downstream tasks. But for low-resource NLP applications, the large language models are hard to be fine-tuned due to a lack of annotated datasets and parameter tuning. We claim that by injecting the knowledge of the contextual word embeddings of the large language models into the sentence embedding model in an unsupervised way the performance of the sentence embedding model can be improved. When it comes to learning contextual unsupervised cross-lingual sentence embeddings, collecting and building a cross-lingual dataset for low-resourced languages is itself a challenging task. Conneau et al., 2020 highlighted the challenges while collecting datasets for most of the low-resource languages and the need of identifying these languages using a language identification tool. Chakravarthi et al., 2020b designed a corpus for Dravidian languages scraping user comments from social media site *YouTube*. An open sourced *twitter-scraper* like *APIFY*[2] can scrape the tweets and build a multilingual corpus. But these corpora are inter-sententially code-mixed and identifying languages is the primary step before training embedding models. Existing language identification tools like *FastText Language Identifier*[3] cover limited languages. The task becomes harder for closely-related languages due to orthographic and syntactic similarities. We argue that understanding the global representation of sentences provides a better understanding of these languages. Thus, we aim to learn the sentence representations of a language in an unsupervised way and cluster them.

- **Enhancing Sentence Representations for Closely-related Languages:** In this task, we aim to improve the efficiency of sentence embedding frameworks for low-resourced closely related languages. We aim to exploit the similarities between words of the low-resourced closely-related languages. Words with similar meanings and spellings (cognates) often convey the similar contextual information of cross-lingual sentences (Fourrier & Montariol, 2022). Thus we claim that cognate pairs can help to improve the semantic understanding of the sentences between two closely-related languages. But identifying cognates itself is a very hard task for low-resource languages due to the lack of annotated cognate datasets. Thus, we investigate the possibility of building an unsupervised cognate detection system and whether these cognates enhance the performance of the cross-lingual sentence embedding frameworks. To this end, we use the grammatical and structural knowledge of the words to capture the similarity between closely-related languages.

1.2 RESEARCH QUESTIONS

Designing efficient NLP systems for low-resource languages or scenarios is an active area of research in the community. As we discussed, we aim for unsupervised deep sentence representation learning to overcome the problem of lack of annotated training datasets. However, the performance of the state-of-the-art unsupervised methods drops with respect to the supervised models for different downstream NLP tasks.

2 https://apify.com/vdrmota/twitter-scraper#what-data-can-twitter-scraper-extract
3 https://fasttext.cc/docs/en/language-identification.html

Early research on unsupervised representation learning of sentence or word level constructions involved much user interventions for feature selection which confined the implementation of these methods in wider scenarios. In this thesis, I tried to investigate how unsupervised representation learning can be designed efficiently using novel deep learning algorithms. These algorithms will capture language features of different language families to produce better results for downstream tasks. The research questions are framed to solve the above tasks by first identifying languages of a document followed by understanding the structures and contexts of sentences in the document to capture global and local semantic and syntactic representations.

The research questions are:

- **RQ1.** Is an unsupervised deep sentence embedding framework capable of identifying languages as accurately as supervised language identification models for code-mixed under-resourced and closely-related languages?

- **RQ2.** Does an unsupervised deep sentence embedding framework generate efficient sentence embeddings in cross-lingual domains for under-resourced languages without the use of parallel corpora for downstream natural language tasks? Does unsupervised projection of efficient word representations improve sentence embeddings in cross-lingual domains?

- **RQ3.** Does effective cross-lingual word representations learning improve unsupervised cognate detection without any language information for closely related and low-resourced languages? Having detected cognates, does this assist in producing better sentence embeddings for NLP applications?

These research questions are arranged to address different research challenges while learning the semantic representation of the cross-lingual sentences efficiently. Generally speaking, the first research question is concerned with identifying the languages or dialects in an inter-sentential code-mixed corpus capturing global representations of the sentences. The answer to the second research question produces a novel unsupervised sentence embedding framework which captures the contextual semantic representation of the cross-lingual sentences without the need for parallel corpora. Thus it opens up the applicability of this framework to a wide range of language families including different under-resourced languages. Moreover, through this research question, we find a novel solution for capturing sentence representations for low-resourced natural language processing applications. While we work on capturing the representation of the sentences, we argue that for closely-related languages better understanding of similar words can improve the performance of the cross-lingual sentence embedding models. So, we structure the third research question aiming to find an answer to how to efficiently find words with similar meaning of two closely-related languages and whether these word representations improve the cross-lingual sentence embedding performance for the downstream tasks.

1.3 CONTRIBUTIONS OF THE THESIS

During my research, I have proposed novel deep learning algorithms for each task which cover the proposed research questions. The algorithms mainly exploit the domain of unsupervised machine learning algorithms. The research contributions span from designing new encoding methods to new loss function for backpropagation during unsupervised training. The summary of contributions for the research questions are documented below.

1.3.1 Unsupervised Language and Dialect Identification

RQ1. Is an unsupervised deep sentence embedding framework capable of identifying languages as accurately as supervised language identification models for code-mixed under-resourced and closely-related languages?

Automatic Language Identification (LI) or Dialect Identification (DI) of short texts of closely related languages or dialects, is one of the primary steps in many natural language processing pipelines and is part of tasks such as language modelling, categorization and analysis of these code-mixed datasets. Language identification is considered a solved task in many cases; however, in the case of very closely related languages, or in an unsupervised scenario (where the languages are not known in advance), performance is still poor. Traditionally the LI methods are explored in monolingual documents (Hughes et al., 2006a). In supervised monolingual settings, there are some efficient document based LI methods available (McNamee, 2005). But to train these models a large amount of annotated datasets is required and these methods are still very much confined to the large documents sets and high-resource languages. Later researchers started to explore the multilingual document sets where language and dialect identification is a quite hard task to be solved. Martins and Silva, 2005 have worked on the language identification of web page contents. *Linguini* (Prager, 2000) is one of the famous LI methods for multilingual documents. In relation to this, we have discussed attempts researchers made for the LI and DI for code-mixed documents which are quite inefficient for low-resource languages. In chapter 4, we propose the Unsupervised Deep Language and Dialect Identification (UDLDI) method, which can simultaneously learn sentence embeddings and cluster assignments from short texts. The UDLDI model understands the sentence constructions of languages by applying attention to character relations which helps to optimize the clustering of languages. We have performed our experiments on three short-text datasets for different language families, each consisting of closely related languages or dialects, with very minimal training sets. Our experimental evaluations on these datasets have shown significant improvement over state-of-the-art unsupervised methods and our model has outperformed state-of-the-art LI and DI systems in supervised settings.

Publications: Research Question 1

- Goswami, Koustava, et al. "Unsupervised deep language and dialect identification for short texts." Proceedings of the 28th International Conference on Computational Linguistics. 2020.

1.3.2 Sentence Representation Learning

RQ2. Does an unsupervised deep sentence embedding framework generate efficient sentence embeddings in cross-lingual domains without the use of parallel corpora for downstream natural language tasks? Does unsupervised projection of efficient word representations improve sentence embeddings in cross-lingual domains?

Multilingual sentence embeddings capture rich semantic information not only for measuring similarity between texts but also for catering to a broad range of downstream cross-lingual NLP tasks. State-of-the-art multilingual sentence embedding models require large parallel corpora to learn efficiently, which confines the scope of these models. *Paragraph vectors* were first proposed as sentence embeddings for computing document similarity (Le & Mikolov, 2014). Capturing document representation based on paragraph encoding often misses semantic understanding between sentences. Later researchers have proposed multilingual sentence embeddings (Schwenk, 2018; Schwenk & Douze, 2017) to achieve better neural machine translation in bilingual settings. Recently, multilingual sentence embedding techniques based on parallel sentences are designed to solve broader NLP tasks (Reimers & Gurevych, 2019). But these systems are

supervised sentence embedding techniques and are ineffective when it comes to low-resourced languages. While we have identified the sentences for low-resourced languages on the collected dataset using **RQ1**, the sentences can be used to train an efficient sentence embedding model. In Chapter 5 we will introduce a new unsupervised sentence embedding technique which is very efficient at capturing sentence representation for low-resource languages. We propose a novel *sentence embedding framework* based on an *unsupervised loss function* for generating effective multilingual sentence embeddings, eliminating the need for parallel corpora. We capture semantic similarity and relatedness between sentences using a *multi-task loss function* for training a *dual encoder model* mapping different languages onto the same vector space. We demonstrate the efficacy of an unsupervised as well as a weakly supervised variant of our framework on different downstream NLP benchmark datasets including parallel sentence matching and bitext mining. The proposed unsupervised sentence embedding framework outperforms even supervised state-of-the-art methods for certain under-resourced languages on the parallel sentence matching benchmark dataset. We also showcase the capability of capturing sentence representation for monolingual sentences for better semantic understanding. Further, we show enhanced *zero-shot learning* capabilities for more than 50 languages. This model can be extended to a wide range of languages from any language family, as it overcomes the requirement of parallel corpora for training.

While we work on sentence representation learning for low-resource languages another aspect of this thesis is to build better an algorithm to capture sentence representations for low-resource NLP tasks. With the advancements of Natural Language Understanding (NLU), diverse industrial applications like user intent classification, smart chatbots, sentiment analysis and question answering have become a primary paradigm. Transformers-based multi-lingual language models such as XLM-large (Conneau et al., 2020) have performed significantly well in diverse semantic understanding and classification tasks. However, fine-tuning such large pre-trained architectures is resource and compute intensive, limiting its wide adoption in enterprise environments. We present a novel efficient and light-weight framework based on sentence embeddings to obtain enhanced multi-lingual text representations for domain-specific NLU applications. Our framework combines the concepts of *up-projection*, *alignment* and *meta-embeddings* enhancing the textual semantic similarity knowledge of smaller sentence embedding architectures. Extensive experiments on diverse cross-lingual classification tasks showcase the proposed framework to be comparable to state-of-the-art large language models (in mono-lingual and zero-shot settings), even with fewer training and resource requirements.

Publications: Research Question 2

- Goswami, Koustava, et al. "ULD@ NUIG at SemEval-2020 Task 9: Generative Morphemes with an Attention Model for Sentiment Analysis in Code-Mixed Text." Proceedings of the Fourteenth Workshop on Semantic Evaluation. 2020.
- Goswami, Koustava, et al. "Cross-lingual sentence embedding using multi-task learning." Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. 2021.
- Goswami, Koustava, et al. "Mufin: Enriching Semantic Understanding of Sentence Embedding using Dual Tune Framework." 2021 IEEE International Conference on Big Data (Big Data). IEEE, 2021.

1.3.3 Identifying Cognates using Cross-lingual Word Representation Learning

RQ3. Does effective cross-lingual word representations learning improve unsupervised cognate detection without any language information for closely related and low-resourced languages? Having detected cognates, does this assist in producing better sentence embeddings for NLP applications?

Each language has their own morphological rules to construct a word. Combination of these meaningful words with the help of different grammatical rules, form a sentence which eventually forms a document. Humans understand these word representations to gain knowledge of a language. Therefore better word representation learning for language models can lead to a better sentence representation for the downstream NLP tasks. Multilingual or cross-lingual sentence encoders are being trained on big datasets consisting of multiple languages. Often these sentences have similar meaningful words called cognates (Crystal, 2011). Moreover, it is said that most of the bilingual corpora will have cognates (Kondrak et al., 2003). So, language-agnostic cognate detection is becoming very important for sentence encoder based cross-lingual text understanding discussed in **RQ2**. It helps to improve unsupervised machine translation, named entity recognition and information retrieval. Previous approaches mainly focused on supervised cognate detection tasks based on orthographic, phonetic or state-of-the-art contextual language models, which underperformed for most under-resourced languages. In Chapter 6, we propose a novel language-agnostic unsupervised deep cognate detection framework for under-resourced languages using morphological knowledge from closely related languages. We train an encoder to gain morphological knowledge of a language and transfer the knowledge to perform unsupervised cognate detection tasks with and without the pivot language for the closely-related languages. Being fully unsupervised, it overcomes the need for hand-crafted annotation of "cognates". We performed our experiments on different published cognate detection datasets across language families and observed significant improvement over the state-of-the-art supervised and unsupervised methods.

Publications: Research Question 3

- Goswami, Koustava, et al. "Unsupervised Deep Cognate Detection Framework for Low-Resourced Languages Using Morphological Knowledge of Closely-Related Languages." Submitted and under review.

1.4 OUTLINE

In this thesis, we aim to capture efficient sentence representations of a document in an unsupervised way for low-resource languages and applications. We started by capturing the global sentence structural representations to identify languages in a code-mixed corpus. We then aim to capture the efficient contextual sentence representations in an unsupervised way which is followed by improving these representations for closely-related languages by learning word representations of similar words in a cross-lingual document. This comprises seven chapters including the introduction chapter where we introduce the motivation behind our work in this thesis.

In Chapter 2 we study the background behind low-resource languages and applications as well as the basic introduction of the generic machine learning and deep learning methodologies. We also provide a brief study of existing works on representation learning from document level to word level. In the end, we highlight the key transfer learning concepts used in different novel deep learning algorithms designed in this thesis.

In Chapter 3, we study the existing works related to the tasks we worked on in this thesis. We started with a detailed study on language identification and dialect identification tasks using different neural and classic algorithms. Then we studied the existing supervised and unsupervised sentence embedding frameworks. We have given an overview of how these models are trained and how they are semantically different from

other models. At the end of the chapter, we study how existing works study the word-pair relations to identify cognates which can potentially be used in the different downstream tasks.

In Chapter 4, we introduce a new unsupervised language and dialect identification model capable of clustering sentences according to languages. This framework is designed based on a novel unsupervised sentence embedding framework which is capable of capturing the global context of the sentences. Our introduction of a new loss function named *Maximum Likelihood Clustering Loss (MLC Loss)* helps during unsupervised representation learning without any prior linguistic information. The identified sentences for different low-resource languages can be used as a training dataset for the novel unsupervised sentence embedding framework introduced in Chapter 5.

In Chapter 5, we introduce a novel unsupervised sentence embedding framework capable of getting trained without any parallel sentences. The newly designed *Anchor-learner* framework injects word-level semantic information while mapping semantically similar sentences in the shared multilingual vector space. We introduced a novel unsupervised loss function which minimises the distance between the semantically similar sentences between two languages having semantic information of the words using word movers' distance. We further investigate whether an unsupervised projection of these sentence embeddings to the larger word vector space can improve the performance of the multilingual natural language understanding tasks. We justify our claim by producing state-of-the-art results with our newly introduced frameworks on different downstream tasks. But still, for closely-related low-resource languages, these sentence embeddings perform less efficiently due to limited knowledge about the similar words in cross-lingual sentences, which we address in Chapter 6.

In Chapter 6, we investigate how to identify and learn representations of similar meaningful words for the closely-related low-resource languages. We claim that these word representations can boost the sentence embedding performance for different cross-lingual tasks. To design the system, we capture morphology information of a pivot language and transfer the knowledge to perform unsupervised cognate detection for closely-related languages of the same language family using shared encoders. We use the MLC loss introduced in Chapter 4 to cluster the cognates. Our experimental results highlight the efficacy of this unsupervised framework on different language families. Moreover, this helps to improve the efficiency of the sentence embedding frameworks on different cross-lingual tasks. At the time of thesis submission, the work is submitted and under review in ACL Rolling Review system.

Finally, in Chapter 7, we summarize the conclusions of the works and highlight how our contributions can benefit the low-resource NLP tasks. We also mention some future works related to this thesis.

1.5 PUBLICATIONS

Other publications I published during my PhD tenure:

- Goswami, Koustava, et al. "SwitchPrompt: Learning Domain-Specific Gated Soft Prompts for Classification in Low-Resource Domains" Submitted and under review.

- Rani, Priya, Shardul Suryawanshi, Koustava Goswami, Bharathi Raja Chakravarthi, Theodorus Fransen, and John Philip McCrae. "A comparative study of different state-of-the-art hate speech detection methods in Hindi-English code-mixed data." In Proceedings of the second workshop on trolling, aggression and cyberbullying, co-located with 12th Edition of Language Resources and Evaluation Conference, pp. 42-48. 2020.

- Sarkar, Rajdeep, Koustava Goswami, Mihael Arcan, and John Philip McCrae. "Suggest me a movie for tonight: Leveraging Knowledge Graphs for Conversational Recommendation." In Proceedings of the 28th International Conference on Computational Linguistics, pp. 4179-4189. 2020.
- Ojha, Atul Kr, Priya Rani, Koustava Goswami, Bharathi Raja Chakravarthi, and John P. McCrae. "ULD-NUIG at Social Media Mining for Health Applications (SMM4H) Shared Task 2021." NAACL-HLT 2021 2021 (2021): 149-152.
- McCrae, John P., Theodorus Fransen, Sina Ahmadi, Paul Buitelaar, and Koustava Goswami. "Towards an Integrative Approach for Making Sense Distinctions." Frontiers in Artificial Intelligence (2022): p.3.

2 BACKGROUND

In this chapter, we will discuss the background information of relevance to the rest of the thesis. The discussion will include low-resourced languages and applications as well as some details on representation learning for NLP applications using machine learning and deep learning algorithms.

2.1 LOW-RESOURCE NLP

Recently, algorithm developments for low-resource natural language processing have drawn the attention of the NLP community. In the current era, widely used applications and systems are preferred to be incorporated with artificial intelligence-driven bots. Most of these applications require efficient knowledge of natural language understanding to interact with humans. For example, one of the widely used natural language applications is a conversational AI system where an agent interacts with humans. State-of-the-art Amazon Alexa, Google assistant, and Apple Siri are becoming very popular because of their increasing range of capabilities (Ram et al., 2018). These systems are capable of understanding human instructions in multiple domains and activities starting from playing music to navigating to a destination address. Over time, these AI agents are being incorporated into multiple devices which are being used in daily households globally. But due to limited language understanding of these systems, it is still a quite hard task to reach different corners of the globe. In fact, in many languages and domains, these systems are very hard to develop due to a lack of training data.

Abney and Bird, 2010 have highlighted the existence of more than 7000 languages around the globe. What should be the parameters to describe a language as low-resource is a hard research question itself. The availability of resources for these languages in different forms can be a parameter to define whether a language is low-resourced or high-resourced. Most of the NLP applications are driven by popular languages like English or Chinese. Interestingly some languages do not have resources to be considered at all. In fact, most of these languages do not have a standard writing system or are not encoded in Unicode (Maxwell & Hughes, 2006). The US government project LORELEI[1] has defined those languages as low-resource languages which do not have automated human language technology (Duong, 2017). On this note, languages which do not have adequate support to build an essential natural language processing task can be called low-resource languages. This includes a syntactically annotated corpus which often is at the heart of building statistical or classical NLP applications. If we take an example of a popular syntactic annotated corpus like the universal dependency treebank[2], it only supports over 100 languages. Thus languages outside this dataset might be considered low-resourced languages. But this hypothesis might be argued as in recent times language specific WordNets are introduced. WordNet is a large lexical database of languages grouped into synsets. Having said that the WordNet development for many different language families is still in a very elementary stage, they thus cannot contribute to building datasets for every low-resource language.

1 https://www.darpa.mil/program/low-resource-languages-for-emergent-incidents
2 https://universaldependencies.org/

Alegria et al., 2011 describe six levels of language typology to develop human language technologies. These levels are being defined based on the availability of data sources over the internet. The first level describes the most data-rich languages while the second level consists of the top 10 most used languages. The third level considers the languages which have some amount of training data whereas the fourth level talks about languages that consist of lexical resources. The fifth and sixth level describe languages which do not have digitized datasets or do not have standard writing scripts at all. On this note, Jimerson and Prud'hommeaux, 2018 considered languages as low-resourced languages which do not have extensive parallel corpora in digitized form for their work. According to Berment, 2004, languages which have resources like tools, and digitized corpora less than a threshold can be considered low-resource languages. But the concept of digitized resources is very much time-dependent. Because even for a high-resource language like English, in the domain of automatic hate-speech detection, before the introduction of social media sites, it was hard to get adequate training datasets. But now, it is quite possible to build these systems for many languages including English, Chinese, German etc. Cieri et al., 2016 also describe languages as low-resourced languages which have few computational data sources. So, based on the theories of Berment, 2004 and Alegria et al., 2011 for this thesis we consider languages as *low-resourced* if there exist a small amount of parallel corpora and few monolingual computational data resources to train a distributional NLP system along with the development of very few human language technologies.

While we talk about low-resourced languages, there are multiple domain-based NLP applications which are low-resourced. Duong, 2017 explained a language can be considered as low-resourced based on different existing NLP applications. While this hypothesis can be argued with the introduction of multilingual language models that maps languages in a shared vector space, in some domains like medical or material sciences, NLP applications are still explored majorly in English. Haffari et al., 2018 organized a dedicated workshop where researchers expressed their concerns on multilingual domain adaptability of NLP applications other than the English language. Most of the multilingual distributional models are quite inefficient in these domains due to a lack of multilingual training resources even in some high-resource languages. For example, it is quite challenging to build a material science domain NLP application in Arabic due to the lack of annotated training datasets available (Almansor et al., 2019). Following these theories, in our thesis, we have identified applications as *low-resource multilingual NLP applications* if there are very few annotated parallel or monolingual training datasets available to train distributional models.

This thesis is about designing novel deep learning algorithms to achieve high performance in different low-resource scenarios. On this note, instead of concentrating on a particular language family or domain, we design algorithms which can be deployed to a wide range of low-resource NLP applications and languages.

2.2 MACHINE LEARNING

In this section, we will elaborate on the machine learning methodologies. According to Bhavsar et al., 2017, "*Machine learning is a collection of methods that enable computers to automate data-driven model building and programming through a systematic discovery of statistically significant patterns in the available data*". In textual NLP, machine learning algorithms are trained to capture the relation between words or sentences. In recent times, NLP problems are either being tackled with classical machine learning algorithms or using advanced neural network algorithms called deep learning. Throughout the thesis, we will use different fundamental aspects of machine learning techniques to build our novel deep learning models. Moreover, we have also used some classical and statistical machine learning algorithms as baseline models in the thesis.

Machine learning algorithms encode the textual or other forms of information as vectors[3] $x \in \mathbb{R}^d$ where d denotes features of the dataset. In a general setup, the row vectors represent the full information in a dataset whereas the column vectors denote the features of each information in the dataset. At the time of training, this information is passed as matrices and is divided into batches[4]. Machine learning algorithms are organized in mainly three categories:

- **Supervised Learning**, is when the desired training dataset consists of both inputs and labels. The supervised training algorithms demand gold standard labels as classes annotated by annotators. In this training process, a function is learnt based on the features and annotated classes to predict the output of an unseen input. The effectiveness of the model is determined by the accuracy of predicting the output of the new input. These algorithms are mostly probabilistic and the effectiveness depends on the data distribution of the classes in the training set. In low-resource NLP application training such supervised algorithms is problematic because of the scarcity of the annotated training datasets as well as the amount of effort of the annotators needed for the dataset creation. For example, there exist supervised semantic labelling tasks in English trained on Penn Treebank (Marcus et al., 1993), but annotation of these 4.5 million tokens required over 3 years. Moreover, for many low-resourced languages collecting and building such a tree-bank is a quite hard task to complete. On this note, in Chapter 5, we observed that some state-of-the-art supervised models are comparatively less effective on low-resourced languages.

- **Unsupervised Learning**, is when the desired training dataset only consists of inputs. The algorithms are designed to learn the structural or other meaningful features in the dataset. While seeing new inputs, the algorithms try to find commonalities of the features with the training datasets and group them accordingly. Mostly unsupervised algorithms are designed to perform clustering or feature grouping of the dataset. The mathematical base of these algorithms is mostly based on finding the probability density functions for events. The probability density function can be defined as providing a probabilistic likelihood of the closeness of a random variable to a sample space (Grinstead & Snell, 2009). Researchers used unsupervised learning in many linguistic applications including grouping words based on morphosyntactic features (Biemann, 2006b; Christodoulopoulos et al., 2010). Unsupervised learning algorithms for most of the NLP applications are not as efficient as supervised learning algorithms (Blunsom & Cohn, 2011). But for low-resourced languages developing efficient unsupervised algorithms can be beneficial for many NLP applications (Aharoni & Goldberg, 2020). In this thesis, we have explored and designed novel unsupervised algorithms which are probabilistic, to produce efficient results for low-resource NLP applications and languages using limited training resources.

- **Semi-Supervised Learning**, is when the algorithms are trained using unlabelled and some of the labelled training dataset. This opens up the possibility of using the semi-supervised learning in some NLP applications having a small amount of training datasets. Recently data augmentation methods are explored for semi-supervised learning in NLP applications (J. Chen et al., 2020). These algorithms are also used to produce self-training methods for NLP which opens up another domain of NLP which is transfer learning (Blitzer & Zhu, 2008). In Chapter 6, we have implemented the theory of this self-training (or self-learning) methodology for the training of our algorithms. We will describe the background behind transfer learning in the later part of this chapter.

3 Different machine learning frameworks represent the vectors differently. For example, Pytorch encodes the information as a tensor where as Keras framework encodes it as a numpy array.

4 https://towardsdatascience.com/batch-mini-batch-stochastic-gradient-descent-7a62ecba642a

There is another type of machine learning methodology that is getting used in NLP applications called *weakly-supervised learning*. These algorithms use noisy, limited labelled resources as a training dataset to train the models. Unlike semi-supervised learning, these algorithms can exploit the different types of limited supervision including the semantically related word or sentence mapping. In recent times, researchers have designed algorithms for different NLP applications using the semantic relatedness between sentences with the help of weak supervision. Min et al., 2019 have designed a question-answering system which understands the relation between questions and answers of the training dataset with the help of weak supervision. The trained model then generates answers for the unseen questions. Qin et al., 2021 also proposed an automatic question-answering system based on weakly-supervised question-answer semantic relatedness without defining gold standard examples. Inspired by these algorithms in Chapter 5, we designed a weakly-supervised cross-lingual sentence embedding framework without having any parallel training dataset.

With the advancement of machine learning algorithms, the implementation in the NLP domain has been applied to many tasks including classification and regression. Some other types of NLP applications include text generation and cross-lingual text matching for information retrieval applications. In this thesis, we mainly focused on classification and cross-lingual text matching tasks. We will discuss the brief descriptions of these tasks below:

- **Classification:** These tasks are designed to predict gold standard label y based on input features x. The label y belongs to the existing classes or categories. Mostly classification tasks are restricted to a defined set of training dataset. Classification tasks can be further sub categorized into two categories: **(i) binary classification**, when there are only two classes as gold standard labels, **(i) multi-class classification**, when more than two classes are considered as labels. For multi-class classification, in best case scenarios, every input feature x is mapped to one gold standard label y.
- **Cross-lingual text matching:** The cross-lingual text matching task is defined as identifying the best match target parallel sentence t for the source language s. In some of the previous works, the accuracy of these matching tasks is defined by Spearman rank correlation (Reimers & Gurevych, 2020).

In this thesis we have tackled mostly cross-lingual multi-class classification tasks and cross-lingual text matching tasks.

2.2.1 Loss Functions

The learning methodology of the above-mentioned tasks for different machine learning algorithms is mostly dependent on how close the machines can predict the output with respect to the gold standard sets. Mostly in classification tasks the supervised and semi-supervised models estimate the probability $p(x)$ of a given input x and evaluate the deviation from the gold standard labels y using loss functions. With the help of different optimization techniques, the algorithms then learn to reduce the error while predicting the output. In this section we will introduce readers to the different loss function techniques:

- **Mean Squared Error Loss (MSE):** Mean squared error is computed to understand how far the estimated probability y' differs from the true values y. It is computed irrespective of the direction and focused on the average magnitude of the error. The advantage of computing MSE loss is that highly deviated estimated values are heavily penalized due to the calculation of the square between the distance. The MSE loss is defined as

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y' - y)^2 \tag{2.1}$$

- **Mean Absolute Error (MAE):** Mean absolute error is absolute the measurement calculation of the deviation of the estimated probability y' with respect to the true values y. Like MSE, in this case, the distance calculation does not need to take the direction into account while calculating the loss. In theory, the MAE loss is more suitable and effective to define the outliers as it does not calculate the square of the distance. The MAE loss is defined as

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|(y' - y)| \tag{2.2}$$

- **Mean Bias Error (MBE):** This error function gives us an understanding of how biased the system is with respect to the true values y while predicting y'. This is somewhat similar to MSE loss but the major difference is the distance is not squared which also asks the researchers to be cautious about the negative loss value. The MBE loss is defined as

$$MBE = \frac{1}{n}\sum_{i=1}^{n}(y' - y) \tag{2.3}$$

- **Hinge Loss:** One of the widely used loss functions for boundary-based classical machine learning algorithms is Hinge loss. This pushes the model to represent the positive examples with a higher score than the negative examples. This is controlled by a margin function δ. As a result, this loss function is used mostly for the margin-classification tasks and is widely used by classical machine learning algorithm *Support Vector Machine (SVM)*. This loss function is used on some of the very notable NLP tasks including learning cross-lingual word embeddings. Though this is not differentiable, the convex characteristics of the loss function increase the usability in the machine learning domain. The loss function is defined as

$$HLoss = max(0, \delta - y_p + y_n) \tag{2.4}$$

- **Cross-entropy Loss:** Cross-entropy loss is designed as a logarithmic loss to calculate the difference between the predicted value y' with respect to the true value y. The loss function is mostly designed for classification tasks. The multi-class classification tasks have more than two classes whereas binary classification tasks have two classes having values of 0 or 1. The cross-entropy loss is defined as

$$l = \frac{1}{n}\sum_{i=1}^{n} y_i log(y_i') \tag{2.5}$$

  When there are two classes the loss is calculated with binary cross entropy loss which can be calculated as

$$l_b = \frac{1}{n}\sum_{i=1}^{n} y_i log(y_i') + (1 - y_i) log(1 - y_i') \tag{2.6}$$

While we have discussed the loss functions for model training, the training of these models use gradient descent methodology which we will discuss in the next section.

2.2.2 Gradient Decent

The gradient descent is defined as "*an optimization algorithm for finding a local minimum of a differentiable function*" (Donges, 2019). This is one of the most popular and effective optimization techniques to train a machine learning algorithm. Mathematically it can be described as minimizing an objective function $J(\theta)$

while updating the model parameter θ in the opposite direction of the gradient of the function. As the gradient is the partial derivative of the function, the i-th element is the partial derivative of the function $J(\theta_i)$ (Ruder, 2019). Updating the model parameter θ can be achieved using Equation 2.7

$$\theta = \theta - \lambda \cdot \nabla_\theta J(\theta) \tag{2.7}$$

where λ is the learning rate which decides the magnitude while updating the model parameter. In naive terminology, the above equation gives the direction for the steepest descent until it reaches the position to get the lowest cost function while training the machine learning model. In real-world training examples, one of the most important factors for efficient learning is to set the ∇ correctly. Often to avoid the models getting stuck into local minima points of gradient descent, the learning rate needs to be reduced or increased. In our training methodologies (refer to Chapters 4, 5, 6), we have set the learning rate based on validation results. There are mainly three types of gradient descent:

- **Batch gradient descent:** Batch gradient descent is the vanilla gradient descent where parameters of the models are updated after the loss is calculated and accumulated on the total training dataset. The disadvantage of this process is clearly the amount of memory it demands while updating. Also, it takes more time for one cycle to complete. On the other hand, it produces stable and more accurate gradient as it sees all the training sets in one cycle. It results in a stable convergence. The batch gradient descent can be calculated as

$$\nabla_\theta J(\theta) = \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta l(x, y, y', \theta) \tag{2.8}$$

- **Stochastic Gradient Descent:** Stochastic gradient descent calculates gradient and updates for each example in the training set. This makes the stochastic gradient descent faster than batch gradient descent methodology while demanding less computational memory to perform calculations. Due to single example updating, it is easier to track the changes in parameters. The stochastic gradient descent can be calculated as

$$\nabla_\theta J(\theta) = \nabla_\theta l(x_i, y_i, y_i', \theta) \tag{2.9}$$

- **Mini-batch gradient descent:** Mini-batch gradient descent is widely used to train neural networks. It combines the concept of both stochastic gradient descent and batch gradient descent which ensures efficient performance and stability. It splits the training dataset into mini-batches (the size of the mini-batch depends on the size of the training dataset) and performs updates on the batches. In real-world applications, mini-batch is used as the default settings to train complex models. The mini-batch gradient descent can be calculated as

$$\nabla_\theta J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta l(x_i, y_i, y_i', \theta) \tag{2.10}$$

  where m is the size of the mini-batch.

2.2.3 Clustering

Ezugwu et al., 2022 state clustering technique "*deals with the data structure partition in an unknown area.*" It further can be described as an algorithm to group similar objects together. Unlike supervised classification techniques, clustering algorithms do not consider gold standard levels while training and can be considered

an unsupervised machine learning technique. Ideally, every cluster should contain data points having the same characteristics and similarities. There are various types of clusters. Mostly these algorithms calculate distances between data points to construct the clusters. The most common clustering techniques include:

- **Hierarchical Clustering:** This algorithm is designed to consider the hierarchical relationship between the data points to group them. There are mainly two types of approach: *top-down approach* and *bottom-up approach*. In the *top-down approach* the central cluster is divided by selecting the furthest data point from the centre as the second cluster centre. The data points are then assigned to close centres as two different clusters. This process happens in a finite number of cycles to form other clusters. In *bottom-up approach*, the close clusters are merged and produce a new cluster. This process helps to reduce the number of clusters keeping the groups of data points distinguishable from each other. Hierarchical clustering is a tree structure-based method and is mostly suitable for hierarchical data points. BIRCH (T. Zhang et al., 1996), CURE (Guha et al., 2001), and ROCK (Guha et al., 1999) clustering algorithms are some of the examples of this category. The advantage of using these clustering techniques is the scalability is high and it is comparatively easy to find the hierarchical relationship between data points.

- **Partitioning Clustering:** Partition-based clustering is a well-known clustering technique in the NLP community. At the start of the algorithm, a finite number k is defined and the data points are then divided into this k cluster centres. The aim is to construct the clusters in such a way that the metric of data points to the nearest cluster is minimized. In this process, the cluster centres are shifted until the convergence. A very common partition-based clustering algorithm is k-means clustering (MacQueen, 1967). In this algorithm, the mean of all the data points in a cluster is considered the cluster centre and this centre is updated through iterative calculations. In this thesis, we have used k-means clustering techniques to group the language data points (refer to Chapter 4).

- **Density-based clustering:** The idea behind this algorithm is data points with high data density should be placed in the same cluster. Very popular clustering algorithms based on such methodology are DBSCAN (Ester et al., 1996), OPTICS (Ankerst et al., 1999) and Mean-Shift (Comaniciu & Meer, 2002). Among them, the DBSCAN method is widely used across domains. In this algorithm, predefined neighbouring k data points are objectified and they form a cluster if they have a smaller distance than threshold value ϵ. In the iterative clustering process the data points that are close to the cluster centre, are added to the cluster. On the other hand, the OPTICS algorithm is an improvement over DBSCAN and has overcome the sensitivity of using these two parameters k and ϵ. These algorithms are highly efficient clustering techniques and can be used widely in any data format. But one of the major setbacks is that they require huge computation memory to perform the calculations and are not very effective when it comes down to low-resource data clustering.

2.2.4 Neural Networks

Neural network (NN) has now become the most popular methodology for NLP applications. This is also known as *Artificial Neural Network (ANN)*. Mathematically, neural networks can be considered complex calculations of many functions. A vanilla neural network system consists of an input layer, hidden layer and output layer. A big resemblance can be found with the classical machine learning algorithm *logistic regression*. In logistic regression, we compute a function $f(x)$ over the input x multiplied by randomly initialized weights W using

$$f(x) = Wx + b \tag{2.11}$$

On the output layer, we compute the probability of the function $f(x)$ using a *softmax* function to predict the output. The hidden layers in a neural network consist of such functions mapped through an *activation function* to achieve the non-linearity of the inputs. The hidden layers have multiple neurons which are initialized with a random weight matrix W. On the output layer to obtain categorical distribution, the commonly used function are *softmax* or *sigmoid*. A neural network having a single hidden layer is known as *one-layer feed-forward neural network* whereas a network having multiple hidden layers is called *Multi-layer Perceptron (MLP)*. The function $f(x)$ over the input x can be calculated as

$$f(x) = a(Wx + b) \tag{2.12}$$

where a is the *activation function*, W is the weight matrix and b is bias. This non-linear output is then passed through *softmax* or *sigmoid* to get the output y using

$$y = \mathrm{softmax}(W'f(x) + b') \tag{2.13}$$

For binary classification, two classes are considered as gold standard classes, 0 and 1. To keep the probability of output y in between $(0, 1)$, the non-linear output is passed through the *sigmoid* function, which can be defined as

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.14}$$

One of the advantages of using multiple layers is that they can share parameters among themselves injecting the option of having inductive biases. The introduction of these inductive biases helps to generalize better. Such parameter sharing technique we have used in our thesis using multi-task learning (Chapter 5) or transfer learning (Chapter 6). Mapping inputs through hidden layers to get output y from the output layer is known as *forward propagation*. Throughout the thesis, we will see multiple test case results obtained from a single forward propagation. There are some other activation functions including *ReLU* and *tanh*. The *ReLU* activation function pulls out the max between 0 to value x using

$$a(x) = \max(0, x) \tag{2.15}$$

The advantage of using a piece-wise linear *ReLU* activation function can be seen to give either 0 or the input value which makes data point characteristics more usable. But it is hard to map the negative data points as *ReLU* outputs negative values as zeros. On the other hand *tanh* function, being a sigmoidal shaped, keeps the range between $(-1, 1)$. This function is differentiable. *tanh* maps the negative data inputs strongly negative whereas the zero inputs will be mapped near to zero. The *tanh* function can be calculated as

$$a(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.16}$$

Over time, different neural network models including *Recurrent Neural Network*, *Convolution Neural Network*, *Transformer Architecture* have become widely used in the NLP community. In this thesis, we have used these methodologies to build different NLP applications. Before that, we will discuss another important methodology to train the neural network, *backpropagation*.

2.2.5 Backpropagation

Backpropagation (Rumelhart et al., 1986) is at the heart of neural networks. As we discussed in Section 2.2.2, in practice the neural networks are trained using stochastic gradient descent. While training, the gradients of the loss are calculated with respect to the gold standard labels using the methodology called *backpropagation*. The idea behind the backpropagation algorithm is the chain rule of mathematics. Given output y and input x, if we want to calculate the derivative of z with respect to x where $z = f(g(x))$ and $y = g(x)$ is satisfied, then the chain rule says

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx} \tag{2.17}$$

This rule is also applied when it comes to the partial derivative of z with respect to x. This partial derivative can be considered as the gradient of z with respect to x (the partial derivative of z is calculated with respect to each x_i) (Goodfellow et al., 2016; Ruder, 2019) and can be written as

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j}\frac{\partial y_j}{\partial x_i} \tag{2.18}$$

Let us take the model produced output y' and gold standard values y. The initial loss function obtained by the model from the first feed forward path can be presented as $L = l(y', y)$. As we mentioned in Section 2.2.2, the gradient can be represented with ∇ and the gradient of the loss with respect to y' can be calculated as

$$\nabla'_y L = \nabla'_y l(y', y) \tag{2.19}$$

Following the rule of backpropagation, this calculation starts from the loss function with respect to the produced output y' and follows the chain rule as shown in Equation 2.18. It then calculates the gradient on the hidden layer representations and parameters and goes until the input layer. The chain rule-based computed gradients are then used to update the parameters of each layer using methodologies introduced in Section 2.2.2.

2.2.6 Recurrent Neural Network (RNN)

Recurrent neural networks or RNNs (Elman, 1990) are one of the most widely used methodologies for different NLP applications. Texts are sequential in nature and carry meaningful information over different timesteps. An RNN encodes the information sequentially keeping the link between words or characters. It also shares parameters among the different hidden layers which helps to carry the semantic meaning of the words in a sentence. Rather than accepting inputs at first like a vanilla neural network, RNN encodes inputs at each layer over the time period. The hidden states of RNN h_t over the time step t act as "internal memory", which helps to keep the information of the previous timestep. Thus while predicting the next word of a sentence, the semantic and syntactic relationship between each component of a sentence helps to produce meaningful sentence representations. Considering t as a timestep, the operation of RNN can be calculated as

$$h_t = a(W_h x_t + U_h h_{t-1} + b_h) \tag{2.20}$$

$$y_t = a'(W_y h_t + b_y) \tag{2.21}$$

where a and a' are the activation functions. The previous hidden step information h_{t-1} is added with the current timestep input x_t to get the current hidden step h_t. This highlights the capability of RNN of keeping the previous semantic information intact while generating the current hidden step. For each input, there is an output y_t which helps to implement many NLP applications including token classification. The backpropagation methodology of RNN to compute the gradients is called *backpropagation through time* (Werbos, 1988). In this process, the gradient computation considers the past time step path. This leads to one of the major shortcomings of RNN called *vanishing gradient*. During backpropagation the hidden representations are multiplied with the same weight matrices multiple times over the time step which makes the gradient values very small. As a result, the model is unable to learn during training. To solve this, researchers have designed the well-known deep network called LSTM.

2.2.7 Long Short Term Memory (LSTM)

Hochreiter and Schmidhuber, 1997a proposed the *Long Short Term Memory (LSTM)* architecture for better understanding of sequential data over recurrent neural networks. LSTM has produced efficient results for the NLP applications where long-term dependencies need to be preserved (Rohit et al., 2017). LSTM is an improved variant of RNN architecture having the capability of remembering or forgetting hidden representations. The LSTM network has a forget gate f_t, an input gate i_t and an output gate o_t. The gates interact with the inputs x_t, old cell state c_{t-1} and current cell state c_t. The forget gate is used to decide which non-essential information can be omitted during training. The design of forget gate includes a sigmoid layer as an activation function which works as a binary gate of 0 or 1. The forget gate can be calculated as

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2.22}$$

where h_{t-1} is the previous hidden representation and W is the weight matrix. The input gate selects the information to be updated. This also has a sigmoid layer like a forget gate, but at the same time it also generates a candidate cell state c' which will be added to the previous cell state value to update the current cell state c_t. The previous cell state gets updated using the forget gate f_t whereas the new input gate will work on the new candidate cell state c'. The current cell state, candidate cell state and the input gate can be calculated as

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2.23}$$

$$c'_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{2.24}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot c'_t \tag{2.25}$$

Finally, the output gate decides the output after the cell filtration. The output gate obtains the information from the previous representation h_{t-1} and the current input x_t and updates the current cell state c_t obtained previously using tanh activation function to push the value range between $(-1, 1)$. The output gate can be calculated as

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{2.26}$$

$$h_t = o_t \odot \tanh(c_t) \tag{2.27}$$

There are other variants of the LSTM including *stacked LSTM* and *bidirectional LSTM*. In bidirectional LSTM (Graves et al., 2013), the representations are calculated separately from the forward (h_{for}) and

backward (h_{back}) directions after which the representations are concatenated over the timestep to obtain the final representation using

$$h_t = [h_{for}; h_{back}] \tag{2.28}$$

In all of the above equations, $\odot$ denotes element-wise multiplication. In Chapter 5, we introduce a new model as sentence embedding technique where we have implemented the bidirectional LSTM to get the sentence representations.

2.2.8 Convolution Neural Network (CNN)

Convolution neural networks (CNN) (LeCun et al., 1998) are widely used in character based NLP applications. Primarily CNN was proposed for computer vision tasks; however, CNN produced efficient results for different NLP applications (Kim, 2014). The CNN architecture consists of the differently defined size of filters which works as a sliding window to get the textual representations. On a user-defined k-sized window for input x, each filter generates the new features c using the following Equation,

$$c_i = \sigma(wx_{i+k-1} + b) \tag{2.29}$$

where w is the weight matrix and c_i is the generated features on i-th input. The sliding window then generates the final feature vector c after calculating over each window of k size using the following Equation,

$$c = [c_1, ..., c_{n-k+1}] \tag{2.30}$$

Entries in the feature map contain the representation of the different segments of textual inputs. For different NLP applications, these feature inputs are then fed to another neural network system to get the final representation (refer to Chapter 5, where we have fed the features to a bidirectional LSTM network to get sentence representation) or commonly the min or max important feature output of these features are considered as the final representation of the input x (Kalchbrenner et al., 2014). The CNN architecture has some advantages over other sequential neural networks for representation-based NLP applications. It opens up the idea of designing sentence representations as the combinations of different n-gram units using different pretrained word embeddings for classification-based tasks (Dahou et al., 2016). These methodologies are also fast in calculation and require less memory for the computations. In our thesis we used CNN architectures to design deep algorithms (refer to Chapters 4,5,6).

2.2.9 Transformers

Vaswani et al., 2017a proposed a revolutionary methodology for NLP applications called *Transformer*. It is designed to solve sequence-to-sequence tasks. Due to the design of the architecture, it handles the long-term dependencies of the inputs while requiring less computational memory. Unlike LSTM, the transformer model only relies on a *self-attention mechanism* to compute the representations of the words or characters. In Figure 2.1, we can observe that the transformer architecture has two main modules: encoder and decoder. The main driving factor of this model is the multi-head attention mechanism. This multi-head attention block is the combination of multiple self-attention blocks. According to Vaswani et al., 2017b "*Self attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence*". This implies within the same sentence self-attention helps to capture the semantic relationship between the context word and the target word. This

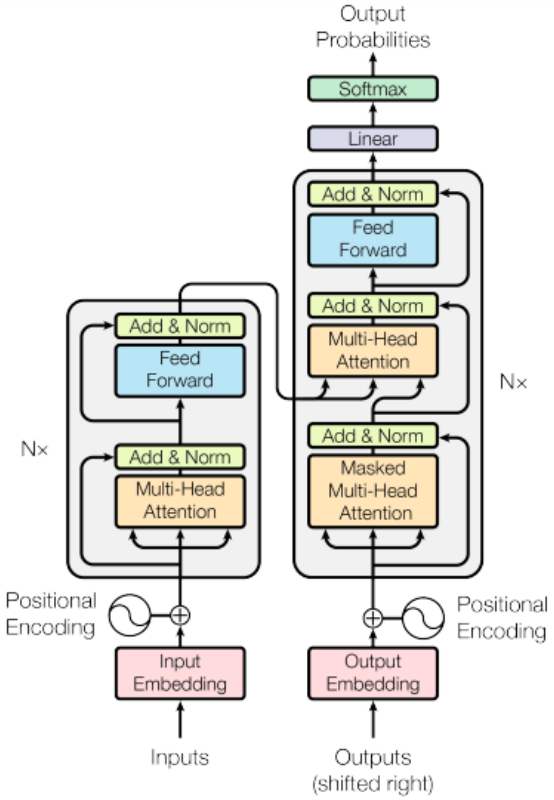

Figure 2.1: The architecture of the Transformer model (Vaswani et al., 2017b)

helps in different token based NLP applications including machine translation, token classification, content identification (Sarkar et al., 2021) etc. The driving parameters of calculating the self-attention mechanism in transformers are query (Q), key (K) and value (V). Dot products between the query and key vectors are calculated to figure out the most similar keys based on queries. *Softmax* probability of this dot product push the closest probable values close to 1 and dissimilar values close to 0. Then it is multiplied with the value vector to get the attention values of the inputs and outputs. All the queries, keys, and values are packed in matrices and can be calculated as

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.31}$$

where d_k is the dimension of the query and key. Instead of calculating a single self-attention function, different learned linear projections of dimension d_k, d_k and d_v are calculated in parallel as a multi-head attention process. Then all the calculated attention values are concatenated to get the multi-head attention values. In the decoder layer, a masked multi-head self-attention mechanism is calculated of the previous iterations which are then processed through a point-wise fully connected feed-forward neural network. The vanilla transformer architecture consists of 6 layers of the encoder-decoder stack whereas some of the state-of-the-art transformer applications like XLM-R (Conneau et al., 2020) have more than 12 layers. The transformer architecture is faster than the sequential LSTM calculation and handles the long-term dependencies more efficiently. In our thesis, we have used different variants of state-of-the-art transformer architectures for different NLP application experiments.

2.3 REPRESENTATION LEARNING IN NLP

Natural language processing exploits unstructured information to teach machines the characteristics of the languages. NLP systems either can be designed based on template-based feature engineering techniques (Hull, 1999; Prager et al., 1999; Y. Tang et al., 2020; Van-Tu & Anh-Cuong, 2016), or can be learnt using representation learning to overcome the shortcomings of feature engineering (Babic et al., 2020). In this section, we will highlight some of the representation-based development in the NLP field.

Representation learning helps to extract useful information from the data. Recently, the state-of-the-art deep neural systems, introduced in Section 2.2, help to extract the representations of the input dataset as features. These feature representations of input data enhanced the efficiency model performance in different NLP applications including unstructured text and image relation extraction (Jayagopal et al., 2022), and speech recognition (Srinivasan et al., 2022). Often NLP applications have multiple granularities. In this thesis, we consider the inclusion of multiple levels of language entries including character-based representations (Chapter 4), word-based representations (Chapter 6) or sentence-based representations (Chapter 5). Understanding the representation helps to get the semantic relation between these entries in the same vector space resulting in a complex semantic space. Jurgens, 2021 highlights the importance of capturing word representation using word embeddings for different NLP applications. Not only the languages but also different domain-specific NLP tasks demand semantic relation extraction across documents. Hussein and Hajj, 2022 highlight that representation learning can benefit domain base adaptations using non-linear mappings. Knowledge transfer between similar domain texts helps to capture semantic relatedness where the vocabulary is shared. In this thesis, we learn these representations using distributed representation learning, commonly known as embeddings.

2.3.1 Approaches for Representation Learning

Here we highlight some of the well known methodologies involved in representation learning for NLP:

- **Statistical Representation Learning:** The classical NLP techniques are enriched with statistical feature extraction techniques. In NLP applications the semantic and syntactic relations are extracted using different statistical calculations including frequency-based methods. The popular *bag-of-words* model represents the occurrence of all the words in a document (Goldberg & Hirst, 2017). It highlights the vocabulary structure of a document as well as defines the frequency of these words. The downside of the model is that it gives an overall structure of the vocabulary but is unable to describe the position of these words in the sentences of a document. The intuition behind finding this vocabulary set involves comparing different documents based on the existence of the common words. For large documents, scoring and collecting these vocabulary sets can be tricky and demand well-established statistical scoring techniques to perform the action. One of the popular scoring techniques is *TF-IDF* (Manning et al., 2008; Salton et al., 1975). In the normal frequency calculation of the bag-of-words model, words carrying less information, often the stop words, are pointed out as the most frequent words which do not carry any information or carry little information. The TF-IDF method helps to rank the informative words in a document. This TF-IDF method is called *Term-Frequency Inverse Document-frequency*, where *Term-frequency* scores the frequency of the words in the current document and *Inverse Document-frequency* measures distribution across documents. Despite these scoring techniques, the bag-of-words methodology is less efficient when it comes to long document calculations due to sparsity. Also, it fails to convey the semantic relations of the words in a document.

Another common approach to statistical word representation is *n-gram* representations of a word. An n-gram is the sequence of different n numbers of words. When n is 2, then it is called *bigram*, when n is 3, then it is called *trigram* and so on. Using the conditional probability, the probability of occurrence of a word is calculated base on other n words. For example, if a diagram is "he is", the probability of "is" will be $P(\text{is}|\text{he})$. The probabilistic dependency of one word on the previous word is referred to *Markov assumption* (Jurafsky & Martin, 2009). To generalize, the occurrence of a word can be represented with the previous $n-1$ words in a sentence using

$$P(w_n|w_{1:n-1}) \tag{2.32}$$

The main difference between bag-of-word methodology and *n-gram* methodology is, bag-of-words disregard word order while *n-gram* models consider word order. Though the statistical representation of words can be effective for many NLP applications, it is often hard to work with long documents. Moreover, these systems are not able to carry the meaning of the words and thus confine the usability of these systems to small NLP domains.

- *Distributed Representation Learning:* Distributed representation learning involves machine learning and state-of-the-art deep learning models to capture the hidden features in a text document (Ferrone & Zanzotto, 2019). All the algorithms we proposed in this thesis are based on distributed representation learning. We have introduced some of the state-of-the-art deep learning methodologies in Section 2.2 to learn the representations. We will go through the state-of-the-art representation learning methodologies along with *self-supervised* learning in Section 2.3.2

2.3.2 Distributed Representation Methodologies

Distributed representation learning involves state-of-the-art deep learning or machine learning techniques as feature extractors from the inputs. Using different optimization techniques the hidden representations from hidden layers of neural networks are considered as the final representation of the words or the sentences. Before going into the details of distributed learning, we will talk about one of the most widely used techniques called *one-hot encoding*. As for the bag-of-words methodology, one-hot encoding encodes a word w with a $|V|$ dimensional vector w, where V is the vocabulary set and each dimension of w represents binary number 0 or 1. If the word is present then it will be encoded as 1 otherwise it will be 0. The words are mapped to an index of the vocabulary and can be represented with the mentioned binary numbers. But, the one-hot encoding, as we discussed in the statistical representation learning methodology, does not convey any semantic or contextual information about the words in a sentence or document. Thus though the one-hot encoded vectors can be considered as embedding inputs to a shallow neural network, the models are unable to produce efficient outputs for the unseen dataset (Babic et al., 2020). Thus the semantic information about the words needs to be carried through the neural network to get efficient representations from the hidden layers.

The word representation learning techniques are trained using a *self-supervised methodology*. While learning word representation, the main bottleneck is there are no gold standard labels. To get the representations of the vocabulary words from the hidden layers of the neural network, labels are required to calculate the loss and backpropagate through the network. Using the self-supervised methodology, labels are constructed from the existing training dataset. Thus this methodology helps to understand the hidden representation of the texts without any manual intervention and prior knowledge. Most of the state-of-the-art semantic and contextual word embedding techniques are designed based on this self-supervision

methodology. In the next section, we will talk about different word-representation techniques present in the NLP community.

2.3.2.1 *Word2Vec Technique*

Mikolov et al., 2013a designed the *Word2Vec* methodology to learn efficient word representations from a large corpus. There are two methodologies to learn these representations called *Continuous Bag-Of-Words (CBOW)* and *Skip-gram*.

- **Continuous Bag-Of-Words (CBOW):** In this methodology, the representation of the input words can be learned based on the context words. The idea is to predict the target word v_t based on the context words window C as self-supervised learning by minimizing

$$l = -\frac{1}{|C|}\sum_{t=1}^{|C|} \log P(v_t|v_{t-C},...,v_{t-1},v_{t+1},...,v_{t+C}) \tag{2.33}$$

  where the probability of the target word with respect to the context words windows can be calculated as

$$P(v_t|v_{t-C},...,v_{t+C}) = \frac{exp(\tilde{v}_t^T v_s)}{\sum_{i=1}^{|V|} exp(\tilde{v}_i^T v_s)} \tag{2.34}$$

  where v_s is the sum of the word embeddings of the context words.

- **Skip-gram:** The skip-gram methodology does the opposite of CBOW methodology. Here the context word is predicted using the other information in the sentence. The context word embedding v_j can be calculated with respect to v_i as

$$P(v_j|v_i) = softmax(Wv_i) \tag{2.35}$$

  Similar to CBOW the model is trained by minimizing the loss function

$$l = -\sum_i \sum_{j(|j-i|\leq C, j\neq i)} \log P(v_j|v_i) \tag{2.36}$$

  where C is the window of context words. The skip-gram model can be considered as a single layer neural network without any hidden layer where the word embeddings of the input word v_i is fed into the softmax layer without passing though any hidden layer (refer to Figure 2.2). The skip-gram model is trained using *negative sampling* methodology. Negative sampling, during training, forces the model to distinguish a target word v_j from negative samples. These negative samples are drawn from a noisy distribution which is calculated as the unigram distribution of the words to the power of $\frac{3}{4}$. The fractional number was adopted based on the experimental results.

2.3.2.2 *Glove Embeddings*

Being a shallow window-based architecture, the CBOW and skip-gram model fail to accumulate global information for the context word from the corpus (Z. Liu et al., 2020). To capture the global information, Pennington et al., 2014 have proposed *Glove* word embeddings. The model is trained to minimize the

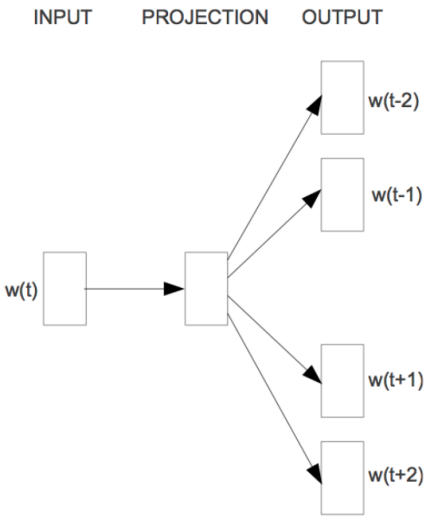

Figure 2.2: The architecture of the skip-gram model (Ruder, 2019)

difference between the dot product of the input and context words and the logarithmic value of the number of co-occurrences of these words in a fixed context window using

$$l = \sum_{i,j=1}^{|V|} f(C_{ij})(v_i^T \tilde{v}_j + b_i + \tilde{b_j} - \log(C_{ij}))^2 \tag{2.37}$$

where b_i and $\tilde{b_j}$ are the biases of the input word and the context word respectively. C_{ij} denotes the co-occurrence of the input word v_i and context word v_j in a fixed size context window and f acts as a scoring function to assign weights to the input and context word co-occurrences based on their frequency.

2.3.2.3 *FastText Embeddings*

The previous two static embedding methods described in Section 2.3.2.1, 2.3.2.2 fail to understand the internal structure of the word. Moreover, there exists one major shortcoming of these methods with *out-of-vocabulary* words. Bojanowski et al., 2017 proposed a new methodology called **FastText**, which is capable of capturing the sub-word level information of a word, resulting in injecting structural information of the words in the shared embedding space. The words are divided into subwords using user-defined n-grams and then trained using a skip-gram model to learn the word embeddings. This n-gram methodology works as a sliding window feature extractor of the words. Dividing the words into different subwords also helps to encode rare words which are not in the vocabulary.

2.3.2.4 *Contextual Embeddings*

Understanding the context of the words in a sentence is very important for different downstream NLP applications. The static word embedding techniques we discussed so far are unable to convey the contextual meaning of the words with respect to the surrounding words in a sentence. The training methodology of these models captures a unique representation of each word in the corpus and is thus unable to understand the semantic meaning of the words. However, the contextual word embedding methodologies can represent the words according to their contexts in the sentence. Peters et al., 2018 proposed *ELMo*, a contextual word embedding technique designed using bidirectional LSTMs. It converts the words into low-dimensional

word vectors using the context word and the surrounding words of a sentence. It captures the conditional probable representation of each word with respect to the previous words using

$$P(w_1, w_2, ..., w_n) = \sum_{i=1}^{n} P(w_i | w_1, w_2, ..., w_{i-1}) \tag{2.38}$$

The bidirectional LSTM captures the contextual meaning of the word with respect to the forward and backward surrounding words in a sentence. It combines all layer representations of the LSTM to compute the task-specific weighting of all the layers using

$$w_k = \alpha^{task} \sum_{j=0}^{L} s_j^{task} h_{k,j}^{LM} \tag{2.39}$$

where w_k is the representation for each word in the sentence, α^{task} is the task-specific weight of the word vectors and s_j^{task} is the softmax normalized weights of the layers. The $h_{k,j}^{LM}$ is the context-dependent word representation generated by the bidirectional LSTM as described (the working process of LSTM is described in Section 2.2.7). The word representation is generated by j-th layer of the LSTM for the word w_k.

Later Devlin et al., 2019 has come with an improved contextual language model called *BERT*. *BERT* is based on transformer architecture (refer to Section 2.2 for transformer architecture) and trained in a self-supervised way. It uses the *masked-language-model* pretraining objective. During training, it randomly masks some of the words in the sentence and forces the model to predict the masked words based on the context. In addition to this, it also incorporates "next sentence prediction" methodology to capture the interrelation between the sentences. BERT, being a bidirectional transformer architecture, is able to capture the left and right contexts of a word in the sentence.

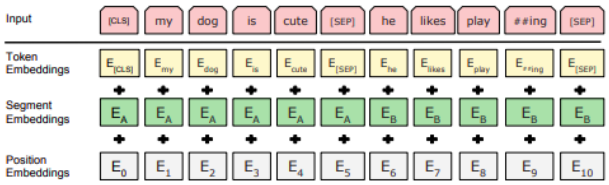

Figure 2.3: The architecture of the BERT model (Devlin et al., 2019)

Figure 2.3 depicts the architecture of the BERT model which consists of positional embeddings, token embeddings and segment embeddings. With the next sentence prediction methodology, the BERT architecture consists of two sentences named S_A and S_B. The segment embeddings are initialized with E_A and E_B respectively. These embeddings are trainable embeddings which along with position and token embeddings provide contextual information of a word in a sentence. The special token *[CLS]* denotes the start of the sentence whereas the *[SEP]* token highlights the end of each sentence. The training strategy of BERT consists of two stages, *pre-training* and *fine-tuning*. In this thesis, we followed these steps while training our novel algorithms. The *pre-training* step trains the model on a big corpus in a self-supervised setup. In the *fine-tuning* stage, the pretrained generic language model is further tuned with the desired application-specific training dataset. Thus the contextual word embeddings carry the semantic meaning and relatedness of the words for different sentence structures. Interestingly the *[CLS]* is used as a sentence representation while finetuning on the downstream tasks. However, researchers have argued about the efficiency of this sentence representation and proposed some other methods including max-pooling and mean-pooling of the final layer of the BERT representation (Reimers & Gurevych, 2019). Later, based on the BERT architecture

some other contextual word embedding techniques were proposed which includes *RoBERTa* (Y. Liu et al., 2019), *DistilBERT* (Sanh et al., 2019b).

2.3.2.5 *Cross-lingual Embeddings*

Cross-lingual word embeddings map multiple languages in the same vector space identifying the semantic similarities and relatedness between them (Ruder et al., 2019). In this thesis, we talk about low-resourced NLP applications and languages, and cross-lingual embedding methodologies play a significant role in that. It also enables the linguistic knowledge transfer between languages to perform downstream NLP tasks. Researchers in the past have has developed language-specific word embeddings (Cui et al., 2021; Douka et al., 2021; Tanvir et al., 2021) using the methodologies we described above. But the monolingual embeddings are unable to connect the semantic relationship between the languages in the same vector space. Later, using different parallel mapping and alignment techniques, the languages are mapped to the same space. In a traditional approach, high-dimensional context counting vectors of the co-occurring multilingual words are mapped into a single vector space using a bilingual dictionary (Gaussier et al., 2004; Tamura et al., 2012). Artetxe et al., 2016 proposed a bilingual dictionary mapping based bilingual word embedding generation technique which preserves the monolingual invariance. Their learning methodology is mostly based on the CBOW method. These methods still require an efficient annotated parallel dataset or bilingual dictionaries which mostly are not available for low-resource languages. Later the self-supervised method based static word mapping technique, commonly known as *vecmap*, was trained using large corpora and somewhat helped to generate word embeddings for the low-resourced languages (Artetxe et al., 2016). But these models are unable to capture the contextual relatedness of multiple languages in the same vector space.

Later based on the BERT language model, Devlin et al., 2019 published a multilingual BERT model called **mBERT**. The model is trained for more than 100 languages on a corpus consisting of Wikipedia articles of these languages. The vocabulary is shared across languages. However, the most efficient multilingual language model which suppressed the performance of **mBERT** model was *XLM-R (XLM-Roberta)* (Conneau et al., 2020). The model is trained with the "masked-language-modeling (MLM)" objective. But unlike the BERT model, it does not consider "next sentence prediction" methodology as the experimental results without next sentence prediction task highlight the improvement in the efficiency of the model for downstream tasks. It follows the architecture of the predecessor model *XLM* (Conneau & Lample, 2019)

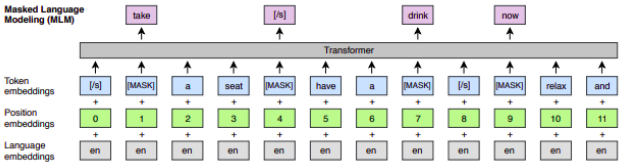

Figure 2.4: The architecture of the XLM model (Conneau & Lample, 2019)

(refer to Figure 2.4) with some conditional changes during training. Unlike *XLM*, it does not encode the language embeddings which increases the efficiency of the model during intra-sentential code-switching. The shared vocabulary is created using the *Byte Pair Encoding (BPE)* methodology (Sennrich et al., 2016). The model accepts text streams of sentences of size 256 tokens as input during training. The training dataset consists of a common crawl dataset for 100 languages. The authors used an in-house language identification model to identify language-specific training dataset. The model has produced state-of-the-art results for different benchmark sentence and token classification datasets. In all the cases, by default, the authors

have chosen the *[CLS]* token as the sentence representation. In our thesis, we incorporated the *XLM-R (XLM-Roberta)* model while designing our sentence representation model in Chapter 5.

2.3.2.6 *Sentence Representations*

Sentence encoders represent sentences by encoding the contextual or static word embeddings into a single vector x. Different machine learning models, introduced in Section 2.2, produce sentence embeddings differently. CNN networks accept the contextual or static word embeddings as inputs and extract local features by different sizes of filters. On a vector sequence w, the convolution operation y can be encoded as

$$y = \text{CNN}(w) \tag{2.40}$$

Mostly a max-pooling operation on the feature y will produce the sentence embedding by selecting the maximum element from the region of the feature map using

$$x = max(y) \tag{2.41}$$

On the other hand using the LSTM network, the temporal features of the input word embeddings can be extracted. Each input word embedding will be put through the LSTM cells step by step. At every time stamp, the LSTM will produce the hidden representation of each word having the semantic understanding of the previous words using

$$h_t = \text{RNN}(w_t, h_{t-1}) \tag{2.42}$$

where w_t is the weight of the t timestep and h_{t-1} is the hidden representation of the previous word. The sentence embedding can be achieved by either accessing the last hidden representation h_t or by extracting the mean or max hidden representation of total timesteps using

$$x = max(h) \tag{2.43}$$

For contextual language models including *BERT*, *mBERT*, *XLM-R*, the special token *[CLS]* is considered as the representation of the sentence. Reimers and Gurevych, 2019 proposed that the maximum or mean of the word embeddings of the last layer can be considered as the more efficient representation of the sentences.

2.3.2.7 *Document Representations*

Document representation aims to encode the whole document in a real-valued representation vector. Recently, document encoding have become popular for many NLP applications including document summarization, question answering, and paragraph summarization. Traditional document encoding technique include *one-hot* vector representation and achieved effective results for NLP tasks (Takase & Kobayashi, 2020). But this methodology has disadvantages and thus the modern neural model techniques (described in Section 2.2) are adopted to encode documents. In the next section we will discuss *one-hot* vector representation.

One-hot Vector Representation: Documents can be represented with a *bag-of-words*. Following the standard *one-hot* vector representation technique, for a document d_v consisting of vocabulary set $V = [v_1, v_2, ..., v_{|V|}]$, the one-hot vector of a word v can be represented as $v = [0, 0, 1, ..., 0]$. Based on this assumption a document with V vocabulary can be represented as

$$d = \sum_{i=1}^{n} v_i \tag{2.44}$$

where n is the length of the document. But this methodology does not take into account how important the word is in the document. That is, the word frequency in the vocabulary is ignored. Therefore, the *Inverse document frequency (IDF)* method can be adopted to understand the importance of a word in the document using

$$idf_v = \log \frac{|d|}{df_v} \tag{2.45}$$

where $|d|$ is the number of all documents in the corpus and df_v represents the document frequency of the word v. Thus a document can be represented using

$$d = d \odot idf \tag{2.46}$$

This methodology has the advantage of being simple and efficient in many small NLP applications. Moreover, this can be used as a feature extractor from a document. But this has some major disadvantages. The word order and context, in this case, are not considered. Moreover, the absence of semantic meaning in the words makes this methodology less efficient in many real-time NLP applications.

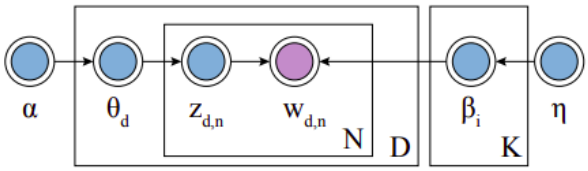

Figure 2.5: The architecture of the Latent Dirichlet Allocation (Z. Liu et al., 2020)

Latent Dirichlet Allocation (LDA): One of the popular traditional algorithm to understand document's structural features is based on *probabilistic topic modelling* (Hua et al., 2020). The statistical algorithms are designed to extract the thematic feature of the documents including the connection between themes and with the time how they change. One of the very popular probabilistic topic modelling methodology is *Latent Dirichlet Allocation* (Blei et al., 2003). LDA methodology is designed based on *Probabilistic Latent Semantic Indexing method (pLSI)* (Hofmann, 1999) by injecting a *Dirichlet* to the model. It considers that a document has multiple topics in it. A topic can be considered as the distribution over the vocabulary. as LDA is a *generative probabilistic modelling*, the data comes from generative process includes *hidden variables* (Z. Liu et al., 2020). The generative process produces joint probability distribution for observed and hidden variables. The conditional probability distribution which is also called *posterior distribution*, is calculated on the hidden variables. In LDA methodology the observed variables are the word vocabulary whereas the hidden variables are the topics. The LDA methodology (refer to Figure 2.5) can be described by the following joint distribution

$$P(\beta_{1:K}, \Theta_{1:D}, z_{1:D}, v_{1:D}) = \prod_{i=1}^{k} P(\beta_i) \prod_{d=1}^{D} P(\Theta_d) \left(\prod_{n=1}^{N} P(z_{d,n}|\Theta_d) P(v_{d,n}|\beta_{1:K}, z_{d,n}) \right) \tag{2.47}$$

where β_K is distribution over vocabulary, θ_d is the topic proportion for the d-th document, θ_{dk} is the topic proportion for the topic k in the document d, $z_{d,n}$ is the topic assignment of n-th word in document d, $v_{d,n}$ is the n-th word from the vocabulary in document d. These various dependencies are specified by different distributions from LDA. These are encoded using different statistical methodologies using different mathematical and graphical forms. On the other hand, the conditional probability distribution is

calculated using the posterior distribution given the observed documents. The posterior distribution can be calculated as

$$P(\beta_{1:K}, \Theta_{1:D}, z_{1:D}|v_{1:D}) = \frac{P(\beta_{1:K}, \Theta_{1:D}, z_{1:D}, v_{1:D})}{P(v_{1:D})} \tag{2.48}$$

The numerator can be calculated as the joint distribution of the random variables where as the denominator is the probability of the observed variables.

Topic modelling algorithms are mainly of two types:

- **Sampling Algorithms:** The algorithm is designed by collecting samples from the posterior. The popular sampling method includes *Gibbs sampling* for LDA methodologies (Gao et al., 2016). The working methodology includes *Markov Chain* consisting of connected random variables with an inter-dependency.
- **Variational Algorithms:** The variational algorithms structure a parameterized family of distributions over the hidden structure and the closest unit to the posterior family member is selected.

2.3.2.8 *Paragraph Vectors*

While the probabilistic and bag-of-words model encodes small documents efficiently, the semantic relationship between words and sentences in the document is missing. Dai et al., 2015 proposed a *paragraph vector* methodology to map every paragraph to an unique vector (refer to Figure 2.6).

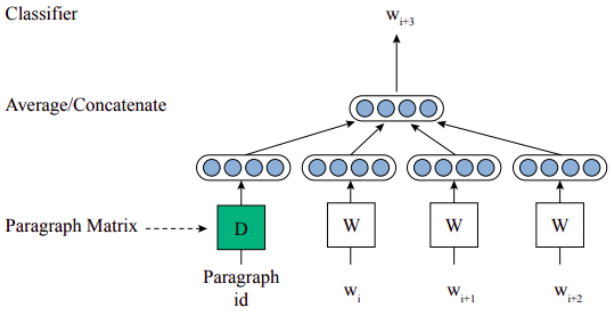

Figure 2.6: The architecture of the Paragraph vector (Dai et al., 2015)

The next context word vector prediction is designed to be calculated with respect to the previous words or paragraphs using the following Equation

$$y = Softmax(h(v_{t-k}, ..., v_{t+k}; E, P)) \tag{2.49}$$

where each word maps to a vector represented by the column in the word embedding matrix E, P is a matrix containing unique vector representations of paragraphs and h denotes the concatenation or average of the word vectors in the paragraph. On the other hand, the model tries to maximize the average log probability of the word vectors with respect to the training words using

$$\Theta = \frac{1}{l} \sum_{i=k}^{l-k} \log P(v_i|v_{i-k}, ..., v_{i+k}) \tag{2.50}$$

The last is a softmax layer used to predict the word using

$$P(v_i|v_{i-k}, ..., v_{i+k}) = \frac{e^{y_{w_i}}}{\sum e^{y_j}} \tag{2.51}$$

The paragraph token is considered as a word and thus it is used as memory function to remember the missing topic of the paragraph.

The paragraph vector has another type of methodology called *Distributed Bag-of-Words version of Paragraph Vector (PV-DBOW)*.

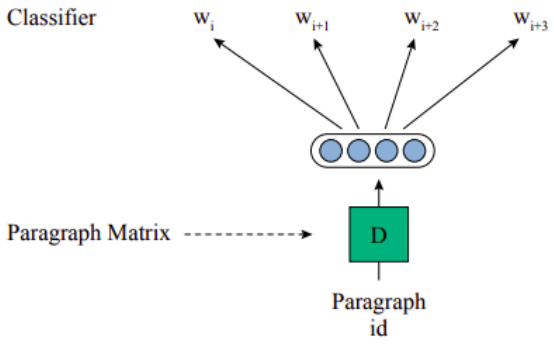

Figure 2.7: The architecture of the PV-DBOW of Paragraph vector (Dai et al., 2015)

As shown in Figure 2.7, the model is forced to predict the words from a paragraph. In a neural network-based training system, a text window is sampled and a random word from the chosen sample window is predicted as a classification task given the paragraph.

2.3.2.9 *Document-Context Language Model*

Recurrent architecture-based document representation learning produced efficient results for different NLP tasks. But backpropagation through timestep in RNN systems is quite tricky and can produce problems like vanishing gradient (as discussed in Section 2.2). To avoid this issue, Ji et al., 2015 proposed a new framework by capturing the context of the documents using a multilevel recurrent structure while encoding the vector space. Their *Context-to-Context Document Context Language Model (ccDCLM)* has the understanding of short-circuiting the RNN structure with the help of the sentence from the previous timestep to generate the words of the longer texts. The generation of the current timestep hidden state can be calculated as

$$h_{t,n} = a(h_{t,n-1}, f(w_{t,n}, c_{t-1})) \tag{2.52}$$

where a is the activation function, $h_{t,n-1}$ is the hidden state obtained from the previous timestep and c_{t-1} is the representation of the sentence for $t-1$ timestep. With the input of t timestep $x_{t,n}$, the previous timestep sentence representation is simply concatenated using

$$f(x_{t,n}, c_{t-1}) = [x_{t,n}; c_{t-1}] \tag{2.53}$$

The final prediction $y_{t,n}$ is computed using the last step representation $h_{t,n}$. The overview of the architecture can be found in Figure 2.8.

While the authors have worked on context-to-context model, they have proposed another variant called *Context-to-Output Document-Context Language Model (coDCLM)* (refer to Figure 2.9).

In this architecture the context is captured at the output level instead of the hidden state of the RNN. The hidden state representation of the timestep t can be calculated as

$$h_{t,n} = a(h_{t,n-1}, x_{t,n}) \tag{2.54}$$

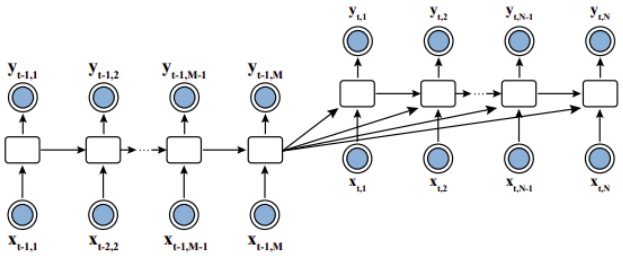

Figure 2.8: The architecture of the ccDCLM (Ji et al., 2015)

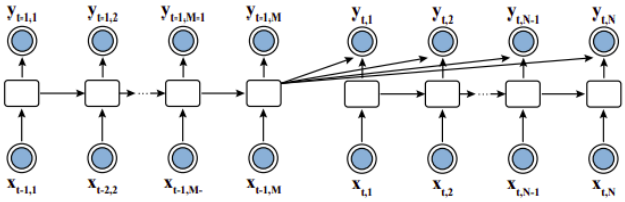

Figure 2.9: The architecture of the coDCLM (Ji et al., 2015)

where a is the activation function. At the output level the context vector c_{t-1} is added and then predicted using

$$y_{t,n} = \text{softmax}(W_h h_{t,n} + W_c c_{t-1} + b) \tag{2.55}$$

2.3.3 Transfer Learning Based Approach

In the classical representation learning algorithms we discussed so far, the learning is based on the in-domain training datasets. For example, if we want to learn representations using different machine learning algorithms for a task or language X, then we need a training dataset for the same task or language. The trained model is expected to perform efficiently for the unseen dataset of the same task or language at the test time. The same procedure needs to be adopted for some other task or language Y to be trained. But this can be challenging when the task has fewer training examples. "Transfer Learning" gives the freedom to train models for low-resource tasks or languages while having semantic or syntactic knowledge of similar tasks or languages (Ruder, 2019). The transfer of knowledge can be observed from the "source" domain or task to "target" domain or task (refer to Figure 2.10) .

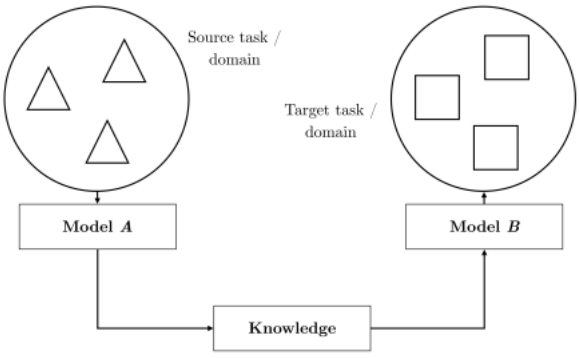

Figure 2.10: The knowledge flow of transfer learning (Ruder, 2019)

Mathematically for the target domain D_T if the input is X_T and output is Y_T, then the transfer learning learns the conditional probability for the target domain task $P(Y_T|X_T)$ having the representation knowledge of the source domain D_S.

Ruder, 2019 has defined the taxonomy for transfer learning which includes *multi-task learning* and *sequential learning*. In our thesis, in Chapter 5, we have used multi-task learning to train our sentence embedding framework. In Chapter 6, our transfer learning process is a sequential transfer learning methodology while injecting morphological knowledge of the pivot language into the cognate detection task.

2.3.3.1 *Multi-task Learning*

Multi-task learning has become very successful in the NLP domain, mainly in machine learning based representation learning (Collobert & Weston, 2008). Individual representation learning techniques (introduced in 2.3) often miss out on the semantic or syntactic relatedness between related tasks or languages. Models trained in a multi-task setup are capable of preserving shared representations between these related languages or tasks, resulting in producing better generalized distributed models (Ruder, 2017). Multi-task learning can be performed in two ways

- **Hard Parameter Sharing:** This is one of the most frequent approaches taken by researchers to train the model. In the case of neural network-based frameworks, the hidden layers are common between different tasks whereas the output layer is task-specific. The benefit of using this architecture is it reduces the chance of overfitting (Baxter, 1997). As multiple tasks use the same hidden layers, the model learns more generic representations while capturing semantic or syntactic relatedness.

- **Soft Parameter Sharing:** In this methodology, each task uses different models with individually initialised parameters. To ensure the parameters similarity, the distance between parameters are controlled and tuned (Duong et al., 2015; Y. Yang & Hospedales, 2017).

Multi-task learning setup involves multiple tasks with parameter sharing and is able to capture the common representations. Thus it can be called efficient feature extractors (Lee et al., 2021). In low-resource NLP tasks where the training dataset is noisy or limited, the traditional deep learning approaches fail to capture relevant features as representations which lead to poor performance. In a multi-task setup due to higher generalisation, the model can identify and give more weight to the relevant features. The introduction of inductive biases projects multi-task learning as an efficient regularizer. It ensures that there is no overfitting while training with limited noisy training sets and reduce the complexity of the model (Søgaard & Goldberg, 2016).

2.3.3.2 *Sequential transfer learning*

Sequential transfer learning is a common transfer learning approach used in different scenarios. In this setting the source and target tasks are different and unlike multi-task learning, the knowledge is shared sequentially from source to target. The goal is to improve the performance of the target task with the knowledge of the source task. Thus D. Wang and Zheng, 2015 have named it *model transfer*. Having the capability of sequential knowledge transfer, this transfer learning approach is quite effective in the case of low-resourced scenarios. T. Q. Nguyen and Chiang, 2017 have described how low-resourced language tasks can be improved using the knowledge of the pre-trained model on a high-resource language. Zoph et al., 2016 highlighted that machine translation for a less-resourced dataset can be improved using the sequential transfer learning methodology. Sequential transfer learning has two phases

- **Pretraining Phase:** In the pretraining phase, the models are capable of capturing the generic properties and the representations of the dataset that might be useful for the different language-specific or domain-specific downstream tasks. The efficient pretraining phase demands a good amount of training data and some amount of supervision (Ruder, 2019). Among them the *self-supervision* techniques produced effective models as we discussed in Section 2.3.2. Another dominant supervision technique is *distant supervised pretraining*. Mintz et al., 2009 proposed the methodology to train a relation extraction model. In social media data analysis, this pretraining strategy is also adopted. Sahni et al., 2017 proposed a model to predict sentiments of the tweets trained on a large twitter corpus. Recently Felbo et al., 2017 has proposed a system to analyse Twitter emoticons trained on a huge Twitter dataset. Later, this model is used to predict the sentiment and sarcasm of the tweets as downstream tasks. These pretraining and experiment strategies highlight the capability of understanding the representation of the source task which can be used as a knowledge transfer to the target tasks. For small language models, distant supervision learning is proven to be effective in capturing the word or sentence structural representations. Ziser and Reichart, 2018 used this strategy for unigram and bigram prediction. When we talk about supervision and pretraining, for many NLP tasks *supervised pretraining* is the common approach to be taken. For example in machine translation (Zoph et al., 2016) or in POS tagging (J. Yang et al., 2017), supervised pretraining on high-resource language and transferring the knowledge to low-resource languages can be seen frequently. Moreover, recently, researchers have trained language-specific models using supervised pretraining on a large dataset to capture semantic representations of the sentences to perform various downstream tasks including natural language inference (Conneau et al., 2017), image captioning (Kiela et al., 2018) and paraphrasing task (Wieting et al., 2016).

- **Adaptation Phase:** After the pretraining phase, the model learns the characteristics of the downstream tasks or languages. The pretrained models are often used as a feature extractor to be run on the tasks-specific dataset. Koehn et al., 2003 have proposed to keep the pretrained model frozen and use the learned weights as the initial features while training for other similar datasets. Recently Miao et al., 2021 designed a *Dual Neural Network Classification for Authorship Verification* where the pretrained *BERT* model is used as a feature extractor. In the small-scale environment, feature extraction has the advantage as it requires less computational memory to perform and an existing model can be reused. On the other hand, by fine-tuning a pretrained model, the weights get updated according to the dataset. Practically during fine-tuning, the weight parameters of the model are considered as a tested initialization phase to be tuned according to needs. While both the adapting processes have their advantages, Peters et al., 2019 argued the effectiveness of these two methods in different scenarios. In this thesis, we adapted the fine-tuning of pretrained model for the downstream tasks. However, the training in this fine-tuning phase is very critical and sensitive to different scenarios. Varying the learning rate up or down has a substantial impact on the performance of the model. Howard and Ruder, 2018a proposed *slanted triangular learning rates* to adapt the parameters of the model according to the task-specific features. Ge et al., 2019 proposed *step decay schedule*, a geometrical learning rate decay scheduler. For our experiments in Chapter 4, we have opted for the step decay scheduling while finetuning the feature layers using a self-learning approach. Another way of controlling the fine-tuning phase is by regularization. Wiese et al., 2017 injected a L2 regularizer between the parameters of the pretrained and the fine-tuning model. We have also opted for and controlled our unsupervised training by regularizing the learning of the model. Our newly introduced loss function (refer to Chapter 4) has a regularization term to control the degree of clustering and finetuning.

2.4 CHAPTER WRAP UP

In this chapter, we have started by defining the definition of "low-resourced languages" and "low-resourced applications" in this thesis. In Section 2.2, we have highlighted the fundamental details of machine learning. We have covered different types of machine learning techniques and the training process. The fundamentals of the training machine learning process consist of loss functions and gradient descent methodology observed in this section. One of the primary unsupervised machine learning methodologies is *Clustering*, covered in Section 2.2.3. From Section 2.2.4 and onward, we have studied the fundamentals of the "neural networks" as well as different types of "deep neural systems". Later in Section 2.3, we talked about "Representation learning in NLP", one of the basics of the thesis. Then we described the existing representation learning approaches from word level to document level either as distributed learning or statistical representation learning. In the next chapter, we will talk about related work for the different tasks investigated in this thesis.

3 RELATED WORK

The existing state-of-the-art cross-lingual sentence embeddings (Hirota et al., 2020; Reimers & Gurevych, 2020) are trained to map contextual embeddings in the shared vector space based on parallel sentences. However, training these models for low-resource languages is a challenging task due to the lack of parallel and annotated training datasets. The existing large language models like *XLM-R-large* (Conneau et al., 2020) have somehow boosted the efficacy of different downstream NLP tasks for a few low-resource languages but that require fine-tuning huge set of 550 parameters, which is a trivial task. Moreover, extending the large language models for diverse low-resource languages requires sufficient training datasets to be collected from different sources. Conneau et al., 2020 highlighted the challenges while collecting datasets for most of the low-resource languages and highlighted the need of identifying these languages using a language identification tool. Thus our decision of designing the efficient multilingual unsupervised deep neural system to capture cross-lingual sentence representations requires deep dive analysis of existing sentence embedding frameworks. It is also important to study the existing language identification tools capable of identifying the languages in a corpus. Recently, Kanojia et al., 2020b in their study, highlighted that injecting cognates in a parallel sentence enhances the efficacy of pre-trained machine translation models by capturing the semantic relativeness between the similar word pairs. In our study, we investigate whether injecting cognates in the training dataset can improve the efficacy of the cross-lingual sentence embedding models. But identifying cognates is a hard task. Researchers have designed different systems to identify cognates which we learn as a part of our literature survey.

In this chapter, we will talk about the related works on language identification (refer 3.1), sentence representation learning (refer 3.2) and cognate detection (refer 3.3).

3.1 LANGUAGE AND DIALECT IDENTIFICATION

Language identification (LI) and dialect identification (DI) for short texts or closely-related languages have been investigated by many researchers using different methodologies. Most of the works are geared towards supervised methodologies. Unsupervised LI is a challenging task and very few works can be found in this area. Jauhiainen et al., 2019 have surveyed different approaches to language identification methodologies.

3.1.1 Language Identification for Short Texts

Hammarström, 2007 has investigated the LI task of short texts as a word classification methodology. He creates a frequency table consisting of each seen word of the training corpus along with the frequencies. The word-to-frequency table is considered the most powerful component for the LI task. The model then performs unsupervised affix extraction detection and this component plays a crucial role in the probabilistic guessing of unseen words. Vatanen et al., 2010 developed a system which considers messages of 5-21 characters long. The system uses a n-gram language model on the dataset created from Universal Declaration

of Human Rights (UDHR) [1]. While these methods considered domain-specific datasets, Konstantopoulos, 2007 investigated language identification for names. The author has created and published a corpus containing names mapped with languages. During the evaluation, they used different name-only language models and general language models to identify the languages of the names. There exist language identification systems capable of identifying languages of single words in a document. The work involves annotating these words with a specific language in a single document. Singh and Gorla, 2007 explored how a multilingual document can be segmented into monolingual documents using language identification on words. Bergsma et al., 2012 explored LI for Twitter data. Their work covers nine languages of Cyrillic, Arabic and Devanagari scripts. They have published a large corpus of tweets annotated with language IDs. Along with the corpus, their experimental results include two LI methodologies. One of the models is built with a discriminative classifier which accepts tweets along with the metadata. The tweet metadata has user names, locations, and URLs for the respective pages. The character features of this model encode the characters of the tweets and metadata. The n-gram character sequence is restricted to four characters. The second system is built with an efficient language model. The methodology is based on the "prediction by partial matching (PPM) family" of algorithms and in this work, the authors have chosen the PPM-A variant (Cleary & Witten, 1984). To be precise, the algorithm determines the number of bits required to encode each character of a string. The only required parameter for the model is the maximal order. Lui and Baldwin, 2014 analysed Twitter data identification systems by their evaluation methodologies. They have produced a language-labelled Twitter dataset for 65 languages. The dataset is constructed in a mostly-automated approach and they provide the algorithm for the dataset creation. They claim that a voting-based ensemble of different language identification systems for Twitter messages outperforms individual systems. King and Abney, 2013 tackled the language identification problem as a word labelling task in a multilingual document. They have applied weakly-supervised methods and solved the task as a sequence labelling problem. They built an annotated corpus having more than $250,000$ words collected from the web. From their experiments, they have concluded that the Conditional Random Fields (CRF) (Lafferty et al., 2001) trained with generalized expectation criteria is the most efficient methodology for the task. D. Nguyen and Doğruöz, 2013 also investigate what the best approach to tag individual words in a document is. They tag the words with the help of a n-gram language model and a dictionary. In the first pass, they try to find a word in the dictionary and if the search is unsuccessfully then the language model conveys the decision. Along with this, they inject the local context of the words to further improve the identification model. Finally, they consider the task as a sequence labelling problem and they used CRF for predictions, but they have only considered the Turkish-Dutch corpus during experiments. Giwa and Davel, 2013 investigated language identification with code-switching. They implemented a Naive Bayes (NB) classifier and a Support Vector Machine (SVM). They also implemented an n-gram smoothing techniques to improve the efficacy of the Naive Bayes classifier. They have tested their approach on 11 South-African languages having 15-300 characters long sentences. Dongen, 2017 in the study performed language identification on English-Dutch code-switching. She has used different machine learning algorithms to perform experiments including SVM, decision tree, and CRF models. She claims that the chain CRF produces the most effective results in comparison to other methods.

These methods are all supervised and this does not allow them to exploit a large amount of unlabelled data and situations where the languages are not known in advance. Unsupervised LI is a challenging task. Next, we will discuss the existing unsupervised language identification methodologies.

1 Can be obtained from https://huggingface.co/datasets/udhr.

3.1.2 Unsupervised Language Identification

Hughes et al., 2006b in their study have highlighted some of the major challenges regarding the language identification task and one of them is about performing language identification on languages having no training data or little training data. Researchers have tried to tackle this problem by conducting unsupervised language identification tasks. Mather, 1998 has approached language identification as a clustering problem. The clusters contain information from the multilingual corpus according to languages. To identify the words that help to discriminate documents, a Singular Value Decomposition (SVD) methodology is used followed by separating the terms into groups. The documents from the same group are then merged. Selamat and Ching, 2008 used Fuzzy ART NNs (Carpenter et al., 1991) for clustering documents written in Arabic, Persian, and Urdu. In Fuzzy ART, they have shown how to update clusters dynamically. Amine et al., 2010 used a character n-gram representation for text. Later *k*-means algorithm followed by particle-swarm optimization is applied to cluster the languages which result in different small clusters. Shiells and Pham, 2010 worked on unsupervised tweet LI with the Chinese Whispers algorithm of Biemann, 2006a and Graclus clustering (Dhillon et al., 2007). The authors found that by applying purity and authority weighting in the cluster assignment, the efficacy of the algorithm increased. Amine et al., 2010 used a character n-gram representation for sentences. In their experiment, they used the combination of $n = (2,3,4,5)$ character sequences using a moving window. They record frequencies of all possible n-grams in a matrix. Later the *k*-means algorithm followed by particle-swarm optimization is applied to cluster the languages which result in different small clusters. C.-C. Lin et al., 2014 performed word-level language identification on code-switched datasets using conditional random field autoencoders. They also used word vectors and word lists on unlabelled data to increase the efficiency of the proposed language identification system. Results show that the injection of the word lists during training helps to capture word similarities. Wan, 2016 proposed a data-driven cluster and label-based approach to identify words in a multilingual diachronic corpus. In their approach, the PPMI (positive point-wise mutual information) matrix is combined with a constant-sized weighted context window to extract features from the text. With the help of Truncated Singular Value Decomposition (TSVD), they have reduced the dimension of the word vectors. Finally, they perform k-means clustering on the word vectors to perform language-based clustering of the words. Rijhwani et al., 2017 researched short-text tweets for unsupervised language detection which mainly focuses on seven languages. They consider word-level language identification to identify code-switching. The system uses the Viterbi algorithm with Hidden Markov Model (HMM) parameters and like previous works, they have considered it as a sequence labelling task. Poulston et al., 2017 used Word2Vec word embeddings and *k*-means clustering to make their LI model. Another approach to performing unsupervised language identification exploits the idea of topic modelling. Voss et al., 2014 used Latent Direchlet Allocation (LDA) methodology to cluster tweets. From their manual observation, the LDA models with 5 topics were the most efficient while separating the languages. They have experimented on Darija, French, and English. W. Zhang et al., 2016a capture n-gram representations of the text as features and perform language identification using Latent Direchlet Allocation. They have used Gibbs sampling model for training. The languages of the topics are finally chosen manually by the authors. During this process, the topics from the same language are merged.

3.1.3 Dialect Identification

There is some research on dialect identification (DI). C.-R. Huang and Lee, 2008 experimented with text classification of a corpus having three varieties of Chinese from Mainland China, Singapore, and Taiwan. Their approach implements the top-bag-of-word similarity metric for making similarity measures of words.

Considering the corpus as bag-of-words, the highest frequent words are chosen to be features of the language based on which they calculate the similarity to do further classification. Zampieri and Gebre, 2012 studied the automatic identification of Portuguese language varieties. In their work, they propose to use both character n-grams and word n-grams to capture linguistic features. The word n-gram captures the lexical differences whereas character n-grams capture the orthographic differences and similarities between language varieties. Later Zampieri et al., 2016 further studied the variants of the Portuguese language from the newspapers printed in Brazil, Macau, and Portugal. They have implemented the SVM classifier to capture representations using unigram and bigram language models. Trieschnigg et al., 2012 compared different language identification methodologies on the Dutch Folktale Database which contains fairy tales written in different Dutch dialects. They claim that the classification accuracy of similar languages having very little training data is low. Zečević and Vujicic-Stankovic, 2013 studied dialect identification using different language identification tools for Serbo-Croatian dialects. The experiments are performed on a collection of Ecavian and Ijekavian documents. They have tested three language identification tools including Langid.py (Lui & Baldwin, 2012), Content Language Detection (CLD)[2] and the classifier developed by Tiedemann and Ljubesic, 2012. They have concluded that it is difficult to identify the Ijekavian variant as no observation tools are trained for it.

Another interesting dialect identification work was carried out for Spanish variants (Zampieri et al., 2013). For feature extraction, the classical characters and word n-gram language models are considered along with the morphological information and parts-of-speech tags. The experiment aimed to understand whether only grammatical information can convey linguistic features for dialect identification. Diwersy et al., 2014 have explained a multivariate approach to the study of a language variation of French. In their experiments, they have applied visualization and semi-supervised examination of the feature space. Some researchers have explored language identification methodologies for Romanian dialect identification. Ciobanu and Dinu, 2016 explained how the orthographic and phonetic features of the words help to design dialect identification systems for Romanian dialects. Using the Needleman-Wunsch alignment algorithm (Needleman & Wunsch, 1970) they have aligned the words followed by extracting features of the words using different n-grams. For classification, they have used logistic regression. Many researchers have performed dialect identification for Arabic dialects. Elfardy and Diab, 2013 proposed a supervised sentence level dialect identification system for Arabic. The sentence-level features are derived from the tokens. With the help of these features, a generative classifier model was trained to predict the dialect for each sentence. O. Zaidan and Callison-Burch, 2014 released a large corpus of Arabic online commentary annotated with different dialects. With different experimental results, they claim that the classifier trained with Arabic dialects outperforms the systems trained on Modern Standard Arabic (MSA). Tillmann et al., 2014 identified Arabic dialects using a sentence-level dialect classifier based on binary feature functions. They have trained a linear classifier linear support-vector machine (linear SVM). Wray, 2018 implemented a multi-class linear kernel support vector machine for Arabic dialect identification. As a feature extractor, the author has used character n-grams ranging from unigram to 5-gram. Along with that, the author has also captured word-level features using unigram to trigram. From the experiments, the author has deduced the classifier performance while having different types of feature extractors.

In Chapter 5 we will discuss our proposed unsupervised sentence representation learning frameworks for low-resourced language and applications. Up until now most researchers have designed supervised sentence embedding frameworks for cross-lingual learning. Very few works exist in unsupervised cross-lingual sentence embedding learning. In Section 3.2, we will discuss the related works for cross-lingual sentence embedding frameworks efficient for downstream cross-lingual NLP tasks.

2 Can be used from https://github.com/mikemccand/chromium-compact-language-detector.

3.2 CROSS-LINGUAL SENTENCE REPRESENTATION

The majority of the current multilingual sentence embedding methods are supervised approaches. There exist some unsupervised sentence embedding frameworks (Pagliardini et al., 2018; Y. Zhang et al., 2020), but they are mostly for English sentence embeddings. Kiros et al., 2015a proposed the SkipThought model trained as an encoder-decoder model to reconstruct the surrounding sentences given a passage. The reconstruction captures the semantics and syntactic properties of the sentences, thus mapping closely into the shared vector space. The encoder-decoder model is constructed using recurrent neural networks. Later FastSent (Hill et al., 2016), a log-linear bag-of-words model, was proposed. The model is trained to predict the adjacent sentences given the source unlabelled sentences. The model was trained on the Toronto Books Corpus dataset. Kenter et al., 2016 proposed Siamese C-BOW, where they average word embeddings out to construct sentence embeddings. The model has used a Siamese architecture to predict surrounding sentences during training. Y. Zhang et al., 2020 has proposed the Sent2Vec model which allows the extraction of sentence embeddings using word vectors along with n-gram embeddings. The model is inspired by the bilinear models and matrix factor models. Due to simplicity in the design of the model, it requires less computational power and memory than other deep learning models. The model is trained on the Toronto book corpus, Wikipedia sentences and tweets. Recently Y. Zhang et al., 2020 produced sentence embedding using mutual information maximization. To capture global sentence representation, the sentence embeddings are extracted using convolutional neural network (CNN) layers on top of the BERT language model. While extracting sentence representations, mean-over-time pooling is used. Their introduction of self-supervised training maximises the mutual information (MI) between all local context embeddings and the global sentence embedding. J. Huang et al., 2021 proposed WhiteningBERT, where final sentence embeddings are produced by normalizing initial sentence embeddings with whitening. Extensive experiments on different benchmark evaluation datasets highlight the efficacy of the strategy over others.

When we talk about cross-lingual sentence embeddings, existing models are mostly supervised sentence embedding models. Initial methods generated sentence embeddings based on a neural machine translation system with a shared encoder. España-Bonet et al., 2017 studied the efficacy of the sentence embeddings generated using the encoders of the neural machine translation (NMT) model. They have trained the system with four language pairs German– English, French–English, Spanish–English and Spanish–French. The encoder-decoder model is designed using a recurrent neural network with attention mechanisms. Extensive experiments on semantic textual similarity (STS) tasks and neural machine translation task showcase the efficacy of the multilingual sentence embeddings generated from encoders. Schwenk and Douze, 2017 learned multilingual sentence representations of six languages using a neural machine translation model. They claim the model can capture the underlying semantics of the sentences irrespective of language as the sentences are the same in meaning. In their work, they have also introduced a sentence similarity mechanism to compare about 1.4 million cross-lingual sentences and studied the characteristics of the closely-related sentences. Schwenk, 2018 learned joint multilingual sentence embeddings for parallel sentence mining in a large news collection. The bilingual sentences are chosen based on the distance between the source and target sentences in the shared embedding space. Manually a threshold value is considered to decide the minimal distant bilingual sentences. The model was trained on nine languages of the Europarl corpus with almost 2 million sentences each. Subsequently, the LASER (Artetxe & Schwenk, 2019; Schwenk et al., 2019) framework considered a sequence-to-sequence architecture using bidirectional LSTM networks with a shared BPE vocabulary, and was trained on parallel corpora designed for neural machine translation across 93 languages. They aim to enhance the efficiency of low-resource languages by joint learning of many languages in a neural translation setup. During training, the semantically similar sentences across languages

are close in shared multilingual vector space. Their model was trained on the parallel dataset collected from the Europarl, United Nations, OpenSubtitles2018, Global Voices, Tanzil and Tatoeba corpora.

Expanding beyond translation-based approaches, researchers have designed sentence embedding systems without considering encoder-decoder-based machine translation strategy. The maximization of dot products between source and target language parallel sentences was studied by Guo et al., 2018 using a dual encoder architecture. During the training, they injected hard negatives by considering sentence pairs which are not a direct translation of each other but carry some degree of semantic similarity. The sentence embeddings are obtained by averaging the word and bi-gram level embeddings learned during backpropagation. The model was trained on two language pairs English-French (en-fr) and English-Spanish (en-es). Experimental results highlight the performance improvement of this model over the previous cross-lingual translation-based sentence embedding models. Y. Yang et al., 2019 generated sentence embeddings for the bitext retrieval task using a bidirectional dual encoder with additive margin softmax. Like the previous work, they have injected hard negatives as a translation ranking problem along with the true translation pairs in the batch-wise training. Their introduction of margin efficiently captures the separation between translations and nearby non-translations. The system was trained on English-French, English-Spanish, English-German, English-Russian and English-Chinese language pairs with nearly 400 million sentences. Chidambaram et al., 2019 proposed Multilingual Universal Sentence Encoder (mUSE), a dual-encoder model trained on large web-mined translation parallel corpora, along with data from Reddit, Wikipedia, and Stanford Natural Language Inference (SNLI) (Bowman et al., 2015) to learn more context, supporting 16 languages. A translation ranking task was used to identify a correct translation pair, and the architecture assumes 5 hard negative pairs for each sample while training. Y. Yang et al., 2020 trained multilingual sentence embedding model for semantic retrieval on three different tasks: multi-feature question-answer prediction, translation ranking, and natural language inference (NLI). In their experiments they have used two models to extract sentence embeddings, one is the transformer architecture and another one is a convolutional neural network (CNN). The use of a CNN architecture enhances the inference in terms of computation and time but reduced the accuracy. In both cases, the sentence embeddings are obtained by averaging the token embeddings. The model supports 16 languages from different language families. Recently, Z. Yang et al., 2020 proposed Conditional Masked Language Modeling (CMLM) to generate sentence embeddings, by co-training the system with bi-text retrieval and Natural Language Inference (NLI) tasks. Their integration of sentence representations into masked language model training conditions on sentence level representations while considering the adjacent sentences. For evaluation, they have also proposed a new multilingual evaluation benchmark corpora called XEVAL that consists of translated sentences of the English SentEval corpus. The translation was in-house and collected using the Google-translate API. From their experimental results, they concluded that models trained with multilingual natural language inference datasets have increased the efficiency for zero-shot multilingual classification tasks. To generate sentence embeddings beyond the naïve CLS token and simple pooling strategies of language models, sentence transformer architectures were proposed (Reimers & Gurevych, 2019). For multilingual S-BERT models, Reimers and Gurevych, 2020 utilized the teacher-student model where the student model is tuned with a parallel corpus from 50 languages based on knowledge transfer from a fine-tuned teacher model developed by Reimers and Gurevych, 2019. During training, the student model tries to mimic the teacher model, trained on the English natural language inference (NLI) dataset. The methodology is easy to be trained and needs less computational power and memory during training. The authors also showcased in cross-lingual space, that the student model is capable of identifying the semantic relatedness between languages. Thus, this methodology is very efficient for semantic textual similarity understanding between two language pairs. LaBSE (Feng et al., 2020) was designed on the BERT architecture and trained on 6 billion sentence pairs by use of additive margin softmax

loss with in-batch negative sampling. The model is trained on 109 languages and produces state-of-the-art results for bitext mining for most of the high-resourced languages across language families. To train the model they have used monolingual as well as bilingual datasets. The monolingual dataset is collected from CommonCrawl and Wikipedia whereas the bilingual datasets are constructed using translation pairs collected from the web.

Another trend of research for sentence embedding involves improving the alignment of contextual embeddings into a shared vector space using iterative self-supervised learning or tuning with synthetic parallel corpora. Hirota et al., 2020 introduced the Enhancing Multilingual Sentence Embeddings (EMU) framework which tries to semantically enhance pre-trained multilingual sentence embeddings. That is, instead of building sentence embeddings from scratch, EMU fine-tunes pre-trained multilingual sentence embeddings with two major components: enhancement of semantic similarity and multilingualism of sentence embeddings using multilingual adversarial training. Cao et al., 2020 used a parallel corpus as an anchor to align representations in a multilingual language model. The degree of alignment is based on the contextual semantic similarities of the words. While word alignment captures the word semantics, the sentence embeddings are then computed by concatenating the average, maximum and minimum pooling of the word vectors to understand sentence semantics. Recently, Kvapilíková et al., 2020 proposed an unsupervised method for improving pre-trained cross-lingual context vectors using synthetic parallel sentences and extracted sentence embeddings via mean pooling. However, the use of machine translation to generate synthetic parallel data for such methods fails to generalize to low-resourced languages for which translations might be erroneous.

While we now have discussed existing sentence embedding frameworks, in Chapter 6 we propose a state-of-the-art cognate detection framework for low-resourced languages. We also highlight the importance of cognates to train cross-lingual sentence embedding frameworks. While we mostly talk about the unsupervised cognate detection framework, researchers have identified some neural and traditional methodologies to identify cognates. In Section 3.3, we will discuss related work on cognate detection.

3.3 AUTOMATIC COGNATE DETECTION

Algorithms for automatic cognate detection (ACD) are mostly based on phonetic or orthographic similarity measures and often language-dependent or supervised approaches. Covington, 1996 developed an algorithm to align words for historical comparison by their phonetic similarity. The algorithm is designed to consider surface forms instead of phonetic rules and sound laws. During alignment, the aim is to minimize the distance between similar words. The author designed a "no-alternating-skips rule" which restricts the algorithm from skipping the strings during one-to-one matching. A penalty score is assigned to every skip or match to find the best possible alignment. The best match gets 0.0 penalty score whereas a total mismatch gets 1.0 penalty score. The experiments were performed on 82 cognate pairs in various languages. Kondrak, 2000 released a novel algorithm for aligning phonetically similar sequences. The methodology is a combination of different state-of-the-art sequence comparison methodologies. It assigns a score on the phonetic similarities of the words based on multivalued features. It records the operations of the insertions and deletions while comparing the individual phones. A rule-based distance metric was established to understand the similarity between the phones. Similarity-based algorithms follow these works to identify cognates between a language pair of different families. The orthographic similarity-based works calculate distances or string similarities between word pairs and define similarity scores to identify cognates. Melamed, 1999 proposed an algorithm for bitext mapping and alignment. In their implementation, the orthographic similarity matching threshold is the longest common subsequence ratio (LCSR). The ratio is calculated using the longest

common subsequence between a cognate pair and the length of the longer token. The method is implemented on cognate pairs of three languages French-English, Spanish-English, and Korean-English. Some researchers have also taken parallel datasets into account to identify cognates. Distance measurement-based scores have become the feature set to identify cognates in these cases. Tiedemann, 1999 has investigated whether capturing the similarity of bilingual corpora boosts the efficacy of cognate identification. A similarity matrix was applied to every possible word pair of the list. For this study, the author has extracted word pairs from aligned technical sentences collected from the PLUG corpus. Similar to bitext mapping and alignment (Melamed, 1999), the author relied on extracting the longest common subsequence followed by statistical scores to perform string matching. Mann and Yarowsky, 2001 studied methods to build a lexicon between languages using cognates. During the experiments, they identified bilingual dictionaries to select cognates. They have proposed an induction algorithm based on string edit distance to learn similarities between cognates. The distances below the threshold value are considered cognates. to calculate the distances, the authors have chosen three functions: Levenshtein distance, a stochastic transducer-based cost function, and a hidden Markov model-based function. From their experiments, they have concluded that neither of these functions can capture the correct cognate pairs having surface form relationships but these functions are efficient to capture the similarity between cognate pairs having the same orthographic or phonological similarity.

Later, Mulloni and Pekar, 2006 designed a method that induces rules capturing regularities of orthographic mutations of words to define cognate pairs. These orthographic mutations of words can be seen as a part of the structural change when migrating from one language to the other. The algorithm has two major operations: **(i)** it tracks the edit operations between two cognate pairs and then calculates the normalized edit distance (NED) between each pair which assigns a score to each cognate pair, **(ii)** it extracts candidate rules based on the edit operations and assigns statistical scores to the rules. The candidates having scored more than a threshold value, are considered cognate pairs. The algorithm was tested on English-German cognate pairs. Rama, 2016 has released a convolutions-based model which also takes phonetic similarity-based scores into account to detect cognates for word pairs. The framework considers words as two-dimensional matrices and is trained using a Siamese network architecture. The deep model understands the phoneme level feature representations and language relatedness from the cross-lingual word pairs in a supervised setup. The CNN network is also able to capture the non-monotonic shared features of two words during backpropagation. The authors also investigate whether the CNN network can capture the patterns of sound change from underlying phonetic representations to be used for cognate identification. Along with phonetic representations, the language relatedness between the cognate pairs are also injected. The architecture was evaluated on Austronesian, Indo-European, and Mayan language families. Kanojia et al., 2019a performed a cognate detection task on Indian languages, but this included a large amount of manual intervention during identification. To perform cognate identification, a dataset was collected from publicly available IndoWordnet [3] (Bhattacharyya, 2010). Manually, cross-lingual word alignment was performed using concept ID. Using string similarity methods and XDice (Brew et al., 1996), the initial cognates were detected. To collect more cognates, orthographic similarity measurements between strings (alignment of substrings) with an SVM classifier were used. During training, they also injected the phonetic features of the words.

While we are talking about identifying cognates using string-based similarity, Kanojia et al., 2019b introduced a character sequence-based recurrent neural network for identifying cognates between Indian language pairs. Like the previous work, they have taken word pairs from IndoWordnet and experimented to identify four types of cognates: true cognates, false cognates, false friends and non-cognates. Along with

3 Can be obtained from https://www.cfilt.iitb.ac.in/indowordnet/.

the Wordnet dataset, they have extracted words from parallel datasets for better word alignment. As a part of their framework, first similarity scores between word pairs are documented using orthographic similarity-based methods. As a part of the experiments, three different similarity measures were used: normalized edit distance (NED) method, cosine similarity, and Jaro-Winkler similarity (JWS). Obtained similarity scores are treated as labels for a cognate detection recurrent neural network model. The last sequence output of the recurrent neural network considered passes through the softmax layer for the classification.

The influence of classical machine learning and dynamic programming-based approaches defines automatic cognate detection tasks as semi-supervised approaches. Hauer and Kondrak, 2011 trained a linear SVM based on word similarity and language-pair features to detect cognates. The word similarity features are calculated using edit distance, longest common prefix length, commonality in bigrams, and commonality in length difference between short and long words. As capturing the word feature only demonstrate the local context, the injection of language features highlights the commonality between language pairs. During training, the model learns the weights of language-pair features highlighting the language relatedness along with the word features. Finally, the cognate groups are clustered using average similarity between clusters. The methodology was tested on the Indo-European, Austro-Asiatic, Hmong-Mien, Mixe-Zoque, Sino-Tibetan, and Tai-Kadai cognate datasets. Ciobanu and Dinu, 2015 implemented SVM on the orthographic similarity-based word features. Inspired by the previous research works, the words are first aligned using the string alignment algorithm. From the aligned words, characters n-grams are extracted as features and used as input to the SVM classifier to identify cognates. Phonetic alignment-based SVM models performed quite efficiently on different language families while detecting cognates. Jäger et al., 2017 proposed classification of cognates using phonetic alignment of words using an SVM classifier. The SVM primarily operated on string similarity measures. As a feature input, auxiliary features of the string pairs obtained from string similarity are used. The inferred similarities are then used as inputs to perform unsupervised clustering.

Another thread of research for cognate detection recently involved multilingual contextual knowledge injection methods. Merlo and Rodriguez, 2019b explored the effect of cross-lingual features on bilingual lexicon building, which was later implemented in Indian language cognate detection methods. They study whether cross-lingual word embedding space can learn feature representation like human bilingual behaviour. The evaluation was done on two cross-lingual word embeddings VECMAP (Artetxe et al., 2018) and M2VEC (H. Wang et al., 2019). The experimental evaluation highlights the capability of the cross-lingual word vectors to identify the cognates efficiently because of their semantic knowledge in the shared vector space. Recently, Kanojia et al., 2020b injected cross-lingual semantic features for cognate detection tasks using newly trained language models, which showed better results than state-of-the-art methods. Extending the work, Kanojia et al., 2021 proposed incorporating gaze features in the context-aware cognate detection task, which improved results for Hindi-Marathi language pairs. These methods may produce better results; however, the main disadvantages lie in the training of these models as they are data-hungry models. Thus, incorporating the manually annotated gaze features and newly generated cross-lingual contextual embeddings becomes an impossible solution for many under-resourced languages.

3.4 CHAPTER WRAP UP

In this chapter, we provided a detailed overview of the related work. We first reviewed the language and dialect identification systems. Moving to the next section, we gave a detailed overview of the existing multilingual sentence embedding frameworks. Lastly, we have talked about the existing models for the cognate

detection task. Although our brief survey showed that there exist sentence representation frameworks which capture contextual and structural information, the experimental results highlight the lower efficacy of these models for low-resource languages and applications than for high-resource languages and applications due to the absence of high-quality training datasets. Most of the language and dialect identification frameworks rely on word-level information. In this thesis, we aim to capture better sentence representations of the different low-resourced languages and applications. During language identification, existing researchers have relied on word-level orthographic and phonetic information whereas, in our thesis, we capture the global sentence representations of each of the languages by understanding the linguistic features on the character level. While we learn the global sentence representations, we then focus on capturing the contextual sentence representations. Our approach differs from the existing contextual sentence representation learning as we introduce unsupervised multilingual sentence embedding learning to eliminate the need for parallel corpora during training. Also, we considered the word semantics during training which helps to preserve the relative semantic distance between the input sentences. Thus, our proposed model outperformed the other existing works for the downstream NLP tasks for low-resource language and applications. In the end, we aim to improve the sentence embedding frameworks for closely-related languages by injecting the cognate information during training. This work requires cognates to be correctly identified. The existing systems mostly rely on efficient cognate-pair annotations during training which confines the applicability of these systems to a small number of languages. The majority of the systems are designed to capture string similarities using orthographic or phonetic statures of the words. Some of them have used classical machine learning classifiers for cognate detection. Recently, researchers explored the possibility of designing contextual word embeddings-based classifiers to detect cognates. Most of them have worked in Indian languages. In our work, we propose an unsupervised cognate detection framework to eliminate the need for cognate-pair annotations. We capture the grammatical and structural word representations of a language and transfer that knowledge to the closely-related languages of the same language family during the cognate detection task. Our approach of injecting the word level knowledge helps to identify the linguistic similarity between closely-related language pairs.

4 LANGUAGE AND DIALECT IDENTIFICATION FOR CODE-MIXED DOCUMENTS

4.1 INTRODUCTION

In the previous chapter, we have discussed about the background of different existing deep neural models for NLP applications and literature related to deep neural architectures as well as representation learning for document understanding. In this chapter, we introduce a novel unsupervised deep language and dialect identification framework, which can exploit very minimal training sets of closely related under-resourced languages or dialects and identify texts from the same languages.

Data collection is the primary step in training distributed deep neural architectures for any under-resourced languages. The sources of the datasets are primarily social networking sites or crawled web pages. Most often these datasets are inter-sentential code-mixed datasets. Inter-sentential code-mixed documents contain sentences from different languages (Barman et al., 2014). For example :

- *Abalandeli bembangi yakhe enkulu uMathandeni Manqele kepha bayakuphikisa lokhu ->* [Zulu]
- *Aditif kadaharan nyaéta bahan kimia tambahan nu biasa digunakeun dina industri kadaharan ->* [Sundanese]

For most under-resourced languages, it is very hard to find a well-defined corpus which forces us to rely on datasets created by volunteers. The collected datasets can contain either user comments from movie sites in their native languages or stories (Chakravarthi et al., 2020b). These documents are more prone to contain sentences in different dialects. In the absence of linguistic knowledge, it is often very hard to identify sentences for closely-related languages or dialects.

This identifies automatic language identification (LI) and dialect identification (DI) as a crucial part of natural language processing (NLP) pipelines and is part of tasks such as language modelling, categorization and analysis of these code-mixed datasets. Many researchers have carried out experiments in these domains both in speech and texts starting from House and Neuburg, 1977. Unsupervised LI is an under-explored area, but is highly useful as it can exploit the large amount of unlabelled data and, more importantly, can be employed when the languages to be identified are not known in advance. However, unsupervised LI for short texts is a very difficult task, for which performance is still comparatively poor. W. Zhang et al., 2016b have explored unsupervised language identification but this does not work well with short texts for closely related languages. O. F. Zaidan and Callison-Burch, 2014 have worked on the identification of Arabic dialects. Ciobanu et al., 2018 studied German dialect identification. These works are mostly supervised works. The task is even harder when it comes to unsupervised DI.

In this chapter we will answer our first research question:

- **RQ1:** Is an unsupervised deep sentence embedding framework capable of identifying languages as accurately as supervised language identification models for code-mixed under-resourced and closely-related languages?

While designing novel deep neural architecture to answer RQ1, we encountered another sub-research question which opens up the broader aspect of designing a novel language clustering technique for different under-resourced languages:

- **RQ1.1:** What is the best way to construct sentence embeddings and how to cluster them efficiently for short texts when there is no labelled training data?

In this chapter, we have answered both the research questions by designing a novel deep learning framework for unsupervised language and dialect identification. Our framework is designed by taking inspiration from iterative clustering (W. Zhang et al., 2016b) and self-attention-based sentence embeddings from Z. Lin et al., 2017. Both of the papers have shown good results on unsupervised clustering and sentence embeddings for a single language dataset whereas our goal is to design a system which can work on different short texts of closely related languages and understand every distinct feature. Our system takes different n-gram character features of a sentence and computes attention weights based on their importance in sentence construction. We then fine-tune sentence embeddings with an iterative clustering process with the help of the Stochastic Gradient Descent (SGD) and backpropagation. We call the approach Unsupervised Deep Language and Dialect Identification (UDLDI). One of the main challenges of the model is to get sentence embeddings. As we do not have any labels here, we cannot use typical loss functions to update character-to-sentence embeddings. Further, as the datasets are small short text datasets, there is much less information to learn features to construct efficient sentence embeddings. To overcome this challenge we propose a loss function which considers the probability distribution of assigning sentences in each cluster and tries to maximize the unique sentence assignments in each cluster. These sentence embeddings define the initial cluster centre assignments, which we then fine-tune with the iterative clustering learning process. Our method is tested against three different datasets consisting of under-resourced and closely related languages, the language families and/or varieties in question being South African, Dravidian and Swiss German. The South African dataset consists of languages from different language families, some of which are similar. We have achieved good results on each dataset and the results are better than baseline apporaches. We have also compared our sentence-embedding model to supervised LI and DI methods which outperformed other state-of-the-art models. The model is robust as it does not require any manual intervention and it works well for short texts in any dataset consisting of closely related languages, for any language family.

In a nutshell, our contributions are:

- The aforementioned UDLDI model for language and dialect identification in short texts of closely related languages,
- Efficient sentence embedding construction for short texts,
- Efficient clustering of closely related languages with a very small dataset,
- Designing an probabilistic unsupervised loss function to produce efficient sentence embeddings.

4.2 OUR APPROACH

Unsupervised Deep Language and Dialect Identification (UDLDI) clusters the short texts of closely related languages in an efficient way using character-to-sentence embeddings with self-attention, based on an unsupervised loss function and an iterative clustering method (refer to Figure 4.1). The model uses an unsupervised loss function to build the character-to-sentence embeddings. The iterative clustering process fine-tunes the sentence embedding and enhances the cluster assignment.

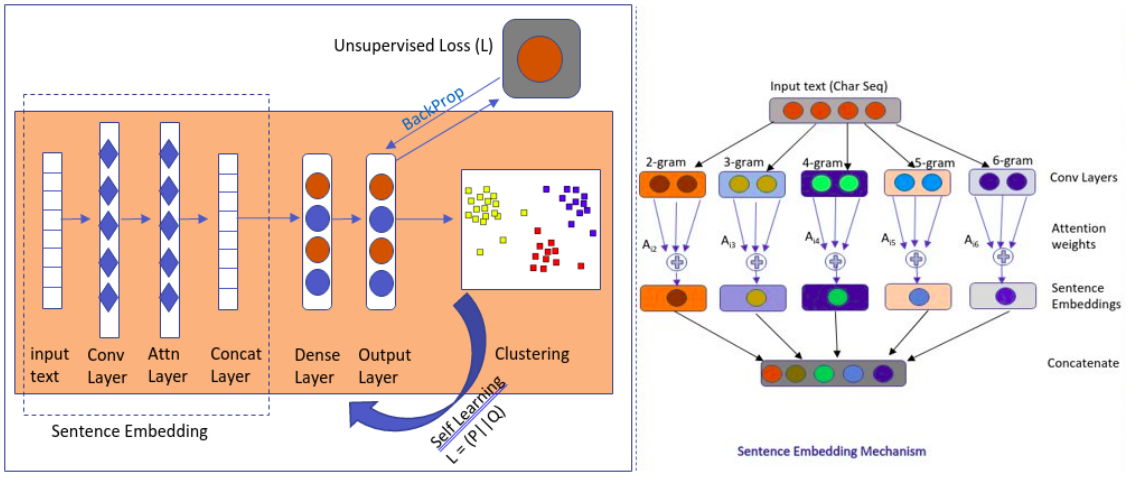

Figure 4.1: Unsupervised deep LI and DI model. The left-hand side shows the architecture design of the UDLDI mechanism; the sentence embedding mechanism is explained on the right-hand side

4.2.1 Sentence Embedding

The sentence-embedding model has two parts. The first one is an n-gram character-level CNN and the second one is a self-attention mechanism, which gives the weighted summed features of sentences based on n-gram ($n \in \{2,3,4,5,6\}$) characters. The attention weights are then multiplied with the feature vectors of the CNN to get the sentence embedding. The different character-level features are achieved with the help of 1-dimensional CNN (X. Zhang et al., 2015). The sentence embeddings are then fed into a dense layer to get the language level probability distribution of the sentence.

The 1-dimensional CNN accepts an input sequence as a character sequence. If a sentence has m characters then the input sequence would be $S = [w_1, w_2, ..., w_m]$, where w_i is a character in the sentence (including spaces). The one-dimensional convolution implements 1-dimensional filters which slide over the sentences as a feature extractor. Let the filters have a shape of $1 \times k$ where k is the filter size. Filter sizes are the same with different n-grams. Let $x_i \in \mathbb{R}^d$ denote the d-dimensional character embedding of the i-th character considering the character vocabulary size is n. For each position j in the sentence, we have a window vector w_j with k consecutive character vectors, as denoted in Equation 4.1.

$$w_j = [x_j, x_{j+1}, ..., x_{j+k-1}] \tag{4.1}$$

The 1-dimensional k-sized filters slide over the window vector w_j to create the feature map s where $s \in \mathbb{R}^{m-k+1}$ and m is the input size. Each element s_j of the feature map for window vector w_j is produced according to Equation 4.2,

$$s_j = a(w_j \cdot m + b_j) \tag{4.2}$$

where vector m is a filter for convolution operation, b_j is the bias for the j-th position and a is the non-linear function. The new feature representation F after rearranging will be represented as $F = [s_1, s_2, ..., s_n]$, where n is the number of convolution filters, s_i is the feature map generated with i-th filter and $F \in \mathbb{R}^{(m-k+1)\times n}$.

To understand the different n-gram feature representation of the sentences, multiple k-sized filters are used which is represented as F_k where $k \in \{2,3,4,5,6\}$.

We aim to get sentence embeddings from different n-gram representations. We have achieved this by getting the weighted summation of T elements of new feature representation F. We perform this for each F_k

separately. To achieve this, we use a self-attention mechanism. The attention mechanism takes the feature representation as input and gives *a* as an output weight vector, as shown in Equation 4.3,

$$a = \text{softmax}(\tanh(W_h \cdot F^T + b_h)) \tag{4.3}$$

where W_h is the weight matrix and b_h is the bias for the attention mechanism. The use of softmax is to ensure that the summation of attention weights $(a_1, a_2, .., a_T)$ are non-negative and sum up to 1. Then we sum up the elements of feature representation *F* according to the weight vectors provided by *a* to get a vector representation *r* of the input sentence by Equation 4.4,

$$r = \sum_{i=1}^{T} a_i \cdot F_i \tag{4.4}$$

where a_i is the attention weights for each element of the feature representation, and $\cdot$ is the element-wise product between elements.

Different n-gram representations carry different weighted sentence embeddings. To capture all the representations, the final representation is a concatenated vector $r = [r_2, r_3, r_4, r_5, r_6]$ (where n-grams $\in \{2,3,4,5,6\}$), which is then passed to the fully connected layer to get the probability distribution of there being one language or dialect.

The sentence embeddings are used in two phases. The first phase is the pre-training phase which can be obtained as described in section 4.2.2; the second phase is the fine-tuning of sentence embeddings based on self-training with the clustering algorithm, described in section 4.2.4.

4.2.2 Dense Layer and Loss function

The sentence embeddings $r \in \mathbb{R}^{N \times M}$ are passed to a fully connected (FC) layer which gives the output $z \in \mathbb{R}^{N \times K}$ and then a softmax layer ($p \in \mathbb{R}^{N \times K}$) returning the probability distribution of all classes, as per Equation 4.5,

$$p_{ij} = \frac{exp(z_{ij})}{\sum_{t=1}^{K} exp(z_{ik})} \tag{4.5}$$

where *K* is the number of languages or dialects and p_{ij} gives the probability that sample *i* belongs to class *j* ($j \in \{1,2,...,K\}$).

Maximum Likelihood Clustering Loss (MLC Loss): The loss function maximizes the same feature sentence assignment to one language. It can be considered as maximizing cluster purity based on the probability distribution. The loss function can be computed by means of Equation 4.6,

$$L_u = \sum_{i=1}^{N} \max_{j=1}^{i} p_{ij} - \max_{i=1}^{N} \sum_{j=1}^{i} p_{ij}^2 \tag{4.6}$$

where *N* is the batch size.

4.2.3 Intuition Behind MLC Loss

In contrast to existing loss functions, our loss function does not depend on entropy-based calculation as there is no label. Instead the loss function is trying to maximize the probability distribution function based on the feature assignments on each class (or cluster) and minimize the maximum summation of the squared probability distribution of all the languages assigned to one class. The maximization of the probability

function ensures the best possible assignment of the features in each individual class. But in this process, there is a chance of biased feature assignments (for the datasets where features of data points are very near in distance) to a single class. To overcome this trivial situation, the model is penalised with the maximum summation of the feature assignments over all samples. This stabilizes the system by assigning features to different classes in the best possible way.

After training the character embeddings and sentence embedding we cluster the dataset. J. Xie et al., 2016 have shown in the Deep Embedding for Clustering (DEC) architecture how to use clustering as self-training (Nigam & Ghani, 2000) and how to fine-tune the mapping and do the clustering. The DEC framework uses the idea of parameterized non-linear mapping from the data space X to a lower-dimensional feature space Z to optimize a clustering objective. It uses stochastic gradient descent (SGD) via backpropagation on the clustering objective to learn the mapping, which is parameterized by a deep neural network. In our case, we have made use of this iterative clustering process to fine-tune the sentence embeddings; this methodology improves the clustering.

4.2.4 Self Training and Clustering

The initialization of this phase starts by obtaining initial cluster centroids $u_{j}{}_{j=1}^{k}$ by applying the cluster algorithm to the sentence embeddings from the pre-training phase, where k is the set of cluster centres. The self-training phase has the following two steps: (i) the soft assignment between sentence embedding and initial cluster centroids, and (ii) fine-tuning sentence embeddings and cluster centroids using the auxiliary target distribution.

As a first step, we calculate the soft assignment between the cluster centroid u_j and sentence embedding z_i based on Student's t-distribution (Maaten & Hinton, 2008) with a single degree of freedom ($\alpha = 1$), as per Equation 4.7,

$$q_{ij} = \frac{\left(1 + \left\|z_i - u_j\right\|^2\right)^{-1}}{\sum_{j'} \left(1 + \left\|z_i - u_{j'}\right\|^2\right)^{-1}} \tag{4.7}$$

where $z_i \in \{Z\}$ and q_{ij} can be considered as the probability of assigning sample i to cluster j.

The second step involves refining the cluster learning from their high confidence assignments with the help of an auxiliary target distribution (J. Xie et al., 2016). The auxiliary target distribution (p_{ij}) helps to improve cluster purity and puts more emphasis on data points assigned with high confidence. The p_{ij} can be calculated by means of Equation 4.8,

$$p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_{j'} \left(q_{ij}^2 / \sum_i q_{ij'}\right)} \tag{4.8}$$

where $\sum_i q_{ij}$ is the frequency of the soft clusters.

The training loss is then designed based on KL divergence loss between the soft assignments q_i and the auxiliary distribution p_i, as shown in Equation 4.9.

$$L = KL\left(P \| Q\right) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{4.9}$$

4.3 EXPERIMENTAL SETUP

4.3.1 Data

We have taken three different datasets for the training and testing of our model, for the following languages and dialects.

South African languages. This dataset was used by Duvenhage et al., 2017 and consists of short texts for 11 South African languages, many of which are related; these are the Nguni languages (zul, xho, nbl, ssw), Afrikaans (afr) and English (en), the Sotho languages (nso, sot, tsn), tshiVenda (ven) and Xitsonga (tso). The texts are on average 15-20 characters long. We have taken 15,000 samples for training and 5,000 for testing.

Dravidian languages are under-resourced languages spoken mainly in the southern part of India. This dataset contains the languages Tamil, Telugu, Malayalam and Kannada (Chakravarthi et al., 2018). They have different orthographies with a unique script for each language (Chakravarthi et al., 2019a). The training set contains a total of around 14,000 instances and the test set contains a total of 3,000 instances.

Swiss German. This dataset is based on the ArchiMob corpus (Samardžić et al., 2016) and is used for dialect identification. It contains transcriptions of 34 interviews with native speakers of various German dialects spoken in Switzerland. The subset used for German Dialect Identification contains 18 interviews (14 for training and 4 for testing) from four Swiss German dialect areas, i.e., Basel, Bern, Lucerne, and Zurich, with each interview being transcribed using the 'Schwyzertutschi Dialäktschrift' writing system (Dieth, 1986). The training set contains a total of around 14,000 instances and the test set contains a total of 3,638 instances.

Details of the distribution of the data across three different datasets can be found in 4.1

Data-Type	South-African Languages	Dravidian Languages	Swiss-German Dialect
Train	15000	14000	14000
Test	5000	3000	3638

Table 4.1: Data distribution of three different datasets

4.3.2 Evaluation metrics

We have used the standard cluster performance evaluation process. The number of clusters is set to the same number as the original number of classes in the data and evaluated with unsupervised cluster accuracy, as shown in Equation 4.10,

$$ACC = max_m \frac{\sum_{i=1}^{n} 1(l_i = m(c_i))}{n} \tag{4.10}$$

where l_i is the ground truth label, c_i is the cluster assignment produced by the algorithm and m ranges over all possible one-to-one mappings between clusters and labels.

The number of clusters is also an evaluation process in case of unsupervised clustering. The accuracy is determined with the same number of clusters as the ground truth data. But in general, the number of clusters also needs to be optimised and evaluated. It is evaluated by Normalized Mutual Information (NMI). The NMI is useful for comparing the clustering results with different cluster numbers. It is shown in Equation 4.11,

$$NMI(l, c) = \frac{I(l, c)}{\sqrt{H(l)H(c)}} \tag{4.11}$$

where l is the ground truth label and c is the predicted cluster. I is the mutual information and H is entropy. When data is partitioned perfectly, the NMI score is 1, and when l and c are independent, it becomes 0.

4.4 RESULTS AND DISCUSSIONS

We have compared our results with various baselines, both for supervised and unsupervised setup, which are provided in Table 4.2 and Table 4.3.

Model	South African Dataset	Dravidian Dataset	Swiss German Dataset
AE + Kmeans	19.04%	51.00%	14.00%
Fasttext +Kmeans	21.00%	67.94%	20.00%
LDA (W. Zhang et al., 2016b)	21.56%	69.34%	19.00%
Skip-Thought (Kiros et al., 2015b) + Kmeans	24.67%	71.00%	20.35%
AE + DEC (J. Xie et al., 2016)	25.00%	74.97%	22.35%
LSTM (Z. Lin et al., 2017) + DEC	31.67%	81.50%	26.00%
UDLDI	**37.00%**	**82.00%**	**30.00%**

Table 4.2: Accuracy of different models for unsupervised LI and DI

Unsupervised setup. The model is compared with one of the most well known unsupervised clustering methods Auto-encoder based k-means clustering. The sentences are projected in a latent space Z from the feature space X and then clustered it applying k-means. Our second baseline model is fasttext[1] word embeddings and k-means. The sentence embedding is the average of word embeddings. We have also compared our experimental results with the LSTM sentence-embedding model (Z. Lin et al., 2017) along with the DEC framework (Table 4.2). Our model performed substantially better than other state-of-the-art models, achieving 37% accuracy (16.82% improvement upon the LSTM model) for the South African dataset. It has also achieved 82% accuracy (.6% improvement upon the LSTM model) in the Dravidian dataset. For the Swiss German dataset, 30% accuracy (15.83% improvement upon the LSTM model) was achieved; the latter DI task can be considered one of the toughest tasks to perform, as the dialects are very similar. It means that the model is well capable of identifying important character n-gram features in sentence construction. Details about the attention weights distribution are discussed in the section 4.4.1.

Supervised setup. Table 4.3 shows different baseline models including previously published systems in the Swiss German DI shared tasks. Our model has outperformed all the state-of-the-art methods achieving 99%, 99.34% and 81% accuracy for the South African dataset, Dravidian dataset and Swiss German dataset, respectively. Though it has outperformed both the LI and DI method, significant efficiency can be observed in DI which has outperformed the highest performing system (Zampieri et al., 2019) by 8%.

4.4.1 Discussion and Analysis

To analyze sentence embeddings we will refer to the Swiss German DI which can be considered the hardest task. The dialects are very closely related from four different parts of the country —Basel (BS), Bern (BE), Lucerne (LU), and Zurich (ZH). Some of the sentence constructions are very similar to each other. For example we can take two sentences of Bern and Lucerne which are "*aber dää ìsch emaal dä*"

1 https://fasttext.cc/docs/en/unsupervised-tutorial.html

Model	South African Dataset	Dravidian Dataset	Swiss German Dataset
BRNN (Kocmi & Bojar, 2017)	89.00%	93.50%	—
NB+Lex (Duvenhage et al., 2017)	96.00%	97.34%	—-
CLUZH (Clematide & Makarov, 2017)	—	—	67.00%
MAZA (Malmasi & Zampieri, 2017)	—	—	68.00%
tearsofjoy (Zampieri et al., 2019)	—	—	75.00%
LSTM (Z. Lin et al., 2017)	96.67%	98.51%	68.67%
CharCNN (X. Zhang et al., 2015)	98.00%	96.56%	66.09%
UDLDI	**99.01%**	**99.34%**	**81.00%**

Table 4.3: Accuracy of different models for supervised LI and DI

and "*aber dää ìsch seer schträng ggsìì mit öis*" respectively. In the sentence constructions, we can see the first three words are exactly the same consisting of the same characters even though they represent two different dialects. Without extensive morphological and linguistic knowledge, it is very difficult to separate sentences spoken by people from these regions. Character n-grams represent distinct features of sentences of a particular region. Our model is also able to identify dialectal (pronunciation) variants for an inflected form of a verb. We can see this in Figure 4.2. There are two sub-columns under the "Text" column which consist of different sentences formed with the same verb in different contexts. The red colour is the importance degree of the attention weight and light blue is a less important part. The English word "have" has different forms in different dialects. For example, in case of BE, it is written as "hei" whereas in LU it is written as "hend".

Text		Language
Text1	Text2	
und mir hei ja pro bateljoon nume zwei ilkaa ghaa	wasmer do zeersch überchoo hei weissi nìd söimer	BE
und mir hend glik ghaa isch immer es basler batelioon isch choo	oder die hend det unde	LU
und mir häi also s familielääbe	oder woo wo de wo ebe de alarm gsìì ìsch häi ìsch häi mir öisi	BS
und mir händ asoo seer mager müse dure	unfäll ghaa händ und züüg und sache soll me das fiire	ZH

Figure 4.2: Attention visualization of dialect-specific words pointed out by the model. Text1 and Text2 are randomly chosen sentences from the dataset

The verb form is used in different contexts in different sentences. In the first example in the Text1 column, a sentence is shown with words "mir hei" which means "we have". But the "mir" word is the same in different dialects. Our model has given less attention weight to this word whereas the highest attention weight is given to "hei". The character n-gram considered space as well. So in the Text2 column, though the sentence does not contain the word "mir", depending on the "hei" word with space, the sentence is classified as BE. Here the word "söimer" is also given the highest attention weight as the word only appeared in BE. The same can be observed in other sentences as well for other dialects. Conjunctions like "und" ("and") are very common in sentences and are often written the same. As shown in Figure 4.2, this word occurs both in BS and ZH, and as such has very low significance in terms of dialect identification. Consequently, our model has assigned it a low attention weight.

As Scherrer et al., 2019 have pointed out, the ArchiMob corpus shows variation on different levels. Not only are different lexical items pronounced and hence written differently across dialects, the data additionally incorporates intra-speaker variability (i.e., slight variation in the pronunciation of the same word by the

same speaker) and even shows transcriber-related variation. These factors may overestimate the distinctive features as identified by the model. Moreover, the attention weights in our model may point to yet another type of variability, namely, idiosyncratic features of certain speakers such as the repetition of words (idiolect). In Figure 4.3 dialects are compared with each other in pairs. The word "ùùnd" (standard German "und") in the second example is common in both LU and BS. But in the case of LU, the word followed by "ùnd" whereas in case of BS it followed by other words, so the character attention distribution gives the highest attention weights to the n-gram "ùnd ùn". Though the word "ùnd" (like "und") has very low significance in dialect identification as an individual word, in the Swiss German dataset this n-gram is a distinctive feature, although probably being more indicative of idiolect than dialect. The n-gram "im va" carries very little distinctiveness in case of LU, so it has the lowest attention score.

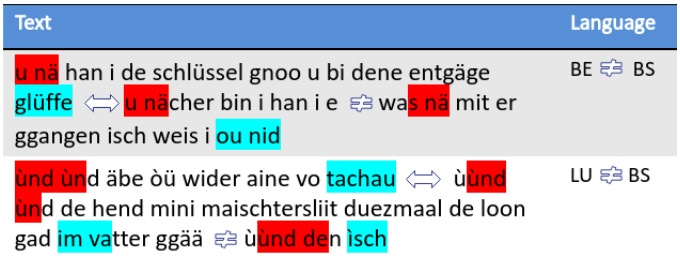

Figure 4.3: The attention visualization for the Swiss German dataset

Text	Languages
a de gränze gsii	ZH
a de gränze sige	BE
aber es isch denn scho bau aus	BE
aber es isch s isch	LU

Figure 4.4: Error analysis for the Swiss German dataset

There are possibilities of very small differences in sentence construction for DI or LI. Our model efficiently captures these differences. In Figure 4.3 the character n-gram "u nä" exists in BE. Even in word "nächer", the first two letters are "nä", which leads to the n-gram "u nä". In case of BS, the construction is different and the n-gram "u nä" does not exist. Thus in the case of BE, the highest attention weight is assigned to "u nä" which makes the sentence embedding more robust and efficient.

4.4.2 Error analysis in the model

The UDLDI works efficiently in comparison to other state-of-the-art methods. However, there are some challenges. As can be seen in Figure 4.4, the sentence construction "*a de gränze*" can be found both in Zurich and Bern region. Though the sentences consist of other words, our methods are unable to differentiate between sentence-level features as the character n-grams are also common in different dialects; for example, the word "gsii" also exists in the Lucerne region. This leads to an insufficient attention weights distribution for common character n-grams. The same is true for the word sequence "*aber es isch*". These lead to biased clustering. This observation is supported by the NMI score of the clustering.

Figure 4.5 shows that the NMI is highest when the number of clusters is 3 (blue line). We suspect that in the case of overlapping word sequences with a non-distinctive right context, the model is wrongly classifying sentences as belonging to the same cluster (dialect). We can compare this with the Dravidian

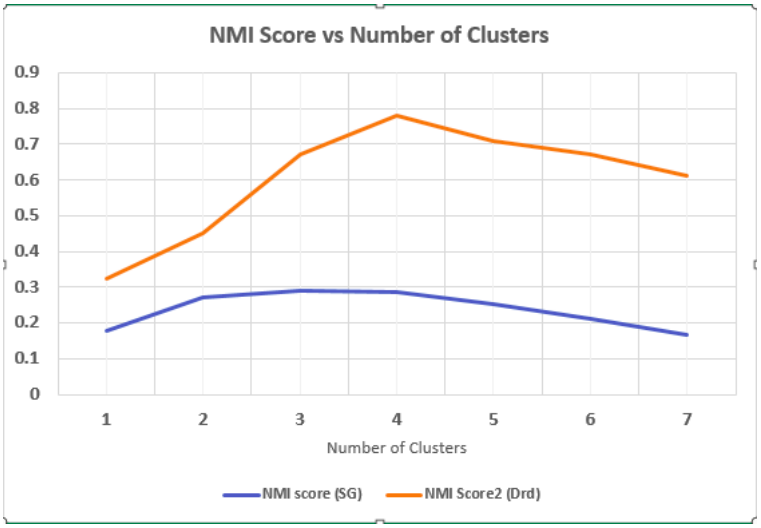

Figure 4.5: NMI Score vs Number of Clusters graph

dataset. The graph shows that the NMI score is highest for 4 clusters (red line), which is equal to the number of languages present in the dataset. Though the Dravidian languages are very similar, they have different orthographies with a unique written script for each language. Thus the model can easily identify the script, directly captured by linguistic features on the character level. The somewhat unstable characterisation of the number of clusters as shown in the NMI graph for Swiss German dataset may to some extent be caused not only by linguistic features transcending one dialect area, but also, as mentioned in 4.4.1, by intra-speaker variation, transcriber-related variation and idiolect.

4.5 CHAPTER WRAP-UP

In this chapter, we propose a novel character attention based unsupervised deep language and dialect identification (UDLDI) model for short texts of closely related languages. The research questions that were raised in the early part of the chapter can be concluded on a positive note. Our proposed sentence embedding technique is able to capture local and global character level contexts in a sentence. Moreover, it is able to identify the importance character sequences which makes a sentence distinguishable for different languages. The proposed maximum likelihood clustering loss function is able to maximize the probability distribution function based on the feature assignments on each class (or cluster) and minimize the maximum summation of the squared probability distribution of all the languages assigned to one class. The maximization of the probability function ensures the best possible assignment of the features in each individual classes. Moreover, the regularization function stabilizes the system by assigning features to different classes in the best possible way.

We have performed our experiments on three different language families and outperformed other state-of-the-art models both as a supervised and an unsupervised model. We have also achieved promising results for dialect identification which is considered to be one of the hardest tasks in NLP. Our experiments show that UDLDI is capable of doing language or even dialect identification irrespective of language families. The results in Section 4.4.1 show that the model is able to correctly identify some of the verbs which are distinguishable in different dialects. We have also shown that, based on NMI, our unsupervised model is correctly predicting the number of languages that are present in a dataset, opening up interesting applica-

tions to large-scale multilingual dataset processing. Our future work will be the improvement of clustering efficiency for closely related languages and dialects. Though the model has outperformed other state-of-the-art models, there is a lot of scope to improve the unsupervised learning model. In this model, for example, co-occurrence of words in different contexts has not yet been considered. In the future, we will aim to capture word contexts along with n-grams in sentence embeddings.

5 SENTENCE REPRESENTATION LEARNING

5.1 INTRODUCTION

In the previous chapter, we have discussed a novel methodology of identifying languages and dialects in a big inter-sentential code-mixed corpus in an unsupervised way. In this chapter, we will discuss how to efficiently design unsupervised sentence embedding frameworks which can be trained on unlabelled corpora after performing language identification. We have showed our frameworks efficiently understands sentence representations, achieving state-of-the-art results on different downstream NLP tasks for low-resourced languages and low-resourced NLP applications.

Documents consist of sentences which convey the linguistic aspects of the languages. Understanding sentence representation is a key aspect in performing different natural language understanding tasks like sentiment analysis, summarizing, sentence inference etc. Multilingual sentence representation understanding is achieved by encoding sentences from different languages in a shared vector space based on their semantic and syntactic similarity, called multilingual sentence embedding. Sentences with similar meanings are very close to each other in the shared vector space. With the abundance of data in diverse languages, cross-lingual sentence embeddings enable the mapping of multilingual texts into a single unified vector space for a wide range of Natural Language Processing (NLP) tasks (refer to Figure 5.1). In multilingual settings, to train an efficient sentence embedding model, a large amount of training data is required.

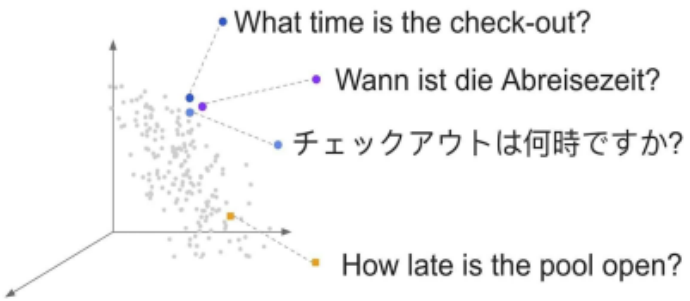

Figure 5.1: Multilingual sentence embedding

While working with low-resourced languages or low-resource natural language applications, there is a big drop in the amount of training datasets which emphasize the need for unsupervised or self-supervised representation learning frameworks. Researchers represented sentences with traditional TF-IDF vectors on top of bag-of-words (Arroyo-Fernández et al., 2019). In this process sentences are represented with the vocabulary sized vectors and each of these vectors convey information about the importance of each word in a corpus. But this method does not consider the structural or semantic information of the sentence, resulting in producing less efficient sentence embeddings. It also makes the vector very sparse because the vector size is equivalent to the big vocabulary size of a document. To capture semantic information, researchers

have introduced novel deep learning architectures for sentence representation learning including capturing sequential information with LSTM models (Palangi et al., 2016) using mostly word vectors like the GloVe embeddings (Pennington et al., 2014) or the fastText embeddings (Bojanowski et al., 2017). But producing these static word vectors are difficult, as to train these models a large corpus is required which is difficult for many low-resource languages. Moreover, these sequential supervised sentence embedding methods require a good amount of training data to produce efficient results on NLP tasks. Recently large language model based multilingual sentence embedding methods (Feng et al., 2022; Reimers & Gurevych, 2020) have been proposed but these methods do not produce efficient results in low-resourced scenarios. We focus on bridging this gap by designing novel unsupervised sentence embedding frameworks to capture sentence representations in low-resourced scenarios.

In this chapter we aim to answer our second research question:

- **RQ2.** Does an unsupervised deep sentence embedding framework generate efficient sentence embeddings in cross-lingual domains without the use of parallel corpora for downstream natural language tasks? Does unsupervised projection of efficient word representations improve sentence embeddings in cross-lingual domains?

Current sentence embedding methods are predominantly monolingual systems, geared mainly towards English (Conneau et al., 2017; X. Yin et al., 2020). While there exist multilingual sentence embedding frameworks, they are mostly supervised methods requiring a large parallel corpus for training. For under-resourced languages, there is not sufficient training data to effectively learn a model and we show that our unsupervised approach can better exploit the available unsupervised data, and thus produce better results for under-resourced languages. This is achieved by using a dual-encoder architecture based on word-level semantic similarity score (via Word Mover's Distance) and learning to embed this into a single vector for sentences. Supervised sentence embedding approaches map parallel sentences from source and target languages into the same vector space by either maximising their cosine similarity or minimising the distance between the generated embeddings (Artetxe & Schwenk, 2019; Reimers & Gurevych, 2020). For example, recent supervised methods using a parallel corpus rely on a *teacher-student model* to minimize cross-lingual embedding distance (Reimers & Gurevych, 2020) or an additive margin softmax function based dual sentence encoder to maximally separate the sentences that are true translations from similar overlapping sentences (Y. Yang et al., 2019). Although such methods produced good results, the use of these loss functions in unsupervised settings fails to efficiently capture cross-lingual semantic similarities across sentences. The state-of-the-art unsupervised approach relies on automated machine translation to generate a "pseudo parallel corpus" (Kvapilíková et al., 2020) for training. This method is affected by the presence of translation errors and fails to generalize to low-resource languages for which translations are not available. This highlights our first part of the research question:

- **RQ2.1.** Does an unsupervised deep sentence embedding framework generate efficient sentence embeddings in cross-lingual domains without the use of parallel corpora for downstream natural language tasks?

To answer, we propose *DuEAM*, a cross-lingual sentence embedding framework based on a novel *dual encoder architecture with an unsupervised joint loss function* using an *anchor-learner* approach, a variant of the teacher-student model. We also depict the performance of a *weakly-supervised variant* of our unsupervised *DuEAM* architecture (obtained by simply changing the training dataset). The weakly-supervised framework to learn semantic relationship between cross-lingual sentences is motivated by the existence of the multilingual natural language inference dataset (XNLI) (Conneau et al., 2018), and the possible creation

of such a dataset from existing comparable corpora. The unsupervised *DuEAM* framework learns from randomly chosen sentence pairs from the XNLI dataset. Thus, we overcome the need for parallel sentences for multilingual sentence embedding generation. To understand the degree of similarity between monolingual and multilingual sentence pairs during training, the anchor module uses the Word Mover's Distance (WMD) (Kusner et al., 2015) (used as a scalar value during backpropagation), while the learner module is trained to generate sentence embeddings. Thus, we learn a low-dimensional embedding of the sentences from the more complex encoding generated by WMD. We show that our joint loss formulation effectively captures cross-lingual semantic similarity between sentences by preserving distances between points across languages.

Extensive experiments on multilingual sentence similarity and parallel sentence mining tasks have showcased the efficacy of our sentence embedding framework. For example, on the cross-lingual STS benchmark (Reimers & Gurevych, 2020) (Sentence Similarity Benchmark), our unsupervised approach achieves state-of-the-art average Spearman rank correlation score of **62.1**, comparable to the supervised sentence embedding approach of LASER (Artetxe & Schwenk, 2019) with an average of 65.8. In fact, for certain languages, our models are even seen to outperform LASER (e.g., for EN-DE our unsupervised model achieves a Spearman rank correlation score of 64.6 and weakly-supervised model achieves a Spearman rank correlation score of 69.4 compared to 64.2 for LASER). On the BUCC task (bitext mining task) (Zweigenbaum et al., 2017) our model achieves a better F1 score compared to the existing unsupervised model of Kvapilíková et al., 2020. Interestingly, for certain under-resourced languages, we outperform both LASER and multilingual S-BERT (Reimers & Gurevych, 2020) by an average of **10**% on the Tatoeba benchmark (cross-lingual parallel sentence matching benchmark). We also show better or comparable performance to LASER even on monolingual classification benchmark tasks. Thus, our model is robust across diverse language families for multilingual sentence embeddings. Details of our results are described in the experiment section later.

To this end, we propose:

- *DuEAM*, a novel dual encoder based on an anchor-learner architecture for unsupervised and weakly-supervised multilingual sentence embedding generation,
- A joint loss function coupling word mover's distance and cosine similarity to capture the degree of text similarity and relatedness between sentence pairs,
- Experimental evaluations on monolingual as well as several cross-lingual benchmark tasks depict that our model effectively captures semantic similarity across languages and provides state-of-the-art unsupervised performance, comparable with supervised models,
- Robustness in zero-shot transfer learning for low-resource languages across language families, outperforming state-of-the-art supervised approaches on sentence matching tasks in certain scenarios.

Though the *DuEAM* methodology has achieved state-of-the-art performance when it comes to multilingual sentence alignment including parallel sentence matching, parallel bitext mining and sentence similarity matching for low-resourced languages, there is a gap when it comes down to the multilingual classification tasks for low-resource NLP applications both for low-resource and high-resource languages. Sentence or short text classification plays a crucial role in a plethora of modern NLU and information retrieval applications, such as automated chatbots and personal digital assistants (L. Chen et al., 2021), email intent identification (W. Wang et al., 2019), business intelligence (Qazi et al., 2014), user feedback analytics (Salinca, 2016), and domain-specific question answering (Sánchez-Pi et al., 2013). Understanding of structured as well as unstructured text has become pivotal in driving organizational efficiency and augmented assis-

tance for improving user experience and quality-of-service (QoS) (Navigli, 2018) across diverse domains and applications.

The presence of domain-specific terminologies has necessitated generic NLU methodologies to be further fine-tuned to the specific use-cases. Thus, *domain adaptation* of pre-trained models, known as *generative pre-training* (Howard & Ruder, 2018b), has been extensively studied. In this context Transformer (Vaswani et al., 2017b) based language models are trained on a huge unsupervised text corpus and the *whole architecture* is fine-tuned on specific down-streaming applications like classification task in different languages. Multi-lingual language models like mBERT (Pires et al., 2019) and XLM-R (Conneau et al., 2020) allow effective cross-lingual knowledge transfer. *XLM-R-large* currently provides state-of-the-art results on several benchmark tasks, but involves an architecture with a staggering 550 million parameters.

Adapting such large multi-lingual language models to classification tasks requires appropriate fine-tuning.The current trend of fine-tuning bigger models raises several concerns and challenges – more so in enterprise settings:

- First, fine-tuning of such large models is extremely resource intensive with demanding memory requirements (Sanh et al., 2019a). Moreover, precise parameter setting (like learning rate) is essential for convergence and in zero-shot setups may result in over-fitting as shown in Casanueva et al., 2020,
- Second, re-training such large models with the emergence of new data poses a significant effort and,
- Third, the requirement of sufficient annotated training data to properly tune such models further complicates their deployment in the production pipeline. This also confines the implementation of these large models for low-resourced NLP languages and applications.

Sentence embedding architectures providing fixed-length dense vectors, have been introduced as "generic" text representation to address unsupervised learning use cases, like semantic search, and have been shown to perform relatively well on Semantic Textual Similarity (STS) tasks. Existing sentence embedding frameworks (Artetxe & Schwenk, 2019; Chidambaram et al., 2019; Pagliardini et al., 2018; Reimers & Gurevych, 2020) produce efficient multi-lingual sentence embeddings. For example, SBERT (Reimers & Gurevych, 2020) uses *knowledge distillation* to train a smaller student model from a larger teacher model, while LASER (Artetxe & Schwenk, 2019) relies on training an encoder-decoder on parallel data in multiple languages. Our proposed *DuEAM* framework maps cross-lingual sentences in a *anchor-learner* setup while considering the word level contexts between source and target sentences from the anchor.

It is suggested that sentence embeddings are not particularly suited for transfer learning for classification tasks (Reimers & Gurevych, 2020) [1]. The absence of model fine-tuning (termed as "frozen") of generic sentence embedding architecture, leads to significant performance loss compared to the fine-tuned language models.

This motivates the following research question

- **RQ2.2.** Does unsupervised projection of efficient word representations improve sentence embeddings in cross-lingual domains?

To answer this, we propose the *Meta Embedding with Up-Projection and Fine-Tuning* (MUFIN) framework to learn enhanced sentence representation geared towards text classification applications, predominant in industrial settings. We showcase that *MUFIN* enables better understanding of cross-lingual textual semantic similarity by leveraging the knowledge of both the large language models and sentence embeddings.

1 EMU (Hirota et al., 2020) sentence embedding was proposed for such semantic specialisation tasks, but the performance was not observed to be comparable to that of the language models.

We show that our framework is practical and light-weight with faster and robust training, lesser memory footprint, along with competitive performance to large language models. To the best of our knowledge, this is the first work that *successfully bridges the classification performance gap* between sentence embeddings and language models.

To this end, we propose:

- MUFIN – an effective and light-weight framework to extract *task-specific multi-lingual sentence embeddings* for enhanced textual semantic understanding,
- A novel methodology based on the combination of *unsupervised up-projection*, *sentence embedding alignment*, and *meta-embeddings* – capturing the knowledge of large language models and sentence embeddings within a single representation,
- Experimental results showcasing comparable performance of MUFIN to large language models, albeit with lower training time and lesser memory requirement – amenable to deployment even in smaller enterprises.

We will describe the details of the architectures and experiments in two parts:

- **Unsupervised Sentence Embedding Framework:** In this part we will introduce our novel work on building an unsupervised sentence embedding framework trained on unlabelled corpora mitigating the need of parallel corpora for multilingual sentence embedding framework training
- **Enhancing Semantic Understanding of Sentence Embedding Frameworks:** In this part we will explain our work on unsupervised projection of this sentence embedding framework for better semantic understanding of downstream natural language understanding tasks.

5.2 UNSUPERVISED SENTENCE EMBEDDING FRAMEWORK

5.2.1 Our Approach

In this section, we describe the components and working of the proposed *Dual Encoder with Anchor Model* (DuEAM) architecture for multilingual sentence embeddings, trained using an unsupervised multi-task joint loss function.

5.2.1.1 *Dual Encoder with Anchor-Learner*

Figure 5.2 depicts the *dual-encoder* based anchor module, where the same sentence pair is fed as inputs into both the anchor and the learner components. Note that these sentence pairs are not considered to be parallel translations and can even be from either the same language or different languages. Such architectures are well suited to capture semantically and contextually similar sentences and map them close to each other in a shared vector space.

We use word-level semantic knowledge from pre-trained multilingual language models in the anchor module to gauge the semantic similarity between the source (s_i) and target (t_i) sentences. Subsequently, embeddings for s_i and t_i are generated by the learner such that their vector space distances reflect their degree of semantic relatedness. Inspired by MoverScore measure (Zhao et al., 2019), using the pre-trained multilingual word-embeddings, the anchor module computes the semantic similarity between source and target sentences by use of *Word Mover's Distance* (WMD) (Kusner et al., 2015).

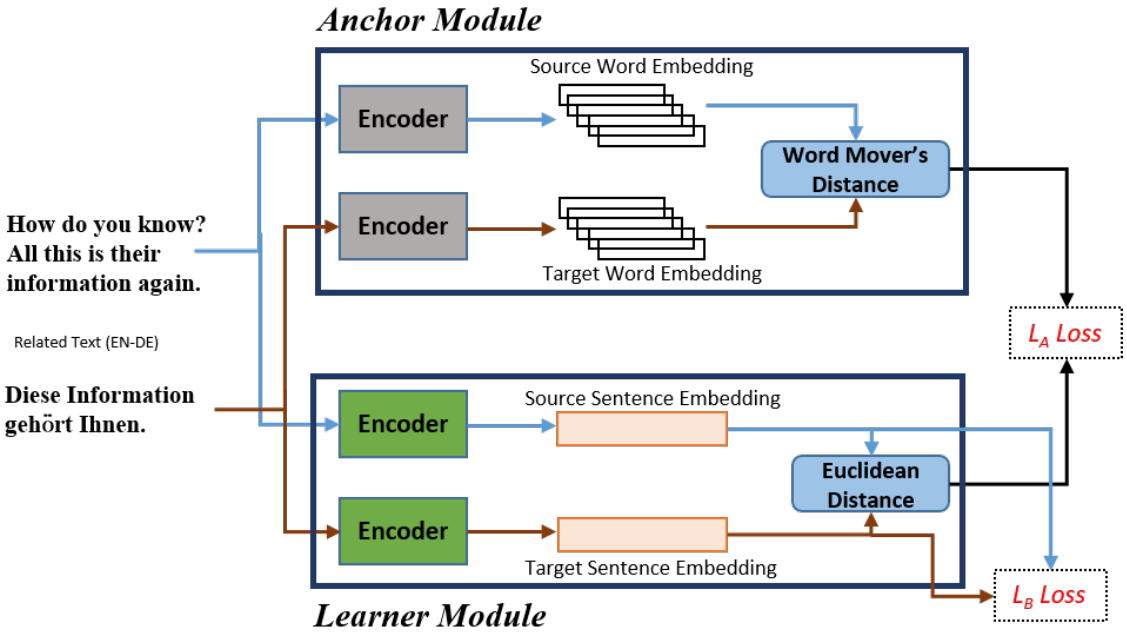

Figure 5.2: *DuEAM*: Proposed Dual Encoder based Anchor-Learner model with multi-task learning. For training, in our anchor module as well as our learner model, we use the XLM-RoBERTa-base (XLM-R-base) language model (Conneau et al., 2020).

The learner module is then trained to generate source and target sentence embeddings such that their Euclidean distance closely approximates the WMD obtained from the anchor. We force the system to consider the knowledge of our anchor system about the semantic relationships of the sentences at the word-level, which also helps to stabilize the training process as the pre-trained anchor model is fixed and the WMD score is considered as a scalar. Thus, during training, we generate embeddings to *minimize the semantic loss*, $\mathcal{L}_{\mathcal{A}}$ as:

$$\mathcal{L}_{\mathcal{A}} = \frac{1}{N}\sum_{i=1}^{N} \exp^{|\exp^{-d_{euc}(s'_i,t'_i)} - \exp^{-d_{wmd}(s_i,t_i)}|} \tag{5.1}$$

where $d_{wmd}(s_i,t_i) = WMD(s_i,t_i)$ is the Word Mover's Distance between the input source and target sentences s_i and t_i, while $d_{euc}(s'_i,t'_i) = \sqrt{\sum_j \left(s'_{i_j} - t'_{i_j}\right)^2}$ is the Euclidean distance between the generated embeddings s'_i and t'_i (by the learner) corresponding to the source and target sentences respectively. The use of our WMD based semantic loss factor enables *DuEAM* to capture a more compact representation between the sentence embeddings, better capturing cross-lingual semantic relationships in the shared vector space.

5.2.1.2 *Dual Encoder with Translation Mining*

While mapping semantically similar sentences close to each other in the shared vector space, an effective embedding framework should also address the translation ranking problem to efficiently map correct translations of source-target sentence pairs within a compact zone of the vector space. Y. Yang et al., 2019 addressed this problem by introducing hard negative sentence pairs along with parallel data during the training process. Since *DuEAM* is an unsupervised approach, we introduce *translation mining based loss*, $\mathcal{L}_{\mathcal{B}}$, using the cosine similarity score between source and target sentences, to handle translation mining, as:

$$\mathcal{L}_{\mathcal{B}} = \frac{1}{N}\sum_{i=1}^{N} \text{cossim}(s'_i, t'_i) \tag{5.2}$$

5.2.1.3 *Multi-Task Dual Encoder Learning*

To bring both loss functions under the same umbrella, we construct a *multi-task learning setup* where we minimize Eq. 5.1 while maximizing Eq. 5.2. Hence, to efficiently generate sentence embeddings, the final multi-task loss function for training *DuEAM* is given by: $\boxed{\textit{minimize } \mathcal{L} = \mathcal{L}_{\mathcal{A}} - \lambda\mathcal{L}_{\mathcal{B}}}$, where λ is the weight parameter.

This multi-task joint learning enables *DuEAM* to effectively capture both cross-lingual semantic similarity and text translation ranking relationship.

Overall, our *unsupervised loss function* aims to learn sentence embeddings such that the Euclidean distance between them are proportional to the semantic distance obtained from WMD, thereby providing a low-dimensional embedding from the more complex word-level similarity space (using Eq. (5.1)). Additionally, Eq. (5.2) enables our framework to align sentence embeddings in the cosine space for translation understanding.

5.2.2 Intuitions behind Loss Function

In *DuEAM*, WMD is computed between the contextual token embeddings (of the pair of input sentences) obtained from anchor encoders, without any stopword removal (as standard while using language models (Conneau et al., 2020)). While multi-lingual language models capture different languages in a common space, WMD captures the contextualized semantic distance between the multi-lingual input sentences. Our use of WMD is motivated by MoverScore (Zhao et al., 2019). Specifically, the use of WMD and cosine in the loss function of *DuEAM* is based on the following intuitions:

- The learner module is trained to generate sentence embeddings such that the Euclidian distance between the learnt sentence embeddings closely approximates the WMD (calculated using token embeddings by anchor) between sentence pairs. This tends to preserve the "relative semantic distance" between the input sentences, enabling *DuEAM* to capture the "semantic relation at word level" within the sentence embeddings obtained.

- Existing methods using parallel sentences for training effectively teach the architecture to learn similar embeddings for similar context – however, the distance in the embedding space between dissimilar sentences are not considered. By using WMD, *DuEAM* generates closer representations for similar sentences, while at the same time forcing dissimilar sentences to have embeddings that are apart in the embedding space. This provides better semantic understanding for improved performance in downstream tasks, as observed in our experiments. For example, the WMD between the German sentence "Sie ist keine Lehrerin" ('She is not a teacher') and the English sentence "She is a teacher" is more than that between the German sentence "Sie ist eine Lehrerin" and the English sentence that is a direct translation, "She is a teacher". Thus, the embeddings are different, and *DuEAM* is able to capture negation and other semantic information for better performance.

- The cosine loss enables *DuEAM* to align the learnt sentence embeddings in the cosine space, based on the cosine similarity between the source and target train sentence pairs, to address the translation ranking problem. The weight parameter λ in the final multi-task loss function further controls its effect.

5.2.3 Training Dataset

Following Chidambaram et al., 2019, we train our *DuEAM* architecture on the natural language inference dataset – using only the XNLI dataset (Conneau et al., 2018) on 13 languages, without any parallel corpora [2]. We do not consider the entailment-contradiction labels in XNLI during training.

Unsupervised Data. To create the training dataset for unsupervised training of *DuEAM*, we randomly pick sentences from the premises and hypothesis of XNLI dataset and form the input sentence pairs. Hence, this random shuffling along with the absence of sentence pair labels provides no supervision during the training procedure.

Weakly Supervised Data. This training data contains both monolingual and cross-lingual sentence pairs, where the monolingual sentence pairs are the same as those of XNLI (without annotated labels). To create cross-lingual sentence pairs, we keep the premises from the source language and replace the hypothesis with target language hypothesis sentences, and vice-versa. Note that this dataset does not contain any parallel cross-lingual sentences, but contains semantically related monolingual and cross-lingual sentences – providing weak supervision. Example can be shown in table 5.1, for language pair EN-DE, the premise is taken from English while the hypothesis is from German and vice-versa. We do not consider any labels to train our model.

Our model supports 53 languages

Premise	**Hypothesis**	**Type**
How do you know? All this is their information again.	This information belongs to them.	*Monolingual (EN-EN)*
- woher weißt du das ? All das sind ihre Informationen.	Diese Information gehört Ihnen.	*Monolingual (DE-DE)*
How do you know? All this is their information again.	Diese Information gehört Ihnen.	*Cross-lingual (EN-DE)*
- woher weißt du das ? All das sind ihre Informationen.	This information belongs to them.	*Cross-lingual (DE-EN)*

Table 5.1: Weakly-supervised training dataset example from XLNI for English-German.

Validation (using accuracy of finding parallel sentences) is done on held out 1K parallel sentences (across languages pairs) from the TED2020 corpus. The *DuEAM* models trained on the above unsupervised and weakly supervised datasets are henceforth denoted as $\text{DuEAM}_{\text{unsupv}}$ and $\text{DuEAM}_{\text{wklysupv}}$ respectively.

Our model support 53 languages from diverse language families. The supported languages and their language codes are given in Table 5.2.

5.2.4 Experimental Evaluation

We evaluate the performance of our proposed *DuEAM* framework on the following 3 benchmark tasks: (a) **STS:** monolingual and cross-lingual *semantic textual similarity*; (b) **BUCC:** *bitext mining* to extract parallel sentences; and (c) **Tatoeba:** cross-lingual *parallel sentence matching*.

5.2.4.1 *Baseline Models*

We have compared the performance of *DuEAM* on benchmark datasets with multiple supervised and unsupervised baseline models which are as follows:

2 Trained on EN, BG, DE, EL, ES, FR, HI, RU, TH, TR, UR, VI, ZH

Language Code	Language	Language Code	Language	Language Code	Language	Language Code	Language
bul	Bulgarian	dan	Danish	mkd	Macedonian	epo	Esperanto
cmn	Mandarin Chinese	est	Estonian	mon	Mongolian	eus	Basque
deu	German	fin	Finnish	nob	Norwegian Bokmål	gla	Scottish Gaelic
ell	Greek	glg	Galician	pes	Persian	isl	Icelandic
fra	French	heb	Hebrew	por	Portuguese	jav	Javanese
hin	Hindi	hrv	Croatian	ron	Romanian	khm	Khmer
rus	Russian	hun	Hungarian	slk	Slovak	lat	Latin
spa	Spanish	hye	Armenian	slv	Slovenian	swg	Swabian
tur	Turkish	ita	Italian	sqi	Albanian	swh	Swahili
tha	Thai	jpn	Japanese	srp	Serbian	uzb	Uzbek
urd	Urdu	kat	Georgian	pol	Polish	war	Waray
vie	Vietnamese	kor	Korean	ind	Indonesian	xho	Xhosa
ara	Arabic	lit	Lithuanian	bre	Breton	yid	Yiddish
cat	Catalan	lvs	Latvian	ceb	Cebuano		
ces	Czech	mar	Marathi	cym	Welsh		

Table 5.2: List of supported languages and their codes.

- **mBERT / XLM-R mean:** We use publicly available mBERT and XLM-RoBERTa (XLM-R) language models trained on large datasets (no parallel sentences considered during pre-training phase). We consider mean pooling of the output layer as the sentence embedding.
- **mUSE:** Multilingual Universal Sentence Encoder uses a dual-encoder transformer architecture to generate sentence embeddings trained using parallel corpora for 16 languages.
- **LASER:** Language Agnostic Sentence Representation is designed based on an encoder-decoder architecture using LSTM networks. The model is trained with big parallel datasets and performs max-pooling to get the sentence embedding from the stacked network. It supports 93 languages.
- **LaBSE:** Language-agnostic BERT Sentence Embedding (LaBSE) is a dual-encoder model based on BERT. The model was trained on 6 billion parallel sentence pairs over 109 languages.
- **XLM-R ← SBERT-nli-stsb / XLM-R ← SBERT-paraphrases:** The sentence transformer models generate sentence embeddings using the *teacher-student* architecture, where the XLM-R student model is trained with parallel sentences for across languages with knowledge transfer from the fine-tuned English SBERT-nli-stsb or SBERT-paraphrases as the teacher model.

We have also given the source of the baseline models which can be found in Table 5.3

Datasets	Link
mBERT	https://huggingface.co/bert-base-multilingual-cased
XML-R	https://huggingface.co/transformers/model_doc/xlmroberta.html
LASER	https://tfhub.dev/google/LaBSE/1
XLM-R ← SBERT-nli-stsb / XLM-R ← SBERT-paraphrases	https://www.sbert.net/docs/pretrained_models.html
mUSE	https://tfhub.dev/universal-sentence-encoder-xling/many

Table 5.3: Source of the competing approaches and models used as baselines.

5.2.4.2 *Multilingual Semantic Textual Similarity*

Understanding semantic textual similarity between monolingual and cross-lingual datasets is one of the major tasks for a sentence embedding model. We evaluate our model against the **STS** benchmark dataset (Cer et al., 2017), containing sentence pairs with scores indicating how semantically similar the sentences are. The SemEval dataset consists of annotated sentences for EN-EN, AR-AR, ES-ES, EN-AR, EN-ES,

and EN-TR language pairs. Further, we also use the EN-DE, EN-IT and EN-NL test sets from multilingual SBERT (Reimers & Gurevych, 2020). For evaluation, we compute the cosine similarity between sentence pair embeddings and obtain the *Spearman rank correlation*, ρ across the computed similarities and gold scores.

Approaches / Languages	EN-EN	ES-ES	EN-DE	EN-TR	EN-ES	EN-FR
Supervised Methods						
XLM-R ← SBERT-nli-stsb	82.5	83.5	78.9	74.0	79.7	78.5
XLM-R ← SBERT-paraphrases	**88.8**	86.3	**84.0**	**80.9**	**83.1**	**84.9**
LASER	77.6	79.7	64.2	72.0	57.9	69.1
LaBSE	79.4	80.8	73.8	72.0	65.5	77.0
mUSE	86.4	**86.9**	82.1	75.5	79.6	82.6
Unsupervised Methods						
mBERT mean	54.4	56.7	33.9	16.0	21.5	33.0
XLM-R mean	50.7	51.8	21.3	9.2	10.9	16.6
Proposed methods						
DuEAM$_{wklysupv}$	**81.9**	**83.1**	**69.4**	**68.6**	**64.6**	**69.6**
DuEAM$_{unsupv}$	*80.2*	*81.5*	*64.6*	*63.7*	*58.2*	*62.1*

Table 5.4: Spearman rank correlation (ρ) results for Semantic Textual Similarity (STS) datasets. The results are reported as ρ × 100, with baseline performances as reported in Reimers and Gurevych, 2020.

As shown in Table 5.4, the unsupervised baselines based on the language models (mBERT and XLM-R) perform quite poorly, suggesting that the obtained cross-lingual sentence embeddings are not well aligned in the vector space. While trained with multi-task learning, *DuEAM* achieved a significant improvement both for monolingual and cross-lingual datasets. For cross-lingual settings, both the *DuEAM* models significantly outperform the unsupervised models, with an average improvement of **41.9** and **37.2** respectively on Spearman rank correlation (ρ) score. Similarly, on the *monolingual* datasets (EN-EN and ES-ES), our models achieve an improvement of **26.9** and **21.4** points (on average), based on the rank correlation score.

It is interesting to note that *DuEAM* achieves better results, compared to the supervised LASER and LaBSE approaches, for both the monolingual datasets (EN-EN and ES-ES). In cross-lingual settings, in certain cases, both DuEAM$_{\text{unsupv}}$ and DuEAM$_{\text{wklysupv}}$ are seen to outperform LASER (e.g., EN-DE and EN-ES language pairs).

5.2.4.3 *Zero Shot Testing on Semantic Textual Similarity benchmark*

An important property of the embedding techniques is *"zero shot learning"*, i.e., to robustly generalize to languages that the model has not been trained on, by inherent knowledge transfer from the other languages. To study the efficiency of *DuEAM* in zero-shot scenarios, in this setting, we train the weakly-supervised model only on the EN-DE (English-German) training dataset (as in Section 5.2.3) and test on the other language pairs, including monolingual sentence similarity for ES-ES and AR-AR.

Models	ES-ES	AR-AR	EN-ES	EN-AR	EN-TR	EN-FR	EN-NL	EN-IT
mBERT mean	56.7	50.9	21.5	16.7	16.0	33.0	35.6	34.0
XLM-R mean	51.8	25.7	10.9	17.4	9.2	16.6	26.0	22.9
DuEAM$_{wklysupv}$	**78.64**	**69.67**	**56.54**	**54.29**	**58.35**	**62.03**	**67.61**	**59.8**

Table 5.5: Zero-shot Spearman rank correlation ρ results for Semantic Textual Similarity (STS) datasets, where models are trained on EN-DE non-parallel data.

From Table 5.5, we can see that even for zero shot learning *DuEAM* outperforms the unsupervised baseline models, across all the monolingual and cross-lingual STS datasets, with improvements in Spearman rank correlation score of around **20**. Thus, our architecture based on dual encoder with multi-task

learning provides better cross-lingual sentence embeddings, making it robust across diverse languages with improved performance.

5.2.4.4 *Bitext Mining Task*

Efficient multilingual sentence embeddings should have a good understanding of sentence parallelism and should be able to retrieve good translation pairs across corpora in different languages. Intuitively, sentence translation pairs should be equivalent in terms of semantic similarity, and hence their cross-lingual embeddings should be very similar.

To evaluate the performance of our method, we conduct experiments on the **BUCC** benchmark mining task – parallel sentence extraction from two different monolingual corpora. We use the data available from the 2018 shared task, consisting of corpora for four language pairs (FR-EN, DE-EN, RU-EN, and ZH-EN), with a subset of parallel sentences demarked as the gold mapping for each language pair. The data is split into train and test set, and the training data is used to find a threshold for the scoring function, such that sentence pairs above the threshold are returned as parallel sentences. Performance is measured using *F1 score*. Similar to Reimers and Gurevych, 2020, in this setting, we use the margin scoring function as:

$$\text{score}(x, y) = \text{margin}(\cos(x, y), \cos^*(x, y)), \text{ with}$$
$$\cos^*(x, y) = \sum_{z \in \text{NN}_k(x)} \frac{\cos(x, z)}{2K} + \sum_{z \in \text{NN}_k(y)} \frac{\cos(y, z)}{2K}$$

where x, y are the sentence embeddings, $\text{NN}_k(x)$ is the k nearest neighbours of x in other languages excluding duplicates, and $\text{margin}(a, b) = \frac{a}{b}$.

Approaches / Languages	DE-EN	FR-EN	RU-EN	ZH-EN
Supervised Methods				
XLM-R ← SBERT-nli-stsb	86.8	84.4	86.3	85.1
LASER	95.4	92.4	92.3	91.7
LaBSE	**95.9**	**92.5**	**92.4**	**93**
mUSE	88.5	86.3	89.1	86.9
Unsupervised Methods				
mBERT mean	44.1	47.2	38	37.4
XLM-R mean	5.2	6.6	22.1	12.4
Kvapilíková et al., 2020	80.2	78.8	77.1	67.0
Proposed Methods				
DuEAM$_{wklysupv}$	**84.9**	**81.3**	**82.0**	**78.6**
DuEAM$_{unsupv}$	*80.9*	*79.3*	*78.4*	*70.0*

Table 5.6: F1 score on BUCC bitext mining task. Baseline results taken from Reimers and Gurevych, 2020.

Table 5.6 shows the performance of the approaches on the BUCC task. For the unsupervised setting, we observe that for all language pairs *DuEAM* performs significantly better than XLM-R mean and mBERT mean methods, with an improvement of nearly **35.6** F1 score over mBERT. Additionally, we compare our model with the recent approach by Kvapilíková et al., 2020, specifically trained for bitext mining task with synthetic parallel data. We see that *DuEAM* also outperforms these unsupervised models across all language pairs.

Further, our weakly-supervised model achieves competitive results compared to the supervised model of XLM-R ← SBERT-nli-stsb, trained with large parallel datasets. Observe that LASER and LaBSE achieve high accuracy, as they are specifically designed and trained to identify translations between languages. On the other hand, although *DuEAM* is not trained with any parallel data, we are able to effectively extract parallel sentences (bitext mining task) owing to our multi-task learning.

5.2.4.5 *Cross-lingual Parallel Sentence Matching*

In this section, we compare the performance of the approaches in extracting parallel sentences using the **Tatoeba** benchmark of (Artetxe & Schwenk, 2019). From Table 5.7, on well-resourced languages, we observe *DuEAM* to perform significantly better than the unsupervised approach of Kvapilíková et al., 2020, with results comparable to the supervised methods of SBERT and LASER – similar as above – efficiently extracting parallel sentences.

Model	DE	HI	ZH	EL
XLM-R ← SBERT-paraphrases	98.7	96.4	95.0	95.5
LASER	99.0	94.7	95.4	95.0
Kvapilíková et al., 2020	83.1	53.4	-	51.3
$DuEAM_{wklysupv}$	96.0	92.9	90.2	87.4
$DuEAM_{unsupv}$	93.4	83.5	85.2	82.0

Table 5.7: Average accuracy on Tatoeba dataset in both directions (EN to target language and vice-versa). Here DE,HI,EL,ZH refer to German, Hindi, Greek and Mandarin Chinese respectively. Baseline results taken from Reimers and Gurevych, 2020.

5.2.4.6 *Tatoeba Under-Resourced Languages*

We now evaluate the robustness of the approaches for extracting parallel sentences for under-resourced languages on the Tatoeba benchmark. In this setting, we consider two scenarios – (i) *zero-shot learning* and (ii) training on *small scale* non-parallel datasets. We compare the results with the supervised models of SBERT (Reimers & Gurevych, 2020) and LASER (Artetxe & Schwenk, 2019) respectively, with different baseline languages for LASER for which it has been pre-trained and results have been published.

Zero-shot Transfer. We perform zero-shot transfer on different under-resourced languages: (i) Telugu (TE, Dravidian family), (ii) Tagalog (TL, Malayo-Polynesian family), (iii) Irish (Gaelic) (GA, Celtic family), and (iv) Afrikaans (AF, Germanic family).

Model	AF	TE	TL	GA
XLM-R ← SBERT-para	84.2	89.1	32.4	18.6
$DuEAM_{wklysupv}$	**84.8**	**90.6**	**60.6**	**42.0**
$DuEAM_{unsupv}$	79.9	78.6	*56.8*	*35.0*

(a)

Model	KA	AM
LASER	35.9	42.0
$DuEAM_{wklysupv}$	**76.4**	**56.0**
$DuEAM_{unsupv}$	*70.7*	*46.4*

(b)

Table 5.8: Average accuracy on Tatoeba data in both directions (EN to target language and vice versa) for (a) zero-shot learning, and (b) small training set. Baseline results as in Reimers and Gurevych, 2020.

We observe from Table 5.8(a) that for Tagalog and Irish (Gaelic) both *DuEAM* models performed significantly better than the supervised multilingual S-BERT with an improvement of around 25% on average for weakly-supervised model and 21% on average for unsupervised model, while for Afrikaans and Telugu, we achieved 1.5% better accuracy on average for weakly-supervised model. These results show that, even without explicit learning for different under-resourced languages, our models can robustly handle zero-shot learning across different language families for generating efficient sentence embeddings.

Small Scale Dataset Training. To explore the model performance while trained on small datasets (for scenarios where limited data is available for under-resourced languages), we experiment on two under-resourced languages: Georgian (KA, Kartvelian family) and Amharic (AM, Ethio-Semitic family). We train *DuEAM* with *only* 20K *non-parallel* EN-KA and EN-AM sentence pairs, whereas the baseline supervised model LASER is trained with 296K and 88K parallel sentence pairs respectively. In Table 5.8(b), we see that our models outperform LASER for both Georgian and Amharic, achieving higher accuracy on under-resourced languages even when trained on a much smaller non-parallel dataset. In fact, weakly-supervised

DuEAM performs better than SBERT (Reimers & Gurevych, 2020), producing an accuracy of 72.4% on KA. The anchor-learner architecture with the unsupervised joint loss function provides such robustness and better cross-lingual understanding in *DuEAM*.

5.2.4.7 *Tatoeba Benchmark Results on* 58 *Languages*

We have performed all our experiments on the 58 languages of Tatoeba test datasets. Evaluation for the parallel sentence matching task is done by finding the most similar sentence between two languages based on their cosine similarity. We have calculated accuracy in both directions (English to target language and vice versa), and have reported the average accuracy of the two. We have reported performance results of the different approaches based on three settings:

- Model performance on languages that it has been trained,
- Zero-shot model performance on untrained languages, while compared with supervised trained models,
- Model performance on under-resourced languages for which it is untrained.

Performance on Trained Languages: Table 5.9 reports the performance of the models across 12 languages. We have compared our model with supervised baseline and unsupervised baseline models. We can see that across all 12 languages our model achieved high accuracy compare to unsupervised model. Our model also achieved comparative results with supervised model XLM-R ← SBERT-nli-stsb and LASER for some languages.

Model / Languages	bul	cmn	deu	ell	fra	hin	rus	spa	tur	tha	urd	vie
Supervised Approaches												
XLM-R ← SBERT-paraphrases	94.0	95.0	98.7	**95.5**	94.7	**96.4**	93.5	**98.0**	97.2	**96.3**	**92.2**	**97.2**
LASER	**95.0**	**95.4**	**99.0**	95.0	**95.6**	94.7	**94.6**	**98.0**	**97.5**	95.4	81.9	96.8
Unsupervised Approaches												
(Kvapilíková et al., 2020)	56.0	–	83.1	51.3	–	53.4	–	–	–	–	43.7	–
Proposed Approaches												
$\text{DuEAM}_{\text{wklysupv}}$	**86.0**	**90.2**	**96.0**	**87.4**	**91.5**	**92.9**	**90.0**	**93.0**	**89.6**	**90.1**	**77.8**	**92.0**
$\text{DuEAM}_{\text{unsupv}}$	**82.5**	**85.2**	**93.4**	**82.0**	**87.7**	**83.5**	**85.5**	**89.5**	**84.1**	**82.4**	**67.9**	**89.6**

Table 5.9: Average accuracy on the Tatoeba test set in both directions (en to target language and vice versa) on **trained languages**.

Zero-shot Transfer: We have compared our model accuracy on 30 untrained languages and compared with baseline models. Table 5.10 shows that, for all 30 languages our model has achieved state-of-the-art unsupervised results. While the supervised models are trained on the parallel datasets for these languages, for some languages our model achieved comparative results even in zero-shot settings.

Zero-shot Transfer on Under-Resourced Languages: While our model has achieved high accuracy on a wide range of languages, we have compared our model with a supervised baseline on 16 under-resourced languages from different language families for zero-shot transfer. From Table 5.11 we see that, across all languages, the weakly-supervised *DuEAM* achieved higher accuracy than the supervised baseline of XLM-R ← SBERT-paraphrases. In fact our unsupervised *DuEAM* performed better than XLM-R ← SBERT-paraphrases for most of the languages.

Overall, our unsupervised and weakly-supervised *DuEAM* perform significantly better than the existing unsupervised approach, and is comparable with the supervised models across diverse languages. Our model also efficiently supports zero-shot transfer learning and is robust for under-resourced languages.

Model / Languages	ara	cat	ces	dan	est	fin	glg	heb	hrv	hun	hye
Supervised Approaches											
XLM-R ← SBERT-paraphrases	87.7	**96.4**	96.3	**96.2**	95.8	**96.4**	**96.0**	88.4	97.0	94.7	**91.3**
LASER	**92.0**	95.9	**96.5**	96.0	**96.7**	96.3	95.5	**92.2**	**97.2**	**96.0**	36.1
Unsupervised Approaches											
(Kvapilíková et al., 2020)	41.1	66.9	53.5	–	39.0	47.5	66.9	–	68.2	–	–
Proposed Approaches											
DuEAM$_{wklysupv}$	**70.7**	**83.3**	**85.3**	**92.3**	**73.0**	**88.70**	**85.0**	**73.1**	**88.7**	**86.1**	**79.0**
DuEAM$_{unsupv}$	**61.6**	**80.0**	**79.80**	**90.0**	**70.0**	**83.7**	**80.7**	**69.0**	**84.3**	**81.5**	**74.1**

Model / Languages	ita	jpn	kat	kor	lit	lvs	mar	mkd	mon	nob	pes
Supervised Approaches											
XLM-R ← SBERT-paraphrases	94.9	90.7	**91.4**	**90.1**	95.8	**96.4**	91.0	92.2	**91.7**	98.0	**94.8**
LASER	**95.3**	**94.2**	35.9	88.9	**96.2**	95.4	**91.5**	**94.7**	8.2	**98.8**	93.4
Unsupervised Approaches											
(Kvapilíková et al., 2020)	–	54.4	41.4	–	43.9		37.3	–	29.0	–	–
Proposed Approaches											
DuEAM$_{wklysupv}$	**85.7**	**84.2**	**71.7**	**81.3**	**83.2**	**81.2**	**78.9**	**75.2**	**74.7**	**94.8**	**88.9**
DuEAM$_{unsupv}$	**83.1**	**77.4**	**68.2**	**75.8**	**78.9**	**76.6**	**73.4**	**71.2**	**71.5**	**93.2**	**83.4**

Model / Languages	por	ron	slk	slv	sqi	srp	pol	ind
Supervised Approaches								
XLM-R ← SBERT-paraphrases	94.8	96.4	96.2	95.5	97.5	93.8	97.0	94.1
LASER	**95.2**	**97.4**	**96.6**	**95.9**	**98.0**	**95.3**	**97.8**	**94.5**
Unsupervised Approaches								
(Kvapilíková et al., 2020)	–	–	–	–	–	–	–	64.9
Proposed Approaches								
DuEAM$_{wklysupv}$	**91.2**	**88.5**	**86.2**	**80.5**	**79.9**	**83.7**	**90.4**	**89.5**
DuEAM$_{unsupv}$	**89.5**	**87.0**	**80.2**	**77.2**	**76.6**	**80.3**	**88.4**	**87.7**

Table 5.10: Average accuracy on the Tatoeba test set in both directions (en to target language and vice versa) on **untrained languages**. All the baseline models are trained on parallel training datasets.

5.2.4.8 *Monolingual Classification Performance*

A multilingual sentence embedding framework is expected to produce efficient results in monolingual settings. To evaluate the performance on monolingual classification tasks, we now study the performance of the unsupervised variant of *DuEAM* on the SentEval benchmark (Conneau & Kiela, 2018). We compare *DuEAM* to other sentence embedding frameworks on four tasks :

- **MR**: Movie reviews positive/negative sentiment analysis (Pang & Lee, 2005)
- **SUBJ**: Subjectivity/objectivity prediction of reviews (Pang & Lee, 2004)
- **TREC**: Question type classification on six classes (Li & Roth, 2002)
- **SST2**: Stanford binary sentiment classification (Reimers & Gurevych, 2019; Socher et al., 2013)

In Table 5.12, we can see quite satisfactory results produced by our *DuEAM* framework. On every task, the *DuEAM$_{unsupv}$* model surpasses the performance of the supervised LASER model whereas in case of the **TREC** task the results are better than even the supervised multilingual S-BERT model.

Overall, the monolingual and multi-lingual experimental results depict *DuEAM* to effectively capture cross-lingual semantic understanding (without parallel training data) to generate efficient sentence embed-

Model / Languages	bre	ceb	cym	epo	eus	gla	isl	jav	khm	lat	swg
XLM-R ← SBERT-paraphrases	10.1	11.7	34.9	68.8	48.6	7.5	75.8	37.0	64.8	28.0	**33.9**
$\text{DuEAM}_{\text{wklysupv}}$	**11.5**	**14.8**	**52.7**	**79.2**	**66.0**	**21.9**	**81.9**	**40.2**	**65.7**	**44.0**	**33.9**
$\text{DuEAM}_{\text{unsupv}}$	9.8	**12.3**	**46.0**	**74.0**	**58.2**	**16.0**	**78.5**	36.1	52.7	**38.9**	31.2

Model / Languages	swh	uzb	war	xho	yid
XLM-R ← SBERT-paraphrases	27.6	32.6	11.4	11.6	52.7
$\text{DuEAM}_{\text{wklysupv}}$	**40.2**	**40.1**	**13.0**	**15.5**	**53.7**
$\text{DuEAM}_{\text{unsupv}}$	**33.3**	**35.5**	10.8	**14.9**	46.7

Table 5.11: Average accuracy on the Tatoeba test set in both directions (en to target language and vice versa) on **untrained under-resourced languages**.

Model	MR	SUBJ	TREC	SST2
XLM-R ← SBERT-paraphrases	**81.26**	**93.89**	91.2	**87.7**
LASER	75.29	92.07	91.0	79.9
$\textit{DuEAM}_{\textit{unsupv}}$	76.28	92.86	**92.2**	81.1

Table 5.12: Evaluation accuracy on a subset of SentEval benchmark (results based on 10-fold cross-validation).

dings by alignment of multiple languages in the same vector space. Observe that *DuEAM* is trained on only 1GB of data, while other supervised techniques are trained on around 10x or more data.

5.2.5 Ablation Study

Necessity of Multi-task Learning. One of the important features of *DuEAM* is *multi-task joint learning* via the dual-encoder based anchor-learner architecture. To explore the necessity of the different factors for our learning loss function, we use the Tatoeba dataset for DE-EN, FR-EN, and HI-EN language pairs. We train weakly-supervised $\textit{DuEAM}_{\textit{wklysupv}}$ in three variants:

- With only the anchor module loss term $\mathcal{L}_{\mathcal{A}}$ (Eq. 5.1), which considers semantic similarity between sentences,
- Using only the translation mining loss term $\mathcal{L}_{\mathcal{B}}$ (Eq. 5.2), which identifies the best translation pairs,
- The full multi-task learning objective $\mathcal{L}$.

From Figure 5.3, we observe that the learning objective factors $\mathcal{L}_{\mathcal{A}}$ and $\mathcal{L}_{\mathcal{B}}$ individually perform quite poorly. However, the proposed multi-task training performs efficiently providing a high accuracy in the range of 92% to 96%. Similar results are observed across the language pairs considered.

Training Dataset Size. Table 5.8(b) depicts that *DuEAM* outperforms LASER on under-resourced languages with minimal training. To understand the impact of the size of the training dataset, we evaluated the performance of *DuEAM* on Tatoeba data for Georgian (KA), with varying training sizes.

Zero-Shot	1K	5K	10K	15K	20K	25K	30K
71.8	72.9	73.5	74.8	75.9	76.41	76.40	76.42

Table 5.13: Average accuracy of $\textit{DuEAM}_{\textit{wklysupv}}$ on Tatoeba (in both directions) for Georgian language (KA) with varying training data sizes.

In Table 5.13 we see a healthy performance improvement when our weakly-supervised model is trained with the language-specific dataset, with around 4.5% improvement over zero-shot learning given training data of size 20K sentences. Thereafter, the improvement is seen to be incremental. Thus, although *DuEAM* demonstrates zero-shot learning capabilities, a small amount of language-specific data further boosts the performance.

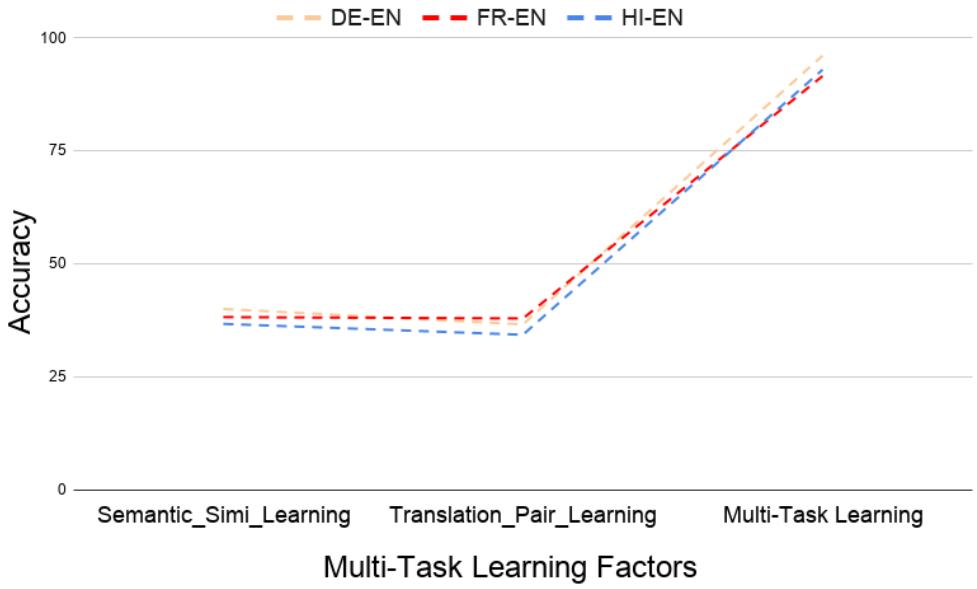

Figure 5.3: Average accuracy on Tatoeba across language pairs with individual objectives of mutli-task learning.

Dataset	DE	FR	ES
25K XNLI dataset	88.7	82.6	86.5
25K TED2020 parallel dataset	89.2	83.8	86.9
12.5K XNLI + 12.5K parallel dataset	90.2	85.1	89.5

Table 5.14: Average accuracy of $DuEAM_{wklysupv}$ on Tatoeba (considering both directions).

Training Dataset Type. We explore the performance of the unsupervised loss function in *DuEAM* on various training data scenarios. We consider EN, DE, FR, and ES languages under 3 training settings:

- 25K sentences from XNLI (weakly supervised),
- 25K parallel sentences from TED2020 (supervised),
- 12.5K sentences from each of the datasets.

Table 5.14 depicts similar performances when trained on XNLI or on a parallel corpus alone, while a combination outperforms the others – showcasing the stability of our proposed *unsupervised loss joint loss function* (on training data), removing the dependency on parallel training datasets.

Weight Parameter Value Selection. Weight parameter selection while training the multi-task model is very important. We experimented with the weight parameter over a range of values and set to the value for which the model has performed best while training. We have given a snapshot of the model performances over the different weight parameters in Figure 5.4. We have calculated the average Spearman rank correlation (ρ) results for STS datasets for six language pairs. From the figure we can see that the best model performance achieved with weight parameter 0.003. Higher weight parameter value decreased the performance. This helps us to understand the importance of the weight parameter while training the model in multi-task settings.

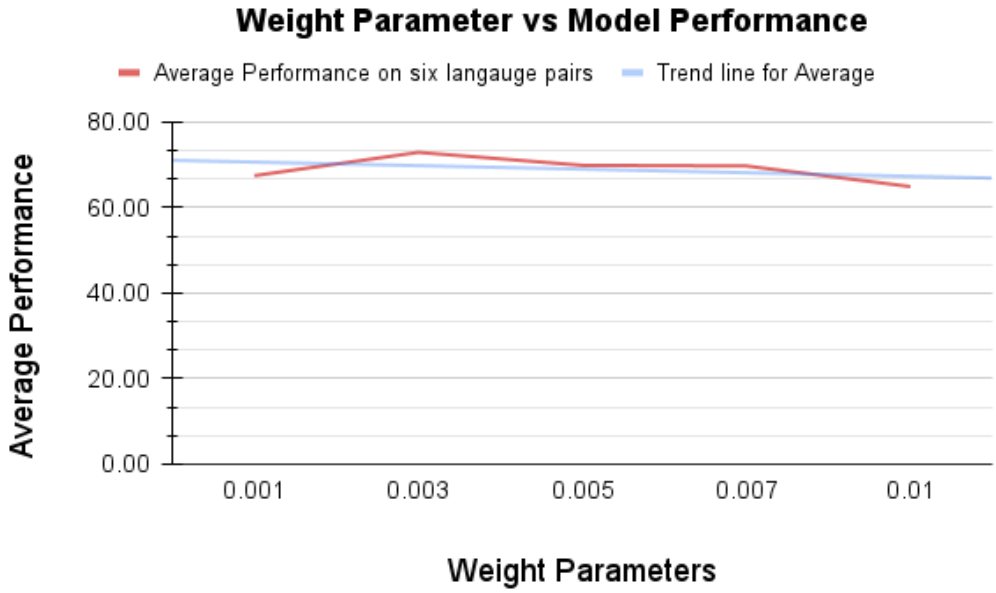

Figure 5.4: Average Spearman rank correlation (ρ) results for Semantic Textual Similarity (STS) datasets for six language pairs.

Cos. Simi.	Sentence pairs	Results
0.9961	**DE:** das ist der Geburtstag von Muiriel! **EN:** it is Muiriel's birthday!	**True Positives**
0.9465	**DE:** das ist der Geburtstag von Muiriel! **EN:** Happy birthday, Muiriel!	
0.9870	**DE:** Das Wesen der Freiheit leigt in der Mathematik. **EN:** The essence of Mathematics is liberty.	**False Positives**
0.9860 (*0.9971*)	**DE:** Das Wesen der Freiheit leigt in der Mathematik. **EN:** The essence of liberty is mathematics. ***(Correct EN Translation: The essence of freedom lies in Mathematics.***	

Table 5.15: Raw cosine similarity value on the Tatoeba DE-EN dataset.

5.2.6 Discussion: Semantic Similarity

In general, applications use cosine similarity between sentence embeddings to gauge semantic textual similarity. We provide a performance analysis of *DuEAM* based on the raw cosine similarity score on Tatoeba DE-EN data. For example, the German sentence "das ist der Geburtstag von Muiriel!" ("That is the birthday of Muiriel") has the highest cosine similarity with its English translation "it is Muiriel's birthday!", although the sentence "Happy birthday, Muiriel!" is very similar. The detailed result can be found in Table 5.15. This depicts that *DuEAM* can capture fine-grained semantic difference among similar sentences.

On the other hand, for the German sentence "Das Wesen der Freiheit liegt in der Mathematik." we obtain a higher cosine similarity score for the English sentence "The essence of mathematics is liberty.". In fact, the true translation "The essence of freedom lies in mathematics" (achieving the highest cosine-similarity but absent in the Tatoeba dataset) is closer to "The essence of liberty is mathematics.". Although the similarity score is almost equal, our model is unable to identify the correct word ordering in highly overlapping sentences as the WMD measure is inherently word-order agnostic. Multilingual SBERT too fails in this scenario, but with a higher difference in cosine-similarity between sentences, 0.01 compared to

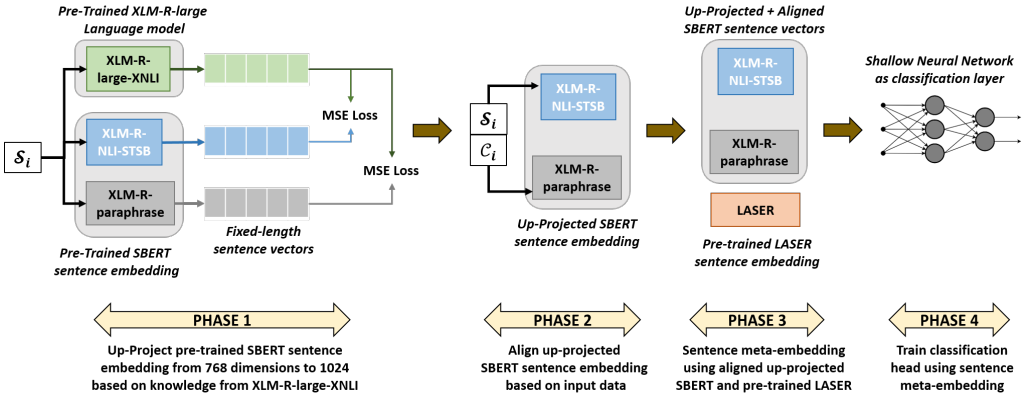

Figure 5.5: Overall architecture of MUFIN framework for learning multi-lingual task-specific sentence embeddings.

0.001 in *DuEAM* (using the translation mining $\mathcal{L}_{\mathcal{B}}$ loss factor). Use of Wikipedia dumps for training such sentence embedding models forms an interesting future study.

In the next section we will describe our proposed MUFIN which enhances the performance of the multilingual sentence embedding frameworks on the multilingual classification tasks.

5.3 ENHANCING SEMANTIC UNDERSTANDING OF SENTENCE EMBEDDING FRAMEWORKS

5.3.1 Our Approach

This section describes the workings of the proposed MUFIN framework for enhanced multi-lingual task-specific sentence embedding generation. It consists of two main components:

- Unsupervised up-projection of fixed-length sentence embeddings to incorporate additional knowledge from language models,
- Alignment of the new projected embedding space for domain-specific task.

The overall architecture comprises 4 modules (shown in Fig. 5.5).

Consider, the training dataset $\mathcal{T}$ to be a collection of short text or sentences $\{s_1, s_2, \cdots, s_n\}$ with associated labels or categories $\{c_1, c_2, \cdots, c_n\}$. Similarly, the test or inference dataset $\mathcal{I}$ consists of sentences but without any knowledge of the labels. The objective here is to assign the sentences in $\mathcal{I}$ to the predefined categories or labels present in the training set.

5.3.1.1 *Unsupervised Embedding Up-Projection Module*

The first module of MUFIN performs *dimensionality expansion* on the fixed-length sentence embeddings, termed as *up-projection*, using a larger "un-fine-tuned" pre-trained language model. Assume $\mathbb{S}$ to be a sentence embedding framework providing vector representation of $\mathbb{N}$ dimensions, while $\mathbb{L}$ is a pre-trained language model providing vector representation of $\mathbb{M}$ dimensions ($\mathbb{M} > \mathbb{N}$).

As shown in Fig. 5.5, we consider the SBERT-XLM-R-paraphrase sentence embedding model [3] to provide sentence vectors of size $N = 768$, and pre-trained *XLM-R-large-XNLI* language model (with vector size $M = 1024$) is used for up-projecting the sentence embeddings.

1. The up-projection is achieved with the help of a fully connected neural network via *self reconstruction* of data based on *back-propagation* (Rumelhart et al., 1986) through the network layers. An input sentence embedding $s_i \in \mathbb{R}^N$ is passed through the neural network which learns to output higher-dimensional embedding $s'_i \in \mathbb{R}^M$ via a non-linear transformation $f : \mathbb{R}^N \rightarrow \mathbb{R}^M$.

2. The projected embeddings are then transferred to the same vector space as that of the large language models by minimising the the mean-squared loss between their sentence embeddings. Thus for a given batch size of β sentences, we optimize Equation 5.3 :

$$\frac{1}{\beta}\sum_{i=0}^{\beta}[\mathbb{S}(s_i) - \mathbb{L}(s_i)]^2 \tag{5.3}$$

   where $\mathbb{L}(s_i)$ is the sentence embedding obtained from the language model, $\mathbb{S}(s_i)$ is the embeddings obtained from sentence embedding model, and $s_i \in \mathcal{T}$.

In our setting, the 768 dimension SBERT embeddings (of the training data) are up-projected and transferred to the 1024 dimensional representation space of the XLM-R-large language model. This up-projection stage helps to supplement sentence embeddings from $\mathbb{S}$ by extracting additional semantic knowledge from the language model $\mathbb{L}$, and represent them together in a common space. One of the main advantages of our framework is that this module is *unsupervised*, which eliminates the need for expensive fine-tuning of the pre-trained large language model.

5.3.1.2 *Domain Alignment of Up-Projected Embedding*

Since, MUFIN aims to learn task specific sentence embeddings, in this module, the up-projected sentence embeddings are aligned to the downstream application domain using the task specific training dataset $\mathcal{T}$. To this end, we refine the obtained embeddings (from the previous step) using the training sentences and associated labels.

The up-projected sentence embeddings $s'_i \in \mathbb{R}^M$ are provided to another fully connected neural network with an output *softmax layer*, which learns to predict the label or category probability distribution (for an input embedding) across the n training labels, using Equation 5.4:

$$p_{ij} = \frac{\exp(z_j)}{\sum_{t=1}^{n}\exp(z_t)} \tag{5.4}$$

where n is the number of classes, z_i provides the hidden layer representation provided to softmax, and p_{ij} gives the probability that sentence embedding s_i belongs to class c_j.

This enables *domain adaptation* and aligns the embeddings to be tuned specifically for the downstream task at hand. This is akin to the process of language model fine-tuning – and gears the obtained text representations in MUFIN for the specific application needs.

5.3.1.3 *Meta Embedding Using Aligned Sentence Embedding*

This diversity of different sentence embedding techniques captures different aspects of semantics, and demonstrates varying degrees of success on different classes of tasks. As such, *meta embedding* studies

3 Obtained from www.sbert.net/docs/pretrained_models.html.

different combinations of embeddings from multiple sources (Coates & Bollegala, 2018). In this module, we adopt the concept of meta embedding on the sentence embeddings obtained from the previous layer. The intuition here is that various aligned sentence encoders (which differ in architecture and pre-training data) demonstrate complementary strengths, which upon being fused together provides improved performance – further enabling MUFIN to bridge the gap with fine-tuned large language models.

To this end, we combine aligned up-projected embeddings obtained from two different SBERT models, namely *XLM-R-paraphrases* and *XLM-R-NLI-STSB*. To demonstrate the flexibility of our framework and to also incorporate diversity, we further combine text representations from LASER (Artetxe & Schwenk, 2019) (in its native form, i.e., without any projection or alignment). The choice to using "frozen" generic LASER embeddings is to enable generalized semantic knowledge to also be integrated within our framework, along with the domain semantic understanding obtained from the aligned up-projected encodings. The final sentence representation is obtained by *concatenating* the 3 vectors obtained from LASER and two aligned SBERT models.

Note, other sentence embedding models can easily be incorporated. However, addition of more sentence embedding models and use of other meta embeddings techniques (using Singular Value Decomposition (SVD) (Klema & Laub, 1980; W. Yin & Schütze, 2016) and Autoencoders (AE) (Baldi, 2012; Bollegala & Bao, 2018)) have not been considered in the current setting for two reasons:

- The proposed framework is meant to be *light-weight* in terms of training time and model complexity, hence adding more complexity will impact these parameters
- Analyzing different combinations of available sentence encoders (there are 10s of such pre-trained models) and finding the best, is orthogonal to our contribution.

5.3.1.4 *Sentence Meta Embedding Classification*

The final module is used to learn to classify the sentence meta-embeddings obtained from the above step. We use a shallow fully connected neural network with a softmax layer (similar to Sec. 5.3.1.2), trained to classify the input sentence meta-embeddings to their corresponding labels.

Inference. Upon arrival of a new test sentence to classify, its SBERT embeddings are obtained from the already trained aligned up-project models (see Sec. 5.3.1.2) along with its "frozen" LASER representations, to create the meta-embedding. This final meta embedding is fed to the final trained classifier layer to infer the class or label of the test sentence. Given, the embedding models are already trained and the final classification module is a small shallow network, the inference time is in the order of milliseconds per sample – making it practical for enterprise settings.

5.3.2 Aligned Individual Sentence Embedding Framework

We showcase the performance of individual supervised and unsupervised sentence embedding frameworks using our proposed methodology on:

- ***Aligned DuEAM UP*** – Classification based on the aligned and up-projected unsupervised sentence embedding framework *DuEAM* proposed in Section 5.2.1.
- ***Aligned SBERT-XLM-R-paraphrases UP*** – Classification based on the aligned and up-projected supervised sentence embedding framework *SBERT-XLM-R-paraphrases*.

While training the models individually, the aligned sentence embedding models individually are passed through the softmax layer as described in Step 5.3.1.4.

5.3.3 Empirical Evaluation

We evaluate the proposed MUFIN framework on different benchmark classification tasks, modelling different applications. Considering the lack and/or expense of multi-lingual training data, we adopt a *zero-shot* scenario – wherein the framework is trained only on a single language (e.g., English) but evaluated on different languages. We compared the performance of the algorithms using the standard *accuracy* measure for classification tasks.

5.3.3.1 *Datasets*

We consider the different application scenarios for dataset selection:

- Applications like question matching (for FAQ retrieval Assem et al., 2021) involve understanding semantic similarity between pairs of short text (or sentence-pairs),
- Automated chatbots and digital assistants involve precise understanding of user intents and parsing of relevant information for the associated actions and,
- Analyzing user feedback and reviews for identifying user pain-points and improving operational processes.

To this end, we have evaluated MUFIN on the following datasets:

- **PAWS-X** - This is a paraphrase identification extending the Wikipedia portion of PAWS (Y. Zhang et al., 2019) to four languages for evaluation - *English* (EN), *German* (DE), *Spanish* (ES), and *French* (FR).
- **Query-Ad Matching (QADSM)** – This task aims to predict whether an advertisement is relevant to an input query, having two labels - *Good* and *Bad* and is evaluated on *English*, *German*, and *French*.
- **Multilingual Amazon Review** – Consists of product reviews across different Amazon products across a time span of four years (from 2015 to 2019). The dataset has total *five classes* based on the user rating provided, with 200,000 reviews for training and 5000 reviews for testing on different languages. We evaluate on *German*, *Spanish*, and *French*, while train only on *English*.
- **Natural Language Understanding (NLU)** – The in-house translated dataset is designed based on the existing benchmark for intent classification (X. Liu et al., 2019) consisting of user utterances with 64 unique intents. The dataset consists of *English* training dataset and is tested on machine translations to *Arabic* (AR) and *Spanish*.
- **MTOP** – *Multilingual Task-Oriented Semantic Parsing Benchmark* data released by Facebook, contains a combination of utterances from 11 domains and 117 user intents. This data consists of 22,288 utterances for English to train, and is tested on *German*, *Spanish*, and *French*.

Observe that PAWS-X and QADSM are text-pair classification tasks, while NLU and MTOP involve single sentence classification. Further, the text size in QADSM and Amazon Review datasets are larger than the others – capturing diverse industrial application settings for NLU.

Table 5.16: Accuracy on PAWS-X dataset.

Approaches / Languages	EN	DE	ES	FR
Baseline Methods				
XLM-R-large	93.5	87.8	88.4	87.2
Aligned SBERT-XLM-R-paraphrases (NP)	86.2	74.6	81.2	78.7
Aligned DuEAM (NP)	84.2	70.4	78.1	73.5
Aligned Meta-embedding (NP)	87.9	78.6	81.7	80.5
Proposed Methods				
Aligned SBERT-XLM-R-paraphrases (UP)	88.7	79.1	82.4	81.5
Aligned DuEAM (UP)	87.6	76.8	81.4	79.2
MUFIN	**90.5**	**82.1**	**83.6**	**82.2**

Here *UP* denotes up-projection and *NP* denotes no projection.

5.3.3.2 *Baselines*

We compare the classification accuracy performance of MUFIN against the following baselines:

- **XLM-Roberta-large:** We have used the large pre-trained language model XLM-R-large (Conneau et al., 2020) as baseline system. We fine-tune the language model to the different datasets – an expensive process that we wish to alleviate through our contribution.
- **SBERT-XLM-R-paraphrases:** The state-of-the-art sentence transformer trained on large parallel corpus for more than 50 languages using *teacher-student* architecture (Reimers & Gurevych, 2020).
- **DuEAM:** We also compared with our proposed unsupervised sentence embedding model DuEAM.

5.3.4 Experimental Evaluation

We have evaluated MUFIN on the above mentioned cross-lingual benchmarks. In all experiments the model is trained on *English* and tested on other languages. In each experiment we also compared MUFIN with our baseline meta embedding framework, consists of state-of-the-art two SBERT models (SBERT-XLM-R-NLI-STSB / SBERT-XLM-R-paraphrases) and LASER (Artetxe & Schwenk, 2019) model.

5.3.4.1 *PAWS-X*

Understanding semantic similarity between sentence pairs while doing sentence classification is one of the important tasks of sentence embedding frameworks. We have evaluated our model on the sentence pairs classification dataset PAWS-X (Liang et al., 2020). Observe, in Table 5.16 the aligned non-projected SBERT-XLM-R-paraphrases and DuEAM model perform less effectively in comparison to the XLM-R-large model. As expected, the baseline non-projected meta-embedding variant outperformed the aligned non-projected SBERT-XLM-R-paraphrases and DuEAM model, but there is a big performance gap with XLM-R-large. The proposed method MUFIN framework has significantly outperformed the baseline models and achieved comparative performance with respect to XLM-R-large model. Moreover, our proposed variant *projected aligned SBERT-XLM-R-paraphrases* has achieved better performance than both the baselines aligned SBERT model and aligned non-projected meta-embedding variant, both in monolingual and zero-shot settings. Interestingly, the *projected aligned DuEAM* not only outperformed the non-projected aligned base variants including the supervised SBERT-XLM-R-paraphrase model, but also it has achieved comparative results with respect to the supervised aligned non-projected meta embedding model for the language English, Spanish and French.

5.3.4.2 *Query-Ad Matching (QADSM)*

In the recent times, users tend to post queries irrespective of domain. Business sectors try to promote their products by reaching the maximum number of users with advertisements of their products, which emphasizes the importance of producing efficient match between query and advertisement using semantic similarity between sentences. We have evaluated our system MUFIN on the benchmark binary classification dataset. We observe from Table 5.17 that *projected aligned SBERT-XLM-R-paraphrases* variant

Table 5.17: Accuracy on QADSM dataset.

Approaches / Languages	EN	DE	FR
Baseline Methods			
XLM-R-large	71.9	69.6	71.7
Aligned SBERT-XLM-R-paraphrases (NP)	66.9	62.0	63.0
Aligned DuEAM (NP)	64.7	61.2	61.3
Aligned Meta-embedding (NP)	68.0	64.2	64.2
Proposed Methods			
Aligned SBERT-XLM-R-paraphrases (UP)	68.1	64.1	64.5
Aligned DuEAM (UP)	67.7	63.8	64.3
MUFIN	**69.7**	**66.2**	**66.7**

Here *UP* denotes up-projection and *NP* denotes no projection.

outperformed the baseline method, aligned SBERT-XLM-R-paraphrases, with an average improvement of 4.5 in accuracy score. The similar kind of results can be observed for the aligned projected DuEAM variant outperforming the baseline models by on an average 3.9 accuracy score. It even outperformed the non-projected meta-embedding model for the French language. Our proposed meta-embedding method MUFIN has outperformed the aligned non-projected meta-embedding variant with an average accuracy of 2.5. This indicates robustness of the proposed model. It understands semantic similarity between sentence pairs and achieve better sentence embeddings for sentence pairs for classification.

5.3.4.3 *Multilingual Amazon Review*

In the real world, user commented sentences tend to be long, noisy and hard to understand. The zero-shot evaluation result on this dataset gives an understating of how the model understands noisy sentence construction of target languages when trained only on *English*. From Table 5.18 we see that on an average the proposed MUFIN has outperformed the baseline variants with almost 2.3 point of accuracy. For this dataset this is a commendable gain in performance as the margin between baseline SBERT and DuEAM variants is very small. In the very noisy in-domain dataset, the performances of the supervised (SBERT) and unsupervised (DuEAM) aligned up-projected variants are increased, which highlights the stability of the framework. Also, the MUFIN model has achieved comparable results in comparison to the XLM-R-large model which highlights, in practical industrial settings in-domain alignment with our approach for the noisy datasets can easily be adaptable rather than training big language models with much more memory and time.

5.3.4.4 *Natural Language Understanding (NLU)*

While understanding semantic relatedness between sentence pairs for classification is an important task for sentence embeddings frameworks, efficiently classifying single sentences is also very important in real world scenario. We evaluate MUFIN on the natural language understanding benchmark data where user utterances are annotated with intents. To understand the model efficiency in zero-shot settings, the target

Table 5.18: Accuracy on Multilingual Amazon Review Dataset.

Approaches / Languages	EN	DE	FR	ES
Baseline Methods				
XLM-R-large	67.7	64.6	59.2	60.0
Aligned SBERT-XLM-R-paraphrases (NP)	65.1	59.6	54.9	55.5
Aligned DuEAM (NP)	64.6	58.9	54.3	54.8
Aligned Meta-embedding (NP)	65.9	61.7	56.2	57.0
Proposed Methods				
Aligned SBERT-XLM-R-paraphrases (UP)	65.7	60.5	55.5	56.7
Aligned DuEAM (UP)	65.2	59.9	54.8	55.9
MUFIN	**66.9**	**63.0**	**58.0**	**58.4**

Here *UP* denotes up-projection and *NP* denotes no projection.

language test datasets are in-house machine translated. Observe, in Table 5.19 MUFIN has comparable re-

Table 5.19: Accuracy on Multilingual NLU dataset.

Approaches / Languages	EN	AR	ES
Baseline Methods			
XLM-R-large	92.7	82.7	86.4
Aligned SBERT-XLM-R-paraphrases (NP)	89.0	80.0	83.0
Aligned DuEAM (NP)	88.2	78.6	82.8
Aligned Meta-embedding (NP)	90.1	80.7	84.7
Proposed Methods			
Aligned SBERT-XLM-R-paraphrases (UP)	91.2	80.3	84.8
DuEAM (UP)	90.4	79.6	83.3
MUFIN	**92.1**	**81.5**	**85.0**

Here *UP* denotes up-projection and *NP* denotes no projection.

sults with XLM-R-large in case of monolingual settings. In case of zero-shot as well as monolingual results, it outperforms the SBERT-XLM-R-paraphrases and non-projected aligned SBERT meta-embedding model. Interestingly in case of monolingual settings, the *aligned projected SBERT-XLM-R-paraphrases* variant outperformed non projected aligned SBERT meta-embeddings, and in the case of zero-shot settings it produced comparable results. This is significant as the *aligned projected SBERT-XLM-R-paraphrases* takes less time than the non projected aligned SBERT based meta-embedding model for training and inference, but produces more effective embeddings for single sentence classification. Moreover, the unsupervised *aligned projected DuEAM* model outperformed the baseline variants and it highlights that the models trained with low-resource settings can be deployed with high efficacy.

5.3.4.5 *MTOP Dataset*

While working with mixed domain single sentence classification datasets, understanding semantic similarity with respect to the annotated classes often hard task to solve. To evaluate the performance of the model MUFIN in mixed domain datasets, we have compared MUFIN with baseline systems for zero-shot learning and on English. Here the classes are annotated intents. In Table 5.20 we can see that the proposed variant *projected aligned SBERT-XLM-R-paraphrases* outperformed the state-of-the-art SBERT-XLM-R-paraphrases model in both monolingual and zero-shot settings. Similarly the *projected aligned DuEAM* outperformed the baseline version in both monolingual and zero-shot settings. In fact the *projected aligned DuEAM* model has outperformed the *projected aligned SBERT-XLM-R-paraphrases* model for the French language. Though, it is expected to get better result with the large language model, XLM-R-large, while trained on an in-domain dataset, our proposed framework MUFIN has achieved comparative performance to

Table 5.20: Accuracy on MTOP dataset.

Approaches / Languages	EN	DE	FR	ES
Baseline Methods				
XLM-R-large	98.1	94.3	94.0	93.5
Aligned SBERT-XLM-R-paraphrases (NP)	95.7	90.7	90.9	92.0
Aligned DuEAM (NP)	95.2	89.8	90.1	91.4
Aligned Meta-embedding (NP)	96.2	92.7	92.4	92.9
Proposed Methods				
Aligned SBERT-XLM-R-paraphrases (UP)	96.0	91.2	91.7	92.4
Aligned DuEAM (UP)	95.8	90.8	91.1	92.6
MUFIN	**97.5**	**93.4**	**93.7**	**93.1**

Here *UP* denotes up-projection and *NP* denotes no projection.

that, highlighting the efficiency of the model. In fact it has outperformed the non projected aligned SBERT based meta-embedding model with on average 1.8% increase in accuracy.

5.3.5 Qualitative Study

Training on Additional In-domain Dataset. In real world scenarios, the quality of in-domain classification dataset is sometimes very hard to get. But, if the training dataset size is increased, the aligned sentence embedding model, if trained with additional dataset, should give better performance. To understand the efficiency of MUFIN in these scenarios, we have studied model performance after doing data up-sampling. The multilingual NLU dataset is up-sampled by 30% by copy-paste technique. We have compared our model with non-projected aligned meta-embedding. The models trained in Section 5.3.4.4 retrained with the up-sampled dataset. Observe, though there is a performance boost for the baseline model in comparison

Table 5.21: Accuracy on 30% up-sampled NLU dataset.

Model	EN	AR	ES
Aligned Meta-embedding (NP)	92.5	84.0	85.2
MUFIN	**94.1**	**85.6**	**87.2**

Here *NP* denotes no projection.

to the accuracy reported in Table 5.21, MUFIN has outperformed quite significantly having healthy performance improvement. This leads to the conclusion that small in-domain data size increment can boost the performance of MUFIN quite significantly.

Alignment on Small Training Datasize. One of the challenges for multilingual sentence embedding frameworks or language models poses is unstable training convergence, while trained on small scale in-domain datasets. Stable training of sentence embedding models, is very important for less-resourced scenarios. We have evaluated proposed MUFIN framework on sub-sampled data from Amazon review datasets and compared the performance with XLM-R-large model. We have randomly picked 2000 training samples from each classes resulting a total of 10000 training samples and tested on provided test datasets. From Table

Table 5.22: Accuracy on 10k sampled Amazon Review Dataset.

Model	EN	DE	FR	ES
XLM-R-large	46.6	44.6	42.2	43.2
MUFIN	**51.1**	**48.2**	**48.9**	**47.4**

5.22, it can be observed that MUFIN has outperformed the large language model with an average 4.5%

increase in accuracy. Thus MUFIN is more effective and easier to train with smaller datasets, giving the opportunity to implement for wide range of natural language processing tasks in industrial settings.

Until now we have discussed about sentence representation learning for low-resourced NLP applications containing of inter-sentential code-mixed sentences or monolingual sentences. But in the current era, recent advances in public communication over different social media sites have led to an increase of another type of sentence construction called intra-sentential code-mixed sentences. In the next section we will talk about learning sentence representations for the intra-sentential code-mixed sentences.

5.4 INTRA-SENTENTIAL CODE-MIXED SENTENCE REPRESENTATION LEARNING

Intra-sentential code mixing is a common phenomena in multilingual societies where people switch from one language to another in the same sentence. In some cases, the words in a code-mixed sentences can be identified by native speakers by orthographic differences. For example in Figure 5.6, the Hindi words are written in Devanagari script where as English words are in Roman scripts.

Tweets: "@gurmeetramrahim एक तेरा सहारा मिल जाए रबा दुनिया दी परवाह नहीं करना।। blessing chahiye bht sari msg dikhani h #blockbustermsg "
Translation: If I get your support @gurmeetramrahim than I won't care for anything else, I need your blessing to show the messages.

Figure 5.6: Hindi-English code-mixed tweet

Researchers have designed language identification tool to identify languages written in different scripts in a code-mixed sentence and encoded with the pretrained word vectors like aligned Fasttext embeddings (Bojanowski et al., 2017). These word embeddings are then fed into sequential deep neural networks to obtain the sentence representation. Rani et al., 2022 performed language identification using our novel language identification method introduced in Chapter 4 on word level for code-mixed Hindi-English sentences.

But quite often the code-mixed sentences are written in the same script in a sentence. We can see in Figure 5.6 the Hindi and English code-mixed sentence "blessing chahiye bht sari msg dikhani h" is written in Roman script. In this sentence the words "blessing" and "msg (message)" are English words whereas the sentence is actually a Hindi sentence. Language identification gets tougher due to the usage of abbreviation words like "msg" which is not an English dictionary word. State-of-the-art language models often do not have these kind of words in their vocabulary which reduce their efficiency for different NLP tasks. On the other hand, often the multilingual language identification models failed to identify words written in non-native scripts (in this case for English the standard language identification models are trained on Devanagari script). In this section we have introduced a novel deep learning framework which addresses these problems and generates efficient sentence embeddings for these intra-sentential code-mixed sentences.

To design intra-sentential code-mixed sentence representation learning, we have worked on one of the recent issues of understanding user sentiments from tweets. Analysis of short texts from micro-blogging platforms such as Twitter is in high demand as the analysis of these texts distill and evaluate the moods and the sentiment of the users and are very useful for different organisations, be it government, business or NGO. Sentiment analysis for Indian code-mixed languages is relatively new (Chakravarthi et al., 2020a; Chakravarthi et al., 2020b; Jose et al., 2020; Rani et al., 2020). The significant difference in style of language, orthography (Chakravarthi et al., 2019b) and grammar used in tweets presents specific challenges for English-Hindi code-mixed data. In this part, we introduce a novel deep neural network (DNN) system

English-Hindi code-mixed dataset (Patwa et al., 2020). The details of the dataset are described later (refer to 5.4.1. We also compare the system with other state-of-the-art systems and describe how the system has outperformed others. During the training process the absence of word level language tags highlights the novelty of this work for generating sentence representations.

5.4.1 Dataset

The provided dataset consists of English-Hindi code-mixed tweets annotated with sentiment labels: positive, negative, or neutral. Besides the sentiment labels the dataset also includes *mixed*, and *univ* (symbols, @ mentions, hashtags). As it is very common for Twitter data to have other forms of text such as URLs and emoticons, this data-set too contains emojis such as ☺ ☹ and URLs. The distribution of labels and sentences for the train and development set can be found in Table 5.23. In the experiments, the model has been trained with the training set and validated with the development set. Both the train and development sets contain higher number of neutral tweets than positive and negative.

Data-set	Neutral	Negative	Positive
Train	5264	4102	4634
Dev	1128	890	982

Table 5.23: Data-set distribution

Three different data processing pipeline were used as we have trained three different models for the experiment. We normalize the data for training the Support Vector Machine (SVM) and deep neural network (DNN), by lower-casing all the tweets and removing punctuation, emojis and URLs. After converting all the tweets into lower case, extra spaces were removed from the tweets. The tweets are tokenized into characters, where each character has been mapped to an index number. The character-index mapping is created with the help of the Keras tokenizer package[4]. As the tweets are of different lengths, we have padded each tweets with maximum length of the sequence using keras padsequences function[5]. We have assigned a new numeric value to each of the sentiment labels for our experiments. **Neutral**, **Negative** and **Positive** has been converted to 0, 1 and 2 consecutively.

5.4.2 System Description

We trained three different models including our own proposed deep learning model. The first experiment started with linear Support Vector Machine (SVM). For our second experiment, we used a state-of-the-art character-level Convolution Neural Network (CNN) model for sentence classification (X. Zhang et al., 2015). The third experiment was carried out using our own proposed GenMA Model, which has outperformed the other two models on the data-set.

4 https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer
5 https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/sequence/pad_sequences

5.4.2.1 *Support Vector Machine*

SVM is an algorithm which maximizes a particular mathematical function with respect to a given collection of data (Noble, 2006). In our experiment, we have focused on the linear SVM methodology. The objective of linear SVM optimization problems is to maximize the given equation:

$$max_{\alpha} \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j (x_i x_j) \tag{5.5}$$

where α_i is the weight of the examples, x is the input and y is the label. After pre-processing the data, we experimented with the most basic input feature TF-IDF, which was created with the help of TfidfVectorizer[6] of the Scikit Learn package. The minimum document frequency for the vocabulary building has been set to 3 and maximum document frequency has been set to 8.

5.4.2.2 *Convolution Neural Network (CNN)*

In this experiment, we have followed the CNN model described by X. Zhang et al., 2015, which has a one-character embedding layer and four convolution (CONV1D) layers. For the first three convolution layers, one max-pooling layer has been added, after each convolution layer. The final convolution layer is followed by one hidden layer, which in turn is followed by one softmax layer. The model accepts sentences as sequence, and characters as input. The character embedding is a one-hot embedding (1-to-n embedding) where the number of unique characters is n. The shape of the filter is one-dimensional in size k. The filter slides over the input sequence matrix to create the feature map of dimension $b \times f \times s$ where b is the batch size, f is the number of filters used, and s is determined by the formula $m - k + 1$ where m is the input size. Stride 1 is used to calculate features based on each character including spaces and special characters.

5.4.2.3 *Generative Morphemes with Attention (GenMA) Model*

We propose an artificial morphemes generative system with Self Attention (SA) layer. The model takes the input sequence as a character sequence. The model has one character-embedding layer and two convolution (CONV1D) layers. Each convolution layer has one max-pooling layer. After the convolution layers, there is one bidirectional LSTM layer, followed by one self-attention layer. The model has one hidden layer and one softmax layer. Z. Liu et al., 2018 designed a network which is based on the Chinese character attention model for sentence classification. We have taken inspiration from that model. Z. Liu et al., 2018 rely on capturing the local context of sentences based on a single convolution layer whereas GenMA is capable of generating new artificial morphemes and framing a sentence as a group of new morphemes irrespective of the language identification of source words. The combination of two CNN layers helps to generate new morphemes based on deep relative co-occurring characters (3 characters frame), and the LSTM layer helps to capture the global information of sentences based on newly generated features. The SA layer helps to construct sentence-level information. It also captures relativity strength among different co-occurring character features. The model architecture can be found in Figure 5.7.

A Convolution Neural Network layer is used as a feature extractor of the sentences. The one-dimensional convolution implements one-dimensional filters which slides over the sentences as a feature extractor. Let the filters have a shape of $1 \times k$ where k is the filter size. Let $x_i \in \{0,1\}^n$ denote the one-hot representation of the i-th character considering character vocabulary size is n. For each position j in the

6 https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

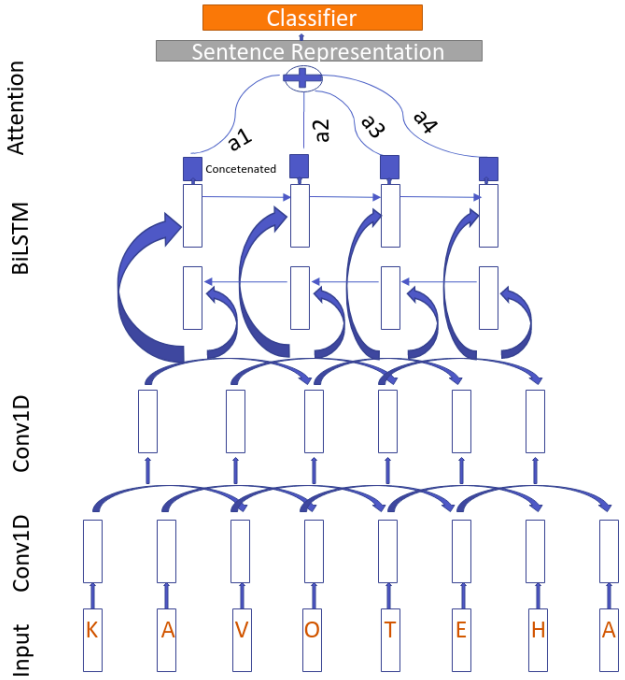

Figure 5.7: Relative Character Attention Model; the input is the character embeddings which is mentioned with physical characters; boxes filled with blue colour are the concatenated outputs of the BiLSTM.

sentence, a window vector w_j is developed with k consecutive character vectors (Zhou et al., 2015) denoted as

$$w_j = [x_j, x_{j+1}, ..., x_{j+k-1}] \tag{5.6}$$

The one-dimensional k-sized filters slide over the window vector w_j to create the feature map s where $s \in \mathbb{R}^{m-k+1}$, and where m is the input size. Multiple filters are used to generate different feature maps for each window w_j. The resulting new feature representation, W_j, represents a new feature map vector for the j-th position of the sentence. The second convolution layer will take feature representations as input and generate a high-order feature representation of the characters. The max-pooling network after each convolution network helps to capture the most important features of size d. The new high-order representations are then fed to the LSTM (Long Short Term Memory Network) as input.

Long Short Term Memory (LSTM) Network layer takes the output of the previous CNN layer as input. The LSTM layer produces a new representation sequences in the form of $h_1, h_2, ...h_n$ where h_t is the hidden state of the LSTM of time step t, summarising all the information of the input features (morphemes) of the sentences. An LSTM unit is composed of one memory cell and three gates (input gate, forget gate and output gate) (Hochreiter & Schmidhuber, 1997b). At each time step t, the hidden state takes the previous time step hidden state h_{t-1} and characters (x_t) as input. Let us denote memory cell, input gate, forget gate and output gate as c_t,i_t,f_t,o_t. The output hidden state h_t and the memory cell c_t of timestep t are defined by Equation 5.7

$$\begin{aligned} i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad , \quad f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad , \quad f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ c_t = f_t \odot c_{t-1} + i_t \odot q_t \quad , \quad h_t = o_t \odot \tanh(c_t) \end{aligned} \tag{5.7}$$

Here $\odot$ is the element-wise multiplication operation; W_i,W_f,W_o,W_q are the weights of the matrices; b_i,b_f,b_o,b_q are the biases; and σ denotes the logistic sigmoid function. A Bidirectional LSTM (BiLSTM) network has been used, which has helped us to summarise the information of the features from both directions. The Bidirectional LSTM consists of a forward pass and a backward pass which provides two annotations of the hidden state h_{for} and h_{back}. We obtained the final hidden state representation by concatenating both hidden states $h_i = h_{i-for} \oplus h_{i-back}$, where h_i is the hidden state of the *i*-th timestep and $\oplus$ is the element-wise sum between the matrices.

The Attention layer helps to determine the importance of one morpheme over others while building sentence embeddings for classification. The LSTM layer will have the hidden states containing the information of time step *t* and its previous time steps. But all morphemes are not equally important while building a sentence (taking into consideration that sentence is build by sequence of characters including space and special characters). A self-attention mechanism has been adopted from Baziotis et al., 2018, which will help to identify the morphemes that are important for capturing the sentiment of the sentence. The self-attention mechanism is built on the attention mechanism developed by Bahdanau et al., 2015. The mechanism assigns weight a_i to each feature's annotation based on output h_i of the LSTM's hidden states, with the help of the softmax function as illustrated in Equation 5.8

$$a_i = \tanh(W_h \cdot h_i + b_h) \quad , \quad a_i = \frac{\exp(a_i)}{\sum_{t=1}^{T} \exp(a_t))} \tag{5.8}$$

The new representation will give a fixed representation of the sentence by taking the weighted sum of all feature-label annotations as shown in Equation 5.9

$$r = \sum_{i=1}^{T} a_i \cdot h_i \tag{5.9}$$

where W_h and b_h are the attention weights and bias respectively.

The Output layer consists of one fully-connected layer with one softmax layer. The sentence representation after the attention layer is the input for the dense layer. The output of the dense layer is the input of the softmax which gives the probability distribution of all the classes with the help of the softmax function as shown in Equation 5.10

$$p_i = \frac{\exp(a_i)}{\sum_{t=1}^{T} \exp(a_t)} \tag{5.10}$$

where a_i is the output of the dense layer.

5.4.3 Results

Overall, the classifiers exhibit varying performance, with some performing much better out-of-sample than others. Table 5.24 shows the class-wise macro F1-score of different models on the test-set.

Model	Pos Class	Neg Class	Neut Class	Score
SVM	0.64	0.62	0.57	0.61
Char-CNN	0.68	0.65	0.56	0.63
GenMA	0.73	0.67	0.63	**0.68**

Table 5.24: F1-scores of three algorithms on dataset

The state-of-the-art character CNN model has performed better than the SVM model. One of the main reasons for this is that a CNN is capable of identifying the features of the sentence with the help of a neural

model weight distribution. It also takes special characters into account which make the sentence embedding more robust to work on. On the other hand, the hyper-tuning settings of the TF-IDF vectors could be the cause of the lower performance of the SVM.

Our GenMA model has outperformed all classical models as well as the state-of-the-art character CNN model as it considers a sentence composed of a different set of morphemes. The individual results on three different sentiment classes show that the model outperforms the other two models while recognizing individual classes, whereas the SVM model recognizes neutral classes better than the CNN model. Our model has achieved 0.68 F1-score in the test set which is seven percent better than the SVM and five percent better than the character CNN model.

5.4.4 Discussion

The proposed GenMA model outperforms other models as it is capable of generating new morphemes out of neighboring characters and it identifies the essential morphemes to classify a sentence. The two main advantages of our model are:

- The model can construct sentence embeddings based on the new generative morphemes which are created artificially in a combination of both Hindi and English. These morphemes carry the features of both languages. As illustrated in figure 5.8, the new morpheme **avo** generated by the model (this is weighted .021 by the model). Here the character "a" belongs to the Hindi word "Ka" and the character sequence "vo" belongs to the English word "vote". This shows that these new artificial generative morphemes have features of both Hindi and English. Thus, multilingual word-level language identification annotations are not required.

- The model is able to correctly identify the co-occurring character sets with highest importance in terms of sentiment analysis. The attention mechanism is visualized in Figure 5.8. The red characters

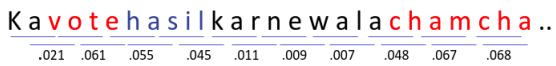

Figure 5.8: Character of tweets (English-Hindi) with attention scores

are the most important characters for sentence classification. The blue characters also contribute significantly; the black characters contribute least significantly to the sentence classification. Some of the generated artificial morphemes put more emphasis on sentence polarity (An example is morpheme "ote", which is weighted approximately five times (0.061) higher than the normal morpheme "arn" (0.011)). The model enables softmax attention weights to rank characters high to low.

5.5 CHAPTER WRAP-UP

In this chapter, we have discussed efficient sentence representation learning frameworks for inter-sentential as well as intra-sentential code-mixed sentences. Our approaches include capturing better semantic representation of sentences which enhances the efficacy of sentence encoders for downstream NLP tasks.

First we proposed an *unsupervised loss function based DuEAM framework* for *multilingual sentence embeddings* based on dual encoder with anchor-learner model via multi-task learning. The joint loss function coupling word mover's distance and cosine similarity to capture the degree of text similarity and relatedness between sentence pairs. This enhances the ability of understanding the syntactic and semantic meaning

of languages, producing better results for low-resourced languages. Our experiments on monolingual and cross-lingual benchmarks showcase the efficacy of our sentence embeddings in capturing semantic similarities across 58 languages. We demonstrate that *DuEAM* significantly outperforms existing unsupervised models for textual similarity understanding. We also depicts robustness in *zero-shot learning* and *limited training*, for catering to under-resourced languages, and achieve results better or comparable to existing supervised methods in certain cases.

Secondly, our proposed *Meta Embedding with Up-Projection and Fine-Tuning* (MUFIN) framework learns enhanced sentence representation geared towards text classification applications. Experiments on cross-lingual benchmarks demonstrates the efficiency of the model while aligned with low-resourced in-domain datasets. We demonstrate that the MUFIN framework successfully outperforms the state-of-the-art multilingual sentence embedding models. Moreover, it achieves comparative performance to large language model *XLM-R-large* while doing different classification with less memory and time to train. We also depict robustness in limited training, for catering to limited in-domain training datasets and achieve better results to large language models.

Lastly, we made steps towards designing an efficient sentence encoder which can handle intra-sentential code-mixed datasets. Our motivation is to encode the code-mixed sentences, delivering better results to downstream NLP tasks. Our proposed novel deep neural model *GenMA* has outperformed the state-of-the-art models on code-mixed data proposed in Patwa et al., 2020 as discussed in Section 5.4.3. Our model is capable of classifying the sentiment of the sentences without considering language difference between words in the sentences with an F1-score of 0.68 on the test data. This reduces the need of gathering orthographic language-dependent knowledge, which opens up the chances of deploying this system widely.

6 IDENTIFYING COGNATES FOR CLOSELY-RELATED LANGUAGES

6.1 INTRODUCTION

In the previous two chapters, we discussed our contribution towards research questions **RQ1** and **RQ2**. In chapter 4 and 5 we have discussed novel methods for getting sentence representations for both low-resourced languages and low-resourced NLP applications. Our discussion of the novel language identification method enhances the possibility of getting sufficient training datasets for different languages to train unsupervised cross-lingual sentence embedding frameworks on different corpora. But often the training datasets in different scenarios are created using automatic machine translation tools (we obtained machine-translated NLU datasets in Section 5.3.4.4). Thus the generation of sentence embeddings is dependent on the quality of these datasets. Often these cross-lingual sentences consist of words having similar meaning and spelling called "cognates", which influence the quality of sentence embeddings. In this chapter, we will introduce a novel unsupervised cognate detection methodology which exploits the understanding of grammatically related word representations between closely-related languages and we will study whether this enhances the sentence representation learning for downstream NLP tasks.

Cognates are etymologically related word pairs across languages (Crystal, 2011). However, cognates are defined in much broader terms in many different fields, including natural language processing (NLP) or psycholinguistics (Labat & Lefever, 2019). In these areas, word pairs with similar meanings and spelling are also considered cognates. In the recent development of automatic machine translation (AMT), automatic cognate detection is found to be a very effective aid for similar language translation (Kondrak, 2005; Kondrak et al., 2003). Moreover, it helps to efficiently perform cross-lingual information retrieval (Makin et al., 2007; Meng et al., 2001) from different sources. Very often, words that have similar spelling are recognised as cognates (Example: the Latin and the English word pair "cultūra" and "culture"). Automatic machine-translated sentences also consist of cognates which on the contextual word level convey the semantic meaning of the sentences (Example: the English sentence "he met an accident" will be translated to Spanish sentences "se encontró con un accidente" where "accident" and "accidente" are cognates). Nevertheless, there are word pairs which are false friends or partial cognates (Kanojia et al., 2020c). Partial cognates are similar words across languages but carry different meanings in different contexts (Kanojia et al., 2019b), thus making automatic cognate detection hard and challenging. Identifying these cognates requires extensive linguistic knowledge across languages, which is quite hard and expensive to annotate. While cross-lingual automatic cognate detection systems (Kanojia et al., 2020a) exist, they have primarily been supervised methods requiring labelled data or language-specific linguistic rules. For under-resourced languages, finding annotators or linguists is a challenging task. This highlights the need for an efficient unsupervised language-agnostic cognate detection framework. Our approach involves designing the framework with the help of grammatical word representation knowledge of the languages.

In this chapter we aim to answer our third research question:

- **RQ3.** Does effective cross-lingual word representations learning improve unsupervised cognate detection without any language information for closely related and low-resourced languages? Having detected cognates, does this assist in producing better sentence embeddings for NLP applications?

We will address the first part of the research question first:

- **RQ3.1.** Does effective cross-lingual word representations learning improve unsupervised cognate detection without any language information for closely related and low-resourced languages?

Cross-lingual or multi-lingual word representation commonly known as multilingual word embeddings represent words from diverse languages into the same shared vector space. Researchers have proposed different word embedding techniques over the year starting from static word embeddings like the GloVe word embeddings (Pennington et al., 2014) to contextual word embedding techniques like the XLM-Roberta model (Conneau et al., 2020). Often two closely-related languages have many structural and grammatical similarities in the words. The traditional character-based word embedding techniques help to highlight the orthographic similarities between the languages, resulting in a cross-lingual orthographic aligned shared vector space (Severini et al., 2020). But this technique is not very efficient when there are orthographic differences. Moreover, it fails to capture the grammatical relatedness between languages of the same language families.

On the other hand, it can be argued that word embeddings infused with morphological knowledge of languages are mapped to related words efficiently in a shared vector space (Cotterell & Schütze, 2015). Morphological learning helps to capture the internal structures of a word of a language. Thus learning of morphological word representations helps to form a one-to-one mapping between the closely-related languages. As cognates are etymologically related word pairs, we believe infusion of deep morphological knowledge in a cognate detection framework can boost the efficiency of the model. We show that our unsupervised approach can better exploit the available data than existing supervised methods and thus produce better results for under-resourced languages. The method transfers the morphological knowledge of a shared encoder in the unsupervised cognate detection framework into a Siamese network setting, where the framework simultaneously learns word representation and cluster assignments in a self-learning setup.

The supervised cognate detection framework understands the relationship between word pairs by either concentrating on their phonetic similarity or lexical similarities based on their annotated positive or negative labels (Jäger et al., 2017; Rama, 2016). Recently, researchers tried to exploit contextual multilingual word embedding techniques to identify cognates which produced better results than only concentrating on phonetic transcriptions (Kanojia et al., 2020a). Although such methods produced good results, for most of the languages the annotation is quite expensive and tedious. Moreover, producing multilingual contextual word embeddings is a challenging task for these languages due to fewer data sources on the web. Merlo and Rodriguez, 2019a highlighted that based on bilingual lexicon matching between two known languages, the similarity score produced by contextual word embeddings could differentiate between true or false cognate pairs. Though this technique is label independent, the framework is very much dependent on the bilingual lexicon availability of the known language pairs.

To alleviate the above challenges, in this chapter, we propose a *language-agnostic unsupervised cognate detection framework based on Siamese architecture* with an iterative clustering approach (J. Xie et al., 2016) during back-propagation. Our encoder design is inspired by our proposed methodology in chapter 4, where we learnt the n-gram character features of a sentence with attention. We introduce a *positional encoder* on n-gram features which, in combination with the attention mechanism, learns sub-words representation of a word. We also depict the performance of our *morphological knowledge-based unsupervised framework*. This variant gives a better understanding of the grammar and structural analysis of the words of a language.

Thus transferring this knowledge with the help of a shared encoder to closely-related languages enhances the understanding of structural and grammatical relatedness between cross-lingual word pairs. Moreover, our word encoding method helps to produce better supervised cognate detection results, which outperform the state-of-the-art supervised results on various language pairs. We present both supervised and unsupervised results in this chapter.

The extensive experiments on three different cognate detection datasets across language families have showcased the efficacy of our unsupervised and supervised cognate detection framework. For example, on six different Indian language pairs, our unsupervised model (with morphological knowledge) has outperformed the state-of-the-art supervised model proposed by Kanojia et al., 2020a by an average of **9** points of F-score whereas for Celtic language pairs, it outperformed by **8.6** points of F-score. At the same time, our supervised framework has produced a state-of-the-art performance by outperforming the existing supervised model by an average of **16** points of F-score. Thus, our model is robust across diverse language families for the supervised and unsupervised cognate detection task. Interestingly, our experiments show that on Indian language pairs like Hindi-Punjabi and Hindi-Marathi, an encoder with morphological knowledge of the Hindi language performed better than an encoder with morphological knowledge of their ancestral language Sanskrit by an average of **1.5** points of F-score. However, the performance of the unsupervised framework for the language pair Marathi-Bengali has improved by **2** points of F-score.

To this end, we propose:

- A language-agnostic unsupervised cognate detection framework without the need for labels;
- Efficiently transferring morphological knowledge of a low-resourced language to closely-related under-resourced languages with or without the need for the pivot language for better cognate detection;
- Introduction of positional embedding along with attention to different n-grams of a word for better understanding of word structures;
- Robustness in unsupervised and supervised cognate detection for low-resource languages across three different datasets of different language families, outperforming state-of-the-art supervised approaches.

6.2 COGNATE DETECTION FRAMEWORK

In this section, we describe the components and working of the proposed *Language Agnostic Unsupervised Cognate Detection Architecture Using Morphological Knowledge*, trained using an unsupervised loss function and iterative clustering method. The iterative clustering process enhances the understanding of word representation and cluster assignment.

Figure 6.1 depicts the shared word encoder based framework. It consists of two parts - **(i)** a *Morphology Learner*, which gathers the morphological knowledge of a language and **(ii)** an *Unsupervised Cognate Detector*, which uses the morphological knowledge learnt using a shared word encoder to cluster the cognates between word pairs.

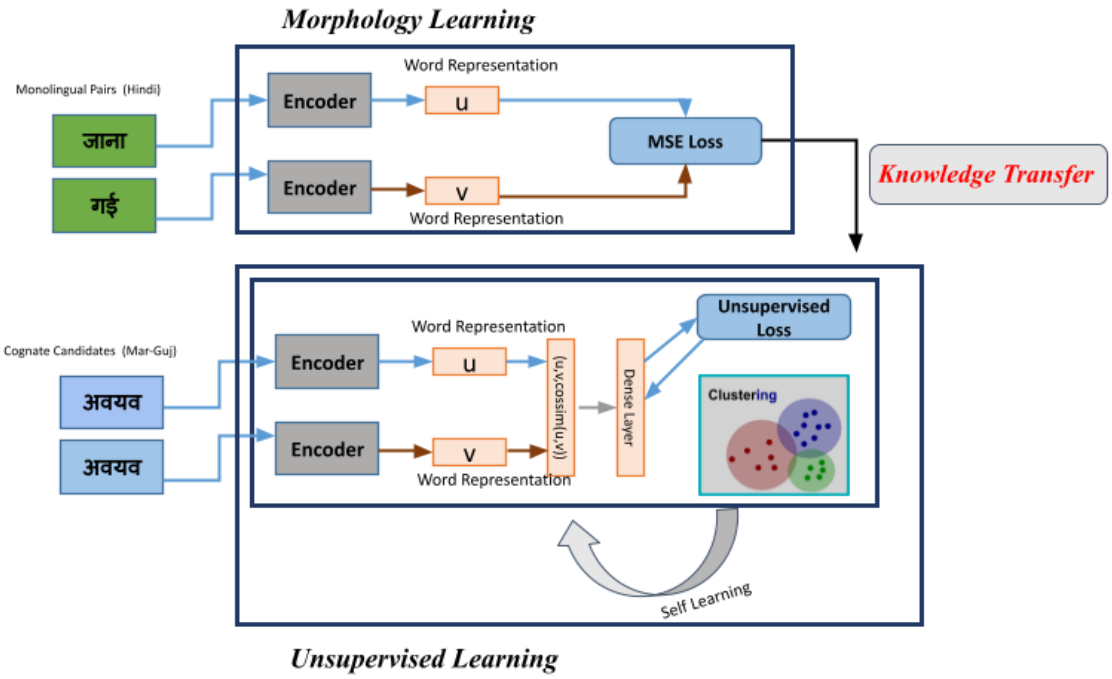

Figure 6.1: Unsupervised Cognate Detection Framework with Morphology Learner and Unsupervised Cognate Detector. For training we pass monolingual word-pairs into morphology learner (coloured in green) and bilingual cognate candidates (coloured in blue) into unsupervised cognate detector.

6.2.1 Word Encoder

The word encoder consists of n-gram character-level CNN and a positional embedding layer, followed by a self-attention layer.

Character Encoding. The character level CNN layer generates word representations on n-gram ($n \in \{2,3,4,5,6\}$) characters, which helps to understand the representation of words on different subword level (Goswami et al., 2020). The different subword level representations of a word are achieved using a 1-dimensional CNNs (X. Zhang et al., 2015). The characters of a word are fed as input sequence $S = [w_1, w_2, ..., w_m]$ to the 1-dimensional CNN, where m is the number of characters present in the word and w_i is a character in the word. Considering the 1-dimensional CNN as a feature extractor, it slides over characters to create a window vector w_j with consecutive character vectors, as denoted in Equation 6.1.

$$w_j = [x_j, x_{j+1}, ..., x_{j+k-1}] \tag{6.1}$$

where k is the size of the feature extractor filter and $x_i \in \mathbb{R}^d$ is the d-dimensional character embedding of the i-th character, where character vocabulary size is n. Thus this k-sized filters create the the feature map s ($s \in \mathbb{R}^{m-k+1}$) from the window vector w_j according to Equation 6.2,

$$s_j = a(w_j \cdot m + b_j) \tag{6.2}$$

where vector m is a filter for convolution operation, b_j is the bias for the j-th position and a is the non-linear function. Thus the new feature representation of the word F ($F \in \mathbb{R}^{(m-k+1)\times n}$) will be expressed as $F = [s_1, s_2, ..., s_n]$, where n is the number of filters and m is the total input size. Observe, the different filter size $k \in \{2,3,4,5,6\}$ represents the n-gram features of the word, which will be represented as F_k.

Positional Encoding of Features. While getting n-gram features of the word representation, the character sequence order carries a significant role in word construction. We try to learn the different n-gram positions in a word with the help of our new introduction of positional encoding. Transformer architectures (Vaswani et al., 2017a) enforce the trainable positional embedding on the input word pieces to understand the position of the words in a sentence. In our input words the same character can appear in multiple positions, which makes positional embedding on each character irrelevant. Rather, our approach of learning the n-gram sequence in a word helps to understand the grammatical and morphological differences from the structural perspective of a word. Following the learning process of the positional embedding of transformer architecture, we encode the trainable positional encoding of different n-gram features to F_k according to Equation 6.3

$$\begin{aligned} pe_{i,2k} &= \sin(i/1000^{2k/d}) \\ pe_{i,2k+1} &= \cos(i/1000^{2k+1/d}) \end{aligned} \tag{6.3}$$

where d is the dimension of the output in the embedding space and k is the position of an object in input sequence.

Attention of Features. The new encoded feature representation F_k produces the ultimate word embedding which is achieved by giving weight to n-grams according to their importance in word construction. A self-attention mechanism takes the feature representation as input and produces output weight vector a for every feature representation F using the following Equation 6.4

$$a = \text{softmax}(\tanh(W_h \cdot F^T + b_h)) \tag{6.4}$$

The summation of feature representation F according to the weight vectors provided by a generates vector representation r of a word by Equation 6.5

$$r = \sum_{i=1}^{T} a_i \cdot F_i \tag{6.5}$$

where a_i is the attention weights, and $\cdot$ is the element-wise product between elements. The final vector representation r is the concatenation of different n-grams $\in \{2,3,4,5,6\}$, which is represented as $r = [r_2, r_3, r_4, r_5, r_6]$.

6.2.2 Morphology learner

We learn the morphological relationship between two words r_i and r_j of the same language in a Siamese setting (details of the morphological training dataset building with an example is given in Section 6.3). The encoded vector representations then passed through a fully connected (FC) layer which gives two vector representations $z_l \in \mathbb{R}^{N \times K}$ and $z_r \in \mathbb{R}^{N \times K}$. The morphological learner model is then trained to minimize the mean-squared loss between two word representations such that their vector space distances reflect their degree of morphological relatedness in Equation 6.6

$$1/N \sum_{i=1}^{N} (z_{l_i} - z_{r_i})^2 \tag{6.6}$$

where N is the mini-batch size.

6.2.3 Unsupervised Cognate Detector

The encoder with morphological knowledge accepts two word representations r_i and r_j from two different languages as input in a Siamese setting. The encoded vector representations then passed through a fully connected (FC) layer which gives two vector representations $u \in \mathbb{R}^{N\times K}$ and $v \in \mathbb{R}^{N\times K}$. We concatenate the word representations u and v with their cosine similarity score and pass it through a sense layer to achieve the combined representation $z \in \mathbb{R}^{N\times K}$. It is then passed through a softmax layer to get the probability distribution of all classes $p \in \mathbb{R}^{N\times K}$, as per Equation 6.7

$$p_{ij} = \frac{\exp(z_{ij})}{\sum_{t=1}^{K} \exp(z_{ik})} \tag{6.7}$$

where k is the number of classes. We train the unsupervised model based on the maximum likelihood clustering loss proposed in Chapter 4, where we try to maximize the probability distribution function for each class and at the same time try to minimize the probability of getting on all the datasets assigned to one class using Equation 6.8.

$$L_u = \sum_{i=1}^{N} \max_{j=1}^{i} p_{ij} - \max_{i=1}^{N} \sum_{j=1}^{i} p_{ij}^2 \tag{6.8}$$

While the unsupervised loss function helps us to get word embeddings and an initial cluster assignment, it is important to improve cluster purity according to datasets. J. Xie et al., 2016 proposed a self learning based deep clustering technique. The framework learns the clustering based on stochastic gradient descent (SGD) during backpropagation. We finetune our word embedding to learn better clustering using this iterative clustering technique. The initial sets of cluster centroids $u_j{}_{j=1}^{k}$ is obtained from the pre-training phase using Equation 6.8. In this self training phase, we assign word embeddings to initial cluster centroid and then fine-tune the word embeddings and cluster centroids using auxiliary target distribution.

The assignment of cluster centroids (u_j) and word embeddings (z_i) are calculated based on Student's t-distribution (Maaten & Hinton, 2008) as per Equation 6.9

$$q_{ij} = \frac{\left(1 + \|z_i - u_j\|^2\right)^{-1}}{\sum_{j'} \left(1 + \|z_i - u_{j'}\|^2\right)^{-1}} \tag{6.9}$$

where q_{ij} is the probability of sample i to cluster j assignment.

We now refine the cluster learning from their high confidence assignments using auxiliary target distribution p_{ij} (J. Xie et al., 2016). This helps to improve cluster purity putting more emphasis on data point assignment, as per Equation 6.10

$$p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_{j'} \left(q_{ij}^2 / \sum_i q_{ij'}\right)} \tag{6.10}$$

where $\sum_i q_{ij}$ is the frequency of clusters.

The cluster assignment self-learning process is trained based on KL divergence loss between assignments q_i and p_i, as shown in Equation 6.11.

$$L = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{6.11}$$

Word1	Word2
nuachtán.	nuachtáin
eolas.	a eolais.
síceolaí.	leis an síceolaí.
Críostaí.	Críostaithe.

Table 6.1: Morphology Learning Dataset for Irish Language

Word1 (Hindi)	Word2 (Marathi)	Label
Ghrina [Hate]	Ghrina [Hate]	Positive
Prerit [Inspired]	Bahan [Vehicle]	Negative
Akshyam [Unable]	Vyasana [Addiction]	Negative
Akashbani [Predict]	Akashbani [Predict]	Positive

Table 6.2: Cognate Detection Dataset for Hindi-Marathi Language pairs

6.3 TRAINING DATASET

The framework has two parts: **(i)** Morphology Learner and **(ii)** Unsupervised Cognate Detector.

Morphological training is based on UniMorph (McCarthy et al., 2020) datasets. As shown in Table 6.1, our Siamese network accepts two words as input. The inputs are the monolingual word pairs of the pivot language in UniMorph data (as shown in Figure 6.1, we train the encoder of the morphological learner on the Hindi dataset in a supervised manner and transfer the knowledge to the unsupervised cognate detector for Marathi-Gujrati word pairs). Though our model is trained with the supervised dataset from UniMorph, we do not consider the annotated morphological class while training the morphological learner. The statistics of datasets for three pivot languages and Sanskrit are given in Table 6.3.

Cognate Detection Task We have evaluated our models on three different datasets for the cognate detection task: Indian, Celtic, and South African languages. For under-resourced Indian language pairs, we have followed the work of Kanojia et al., 2020c. For South African and Celtic languages, as there are no easily available datasets for the cognate detection task, we have built the dataset from the open-source cognate database *CogNet* (Batsuren et al., 2019). The true cognates are directly taken from the dataset and false cognate pairs are randomly shuffled word-pairs with a 60-40 split of the total dataset available in the database for each language pair. We experiment with both supervised and unsupervised learning of these cognate classifiers, based on the encoder that was learned in the previous step. During the training procedure of the unsupervised cognate detector, no word pair labels of cognate datasets are considered. An example of the dataset can be found in Table 6.2.

6.4 EXPERIMENTAL EVALUATION

We evaluated our framework on three different datasets in three different scenarios: (a) Language pairs with pivot language and its morphological knowledge, (b) Language pairs without the pivot language but with the shared encoder having morphological knowledge of the pivot language and (c) The effect of the historical language morphological knowledge transfer on the language pairs.

We compare our models with the following supervised state-of-the-art cognate detection frameworks: (i) **CNN based model** Siamese CNN based approach by Rama, 2016, (ii) **Orthographic similarity** based approach from Labat and Lefever, 2019, (iii) **Recurrent Neural Network** based approach proposed by

Dataset	Hindi	Irish	Zulu	Sanskrit
Training Dataset.	42200	2579	49696	437675

Table 6.3: Morphological Training Dataset of Three Pivot Languages and Sanskrit

Kanojia et al., 2019b, (iv) **Contextual Word Embedding** based approach with XLM-R (Conneau et al., 2020) proposed by Kanojia et al., 2020a.

6.4.1 Pivot Language based Cognate Detection

Understanding word representation is the key to the state-of-the-art deep learning frameworks for different cross-lingual cognate detection tasks. Supervised models rely on distributional learning based on the annotated labels. In contrast, the unsupervised frameworks should be able to learn the structural and syntactical representations of words to do clustering. We have evaluated our model on different language families including Indo-Aryan, Dravidian (Kanojia et al., 2020c), Celtic and South-African languages.

Approaches / Languages	Hi-Mr	Hi-Gu	Hi-Pa	Hi-Bn	Hi-Ta	Hi-As
Orthographic Similarity	0.21	0.23	0.21	0.36	0.20	0.34
(Rama, 2016)	0.69	0.67	0.47	0.65	0.53	0.71
(Kanojia et al., 2019b)	0.72	0.76	0.74	0.68	0.53	0.71
XLM-R + FFNN	0.73	0.76	0.73	0.78	0.56	0.71
Proposed Supervised Methods						
Proposed-method	0.81	0.79	0.80	0.79	0.69	0.78
Proposed-method$_{withknowledge}$	**0.91**	**0.87**	**0.88**	**0.86**	**0.77**	**0.82**
Proposed Unsupervised methods						
Proposed-method	0.72	0.73	0.74	0.75	0.67	0.69
Proposed-method$_{withknowledge}$	**0.85**	**0.84**	**0.81**	**0.82**	**0.74**	**0.79**

Table 6.4: Results of supervised and unsupervised cognate detection task based on F-Score for Indian languages. The baseline performances are as reported in Kanojia et al., 2020a

As shown in Table 6.4, we have evaluated our model on six different language pairs. Our baseline model is based on the orthographic similarity approach. As expected, it does not perform well (Example: Word pair "Alankar (Ornament)" in Hindi, and "Alankaaram (Ornament)" in Tamil with similar word structures classified wrongly). The character-based CNN method proposed by Rama, 2016 followed by the recurrent network-based solution proposed by Kanojia et al., 2019b increases the model efficiency by quite a margin while detecting cognates. The contextual word embedding XLM-R based baseline model gives the best score compared to the previous models' score. This model is capable of injecting contextual knowledge of words from a sentence. Our proposed approach has outperformed all of these baseline frameworks and achieved state-of-the-art results for supervised and unsupervised frameworks. For the language pairs Hindi-Marathi, as they are very closely related, transferring the learnt Hindi morphological knowledge has increased the efficiency of the model by **18** points of F-score. We observed that our unsupervised framework with morphological knowledge outperformed the state-of-the-art supervised baseline system by **13** points of F-score. It is interesting to note that our supervised and unsupervised frameworks achieved state-of-the-art results by outperforming the baseline systems by **21** and **18** points of F- score, respectively, for the language pair Hindi-Tamil. Hindi and Tamil come from different language families, Indo-Aryan and Dravidian, significantly boosting the efficacy of the cognate detection framework. On average, our supervised and unsupervised system has achieved an improvement of **16.8** and **9** points of F- score, respectively, on Indian language pairs.

Approaches / Languages	Zu-Ss	Zu-Xh	Ga-Gd	Ga-Gv
Orthographic Similarity	0.21	0.31	0.29	0.22
(Rama, 2016)	0.24	0.61	0.64	0.59
(Kanojia et al., 2019b)	0.23	0.74	0.72	0.61
Proposed Supervised Methods				
Proposed-method	0.65	0.76	0.77	0.69
Proposed-method$_{withknowledge}$	**0.72**	**0.87**	**0.88**	**0.74**
Proposed Unsupervised methods				
Proposed-method	0.69	0.73	0.71	0.62
Proposed-method$_{withknowledge}$	**0.78**	**0.79**	**0.81**	**0.71**

Table 6.5: Results of supervised and unsupervised cognate detection task based on F-Score for South African and Celtic languages.

Table 6.5 shows the performances of the South African and Celtic language pairs. For South African languages, we have Zulu (Zu), Swati (Ss) and Xhosa (Xh). Irish (Ga), Manx (Gv), and Scottish Gaelic (Gd) are part of the Celtic languages. We transferred morphological knowledge of Zulu and Irish to other South African and Celtic languages, respectively. Our proposed supervised and unsupervised framework outperformed the state-of-the-art baseline models. We reported poor performance of the baseline models for the language pairs Zulu and Swati. Due to very few training datasets availability of Zulu-Swati (only 23 cognate pairs are available), the models cannot be trained effectively. On the other hand, our proposed approaches performed better than the baseline models by a large margin and very interestingly, our unsupervised model is better than the supervised model by **6** points of F-score. Relatively complex word pairs such as "umgqibelo (Saturday)" in Zulu, and "úm-gcibélo (Saturday)" in Swati are correctly identified as cognate pairs.

These results show that the proposed cognate detection framework can efficiently detect cognates across language pairs with the morphological knowledge of the pivot language. Moreover, with fewer training data, the proposed unsupervised framework with or without morphological knowledge is an efficient solution for cognate detection.

6.4.2 Absence of Pivot Language

We now evaluate the robustness of our transfer learning approach on the cross-lingual language pairs in the absence of the pivot language. For Indian language pairs, our pivot language is Hindi (Hi), whereas for the Celtic language pairs pivot language is Irish (Ga). From Table 6.6, we observe that the transfer learning approach is still very efficient when the pivot language is absent. As we can see, for the language pairs Gd-Gv, without knowledge transfer in supervised learning, the recurrent neural network approach by Kanojia et al., 2019b is better than our approach. However, with the morphological knowledge encoded, for both supervised and unsupervised methods, our model outperforms by **11** and **9** points of F-score, respectively. On average, our transferred knowledge-based unsupervised method has outperformed the baseline method by **8.6** points of F-score. Thus, we can see a steady performance across all language pairs, showing the stability of the proposed morphological knowledge transfer supervised and unsupervised framework.

6.4.3 Knowledge of Historical Language

In this section, we will discuss the effect of transferring knowledge from the historical language Sanskrit. In terms of cognate detection, using Sanskrit as the pivot language for transfer learning is well motivated

Approaches / Languages	Mr-Gu	Pa-Gu	Mr-Pa	Mr-Bn	Gd-Gv
Orthographic Similarity	0.22	0.26	0.21	0.32	0.19
(Rama, 2016)	0.64	0.65	0.69	0.61	0.49
(Kanojia et al., 2019b)	0.72	0.75	0.77	0.68	0.64
Proposed Supervised Methods					
Proposed-method	0.79	0.79	0.78	0.74	0.62
Proposed-method$_{withknowledge}$	**0.91**	**0.88**	**0.87**	**0.86**	**0.75**
Proposed Unsupervised methods					
Proposed-method	0.72	0.71	0.74	0.70	0.59
Proposed-method$_{withknowledge}$	**0.86**	**0.83**	**0.84**	**0.80**	**0.73**

Table 6.6: Results of supervised and unsupervised cognate detection task based on F-Score for Indian and Celtic languages in absence of Pivot Languages.

since it is the historical ancestor of almost all Indo-Aryan languages. We have studied the model's efficiency while transferring the knowledge of Sanskrit to modern languages. In this experiment, we have taken Indian language pairs Hindi-Punjabi, Hindi-Marathi and Marathi-Bengali. Comparing the results of the models given in Table 6.7, we can observe a slight dip in model efficiency in both supervised and unsupervised frameworks while transferring the knowledge from Sanskrit compared to transferring the knowledge from Hindi to the language pairs Hindi-Punjabi and Hindi-Marathi. However, the performance for the language pair Marathi-Bengali has improved on an average of **1.5** points of F-score. We believe its performance can be attributed to the closeness and preserving more similarities in the characteristics of the language pairs Marathi and Bengali to Sanskrit than Hindi. Sanskrit is considered a highly agglutinating and morphologically rich language (Kumar, 1975), thus it is hard to parse it computationally. Though Marathi and Bengali are not as morphologically complex as Sanskrit, the languages in this pair are more agglutinating and morphologically richer than Hindi and Punjabi.

Approaches / Languages	Hi-Mr	Hi-Pa	Mr-Bn
Proposed Supervised Methods			
With Hindi Knowledge	0.91	0.88	0.86
With Sanskrit Knowledge	0.90	0.86	0.87
Proposed Unsupervised methods			
With Hindi Knowledge	0.85	0.81	0.80
With Sanskrit Knowledge	0.83	0.80	0.82

Table 6.7: Results of supervised and unsupervised cognate detection task based on F-Score for Indian languages transferring knowledge from Hindi and Sanskrit.

6.4.4 Improved Sentence Embedding

While we have designed the novel unsupervised cognate detection methodology, in this section we have responded to the second part of our **RQ3** which is:

- **RQ3.2.** Having detected cognates, does this assist in producing better sentence embeddings for NLP applications?

Meng et al., 2001 argued that cognate detection helps to perform cross-lingual information retrieval. Recently Y. Yang et al., 2020 proposed multilingual sentence embedding for semantic retrieval task. The State-of-the-art LASER model (Artetxe & Schwenk, 2019) trained with large parallel corpora is said to be the best

performing sentence encoder for parallel sentence matching. This helps to gather different cross-lingual information from the web for different domain specific tasks including question-answer retrieval. We argue that by injecting automatically detected cognates as training dataset, our newly introduced *DuEAM* model (refer to Chapter 5) can outperform LASER. We also argue that in this training process, the *DuEAM* model demands a lower amount of training data than LASER.

To perform the experiment we have taken reference of the training methodology of *DuEAM* model on 25K parallel sentences from TED2020 (refer to Section 5.2.5 in Chapter 5). TED2020 is a corpus containing translated subtitles for about 4000 TED talks and available for almost 100 languages (Reimers & Gurevych, 2020). We have trained the model on Hindi-Marathi and Hindi-Bengali dataset and tested on the same language pairs. For testing we have held out 1K parallel sentences (across languages pairs) from TED2020 and evaluated models performances based on accuracy of finding parallel sentences. We injected the cognate pairs detected by our unsupervised methodology with the 25K parallel sentence pairs as single word sentences for each language pair.

Model	HI	BN	MR
$DuEAM_{withoutcognate}$	88.7	82.6	86.5
LASER	91.2	86.8	87.9
$DuEAM_{withcognate}$	93.4	87.1	89.5

Table 6.8: Average accuracy on TED2020 (considering both directions).

From Table 6.8, we can see that the *DuEAM* framework trained with parallel sentences along with cognates outperformed the state-of-the-art method LASER by on average 14% in accuracy while trained on comparatively lesser training datasets for each language pairs. If we compare the vanilla *DuEAM* methodology with our proposed training process, our theory of injecting cognates in the training corpora improves the model efficiency for parallel sentence matching. This also proves the hypothesis of capturing better semantic understanding of cross-lingual sentences by a sentence encoder while having the knowledge of cognate pairs between languages.

6.4.4.1 *Cross-lingual Parallel Sentence Matching*

In this section, we compare the performance of the approaches in extracting parallel sentences using the **Tatoeba** benchmark of Artetxe and Schwenk, 2019. Our aim is to evaluate the performance of both supervised (Reimers & Gurevych, 2020) and unsupervised (*DuEAM* model introduced in Chapter 5) sentence embedding frameworks when fintuned on cognates. We trained the model with Hindi-Marathi, Hindi-Bengali Indo-Aryan cognate pairs. Evaluation for the parallel sentence matching task is done by finding the most similar sentence between two languages based on their cosine similarity. We have calculated accuracy in both directions (English to target language and vice versa), and have reported the average accuracy of the two.

Model	HI	BN	MR
XLM-R ← SBERT-paraphrases	96.4	77.6	91.0
DuEAM	83.5	67.7	73.4
XLM-R ← SBERT-paraphrases	97.1	79.2	91.7
DuEAM	85.4	70.1	77.5

Table 6.9: Average accuracy on Tatoeba dataset in both directions (EN to target language and vice-versa). Baseline results taken from Reimers and Gurevych, 2020.

From Table 6.9, we can observe performance improvement for both the models in comparison to the vanilla models. Across languages the supervised model gained an improvement of around 1.7%. On the other hand, the unsupervised framework *DuEAM* has achieved 2.8% better accuracy on average for all three

languages. These results show that the semantic information flow with cognates between similar languages is able to produce a better shared vector space having similar sentences close to each other.

6.5 ABLATION STUDY

We now study the effects of the morphological learner on the efficiency of unsupervised cognate detection.

Effect of morphological training datasize. One of the challenges of morphological knowledge transfer is the efficient learning of word structure in the presence of few morphological training datasets. As Irish has a few training datasets, we have evaluated the proposed framework on different samples of the Irish-Manx dataset (Refer to 6.4). We have down-sampled and up-sampled the morphology training set to 30% compared to the existing datasets. From Figure 6.2, we can observe that the best F-score **0.79** is achieved when the morphological learner model is trained with 30% more data size than the original size. This emphasizes our claim of the model's efficacy even with a slight increase in the morphological training datasets, which opens up the opportunity of implementing the unsupervised cognate detection framework using different sets of under-resourced languages.

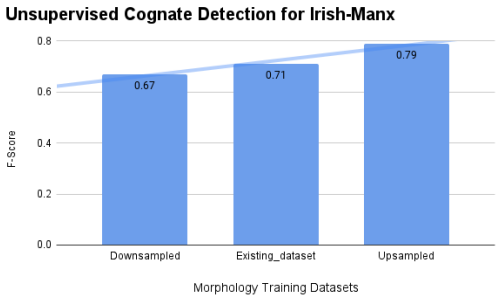

Figure 6.2: F-Score for Irish-Manx language pairs transferring knowledge from different morphological learners

6.6 CHAPTER WRAP-UP

In this chapter, we have discussed a novel methodology for identifying cognates between closely-related languages. The model is a language-agnostic cognate detection framework trained using a Siamese architecture. Experiments on three different datasets consisting of Indian languages, Celtic languages and South-African languages showcase the efficacy of our framework in understanding the structural relations between cognate words across languages. We also show that transferring morphological knowledge to the closely-related word pairs with the help of a shared encoder improves the model's efficacy in different scenarios. Our study on knowledge transfer from historical language depicts changes in the word structures of modern languages. We demonstrate that our approach outperforms the existing supervised and semi-supervised frameworks and establishes state-of-the-art supervised and unsupervised results for the cognate detection task. We also showcase the stability of learning morphology on a small number of training datasets which opens up the possibility of deploying the system across language families. Later, our experimental results on parallel sentence matching and parallel sentence mining tasks showcase the improvement of sentence embedding frameworks while trained using these cognate pairs. Thus, with the help of efficient cognate

pairs, learned representations of different applications for low-resource languages can be improved in a limited infrastructure.

7 CONCLUSIONS AND FUTURE WORK

In this chapter, we will provide the conclusions of the previous chapters and will revisit the research questions we raised in Chapter 1. We will then describe the answers we provided for the research questions. Moreover, we will also highlight the research contributions in this thesis. Lastly, we will point out various possible future research directions in line with the contributions in this thesis.

7.1 RESEARCH QUESTIONS AND CONCLUSIONS

To be trained effectively, a state-of-the-art distributional model requires a large amount of training dataset for different natural language understanding tasks. The training of these models is also dependent on the quality of the training datasets. Moreover, the multilingual sentence embedding or word embedding models rely on parallel training datasets. Often the low-resourced languages do not have sufficient parallel training datasets or may contain only monolingual datasets. Different low-resource NLP applications irrespective of domains suffer from having a low amount of annotated training datasets or noisy training datasets. Recently, datasets from social media sites are being considered for low-resourced languages. Thus, the efficiency of the language models can be improved after training. But these are mostly intra-sentential or inter-sentential code mixed datasets which are hard to be identified. Existing language identification tools like the FastText language identification tool are very limited to some of the well-resourced languages. Thus efficiently identified sentences can be considered unlabelled training datasets. This thesis is written to tackle these problems by designing state-of-the-art algorithms to capture the inner representation of the sentences or words for low-resourced languages and applications. For this, we have identified three research questions:

- **RQ1.** Is an unsupervised deep sentence embedding framework capable of identifying languages as accurately as supervised language identification models for code-mixed under-resourced and closely-related languages?
- **RQ2.** Does an unsupervised deep sentence embedding framework generate efficient sentence embeddings in cross-lingual domains for under-resourced languages without the use of parallel corpora for downstream natural language tasks? Does unsupervised projection of efficient word representations improve sentence embeddings in cross-lingual domains?
- **RQ3.** Does effective cross-lingual word representations learning improve unsupervised cognate detection without any language information for closely related and low-resourced languages? Having detected cognates, does this assist in producing better sentence embeddings for NLP applications?

We started by designing the state-of-the-art language and dialect identification framework for short texts of closely-related low-resourced languages in Chapter 4. We argued that the research question **RQ1** can be answered by capturing the global relationships between characters in a sentence. The research question we aimed to solve was

- **RQ1.1:** What is the best way to construct sentence embeddings and how to cluster them efficiently for short texts when there is no labelled training data?

To answer, we designed a system which considers a sentence as a character sequence instead of a word sequence. The system is capable of capturing the different n-gram relationships globally and locally highlighting the inner syntactic and semantic representation of the sentence. Moreover, the system scores the n-grams according to their importance for language-specific sentence construction. We train the system to capture the sentence representation during the back-propagation. Though there exist entropy-based loss functions to train the system (refer to Section 2.2), in this case, the training is unsupervised and thus the gold standard labels are absent. The newly designed *Maximum Likelihood Clustering Loss (MLC Loss)* function described in Section 4.6, can maximize the probability distribution function based on the feature assignments on each class (or cluster). We tested this methodology on three different language families and in the Table 4.2, observed substantial improvement in language and dialect identification. Therefore, we conclude that the newly designed unsupervised language and dialect identification framework is capable of identifying the under-resourced closely-related languages and dialects in a code-mixed text. This work is also applicable to supervised settings with limited pre-processing and few resource requirements.

After the language identification, the dataset having sentences from low-resource languages can be used by the distributional sentence or document embedding models for downstream tasks. But most of the these datasets do not contain efficient parallel sentences which is a bottleneck for the existing sentence or document embedding models to be trained. We try to tackle this challenge in Chapter 5 and drafted answer for the research question **RQ2**.

In Chapter 5, we tackled two problems. First

- **RQ2.1.** Does an unsupervised deep sentence embedding framework generate efficient sentence embeddings in cross-lingual domains without the use of parallel corpora for downstream natural language tasks?

The existing sentence embedding frameworks are designed to be trained on parallel sentences which is hard for low-resource languages. Our goal was to design an unsupervised sentence embedding framework to be trained with non-parallel datasets. In relation to that, we have come up with a novel idea of a dual encoder based unsupervised and weakly-supervised sentence embedding framework which produces efficient sentence embeddings for low-resourced languages as well as high-resource languages and does not require a parallel training dataset. The framework is called *DuEAM*. To design the same we have introduced a new machine learning framework called *Anchor-learner* architecture. The dual encoder with anchor model, trained in a multi-task setup with a newly introduced unsupervised semantic loss function and a translation mining based loss function. The learner module is trained to generate sentence embeddings such that the Euclidian distance between the learnt sentence embeddings closely approximates the word mover's distance (WMD) (calculated using token embeddings by anchor) between sentence pairs. We obtain word-level semantic knowledge of the source and target sentences from pretrained multilingual language models. Subsequently, embeddings for source and target are generated by the learner such that their vector space distances reflect the degree of semantic relatedness. This tends to preserve the "relative semantic distance" between the input sentences, enabling *DuEAM* to capture the "semantic relation at word level" within the sentence embeddings obtained. Forcing the system to consider the knowledge of our anchor system about the semantic relationships of the sentences at the word-level, helps to stabilize the training process as the pre-trained anchor model is fixed and the WMD score is considered as a scalar. On the other hand, while mapping semantically similar sentences close to each other in the shared vector space, an effective embedding framework should also address the translation ranking problem to efficiently map correct translations

of source-target sentence pairs within a compact zone of the vector space. Since *DuEAM* is an unsupervised approach, we introduce translation mining based loss, using the cosine similarity score between source and target sentences, to handle translation mining. Our experimental results show that the framework has outperformed supervised state-of-the-art sentence embedding frameworks on different NLP tasks including parallel sentence matching, semantic textual similarity and monolingual classification tasks. Extensive experiments for parallel sentence matching task on diverse low-resource languages of different language families highlight the model efficacy even in *zero-shot* settings. From Table 5.8, the small scale training results on low-resourced languages Georgian (KA, Kartvelian family) and Amharic (AM, Ethio-Semitic family) encouraged us to apply the *DuEAM* framework in the wide range of NLP applications with limited amount of training datasets. Moreover, the results documented in Section 5.2.4.8 depict the model stability in wide range of monolingual classification tasks while achieving better performance than supervised framework *LASER*.

Secondly we have answered the second sub research question:

- **RQ2.2.** Does unsupervised projection of efficient word representations improve sentence embeddings in cross-lingual domains?

In the Section 5.3.1 we designed the *MUFIN* framework which enhances the classification accuracy of multilingual sentence embedding models. We introduce a novel methodology which works in four phases:

- In the first phase *MUFIN* performs dimensionality expansion on the fixed-length sentence embeddings by upprojecting the pre-trained sentence embeddings to the vector space of large language models. The up-projection is performed in two steps: The up-projection is achieved with the help of a fully connected neural network via self reconstruction of data based on back-propagation through the network layers. An input sentence embedding is passed through the neural network which learns to output higher-dimensional embedding via a non-linear transformation shown in the slides. The projected embeddings are then transferred to the same vector space as that of the large language models by minimising the mean-squared loss between their sentence embeddings. This up-projection stage injects semantic information of large language models trained on a generic dataset to the sentence embedding model and represents them together in the common space. One of the main advantages of our framework is that this module is unsupervised, which eliminates the need for expensive fine-tuning of the pre-trained large language model.

- In phase two the up-projected sentence embeddings are aligned to the downstream application domain using the task specific training dataset. The up-projected sentence embeddings are provided to another fully connected neural network with an output softmax layer, which learns to predict category probability distribution (for an input embedding) across the n training labels. This enables domain adaptation and aligns the embeddings to be tuned specifically for the downstream task.

- In phase three we used the idea of meta-embedding techniques. The different sentence embedding techniques capture different aspects of semantics, and demonstrates varying degrees of success on different classes of tasks. Also, various aligned sentence encoders demonstrate complementary strengths, which upon being fused together provides improved performance – further enabling *MUFIN* to bridge the gap with fine-tuned large language models. To this end, we combine aligned up-projected embeddings and add the frozen *LASER* model to it. The choice to use "frozen" generic *LASER* embeddings is to enable generalized semantic knowledge to also be integrated within our framework, along with the domain semantic understanding obtained from the aligned up-projected encodings.

- Finally in phase four, the model learns to classify the sentence constructions using sentence meta-embedding obtained from the above step.

Along with that, we also showcase the performance of individual sentence embedding models (supervised and unsupervised) using only phase one, two and four. Extensive experiments highlight the efficacy of this framework on different domain specific or mixed-domain natural language understanding benchmarks. We observe that *MUFIN* has outperformed the baseline systems and produced comparable results to large pretrained language models. In fact, our proposed unsupervised model *DuEAM* has outperformed the state-of-the-art supervised sentence embedding models on the benchmark tasks. Moreover, for the in-domain classification dataset, for some languages it has achieved comparable performance to the supervised vanilla meta embedding models. These results advocate our intuition of improving the performance of the sentence embedding models for natural language understanding tasks by an unsupervised embedding up-projection. The capability of our unsupervised model also opens up the applicability of this methodology in small computational and training environments.

While *DuEAM* has achieved state-of-the-art performance in different downstream tasks, for different closely-related languages the quality of the training datasets are the key factor while training. The sentences often contain etymologically related word pairs called "cognates". For different natural processing tasks including information retrieval or translation mining the words with similar meaning are often also considered as cognates. We argue that these cognates have a direct impact on the sentence embedding models for low-resource languages. Thus it raises the following research question:

- **RQ3.** Does effective cross-lingual word representations learning improve unsupervised cognate detection without any language information for closely related and low-resourced languages? Having detected cognates, does this assist in producing better sentence embeddings for NLP applications?

In Chapter 6, we introduced a novel unsupervised cognate detection framework. The model has two components in which one component is morphological representation learner. This component is trained on the morphological dataset of each pivot language for each language families to learn the grammatical word representations. The second component is a shared encoder based unsupervised cognate learner which clusters the cognate pairs of closely-related languages in an iterative approach. The shared encoder has the knowledge of the grammatical word structures of the pivot language for each language family. We introduce the positional encoder on the sub-word level to capture the structural similarity between cognate pairs. We reuse the MLC loss introduced in Chapter 4 to learn the word representations. We tested our approach on three different language families choosing one language from each of the language family as pivot language. We also tested the performance of the framework in the absence of the pivot languages for cognate detection. It can be observed that the framework has outperformed all the state-of-the-art approaches across language pairs as well as proven to be a stable network without the knowledge transfer. Moreover, we studied the morphological influence of the historical languages on the closely-related modern languages while detecting the cognates. As we argue that these cognates can improve the performance of the supervised and unsupervised sentence embedding frameworks, in Section 6.4.4, we tested the efficacy on parallel sentence matching and parallel sentence extraction. Observe, the efficacy of the sentence embedding methodologies have increased while trained on the identified cognate pairs. Thus it can be concluded that our novel methodology successfully answered the questions we raised. This methodology is also applicable to supervised settings with limited pre-processing and resource requirements.

7.2 FUTURE WORK

We believe our contributions in this thesis highlight the answers to the research questions we raised. In this section, we will introduce some of the possible future research directions and works

- We have contributed towards one of the very important tasks in the NLP pipeline, namely language and dialect identification. We have learnt the global representation of the sentences by understanding the linguistic features on the character level. While the proposed method produced state-of-the-art results in supervised and unsupervised methods, we believe the injection of grammatical and structural word representations of closely-related languages in the framework will also convey the word-level semantics. We would like to conduct experiments by extending our transfer learning approach introduced in Chapter 6 in the proposed language and dialect identification framework introduced in Chapter 4 on a wide range of low-resource languages from diverse language families. We plan to release a pre-trained language identification model for a diverse set of languages from different language families.

- While we talk about injecting the grammatical and structural knowledge into the language identification framework, we plan to conduct experiments on whether the injection of the word-level structural knowledge into the unsupervised sentence embedding framework can produce better sentence embeddings. In Chapter 6, we capture the structural and grammatical knowledge of the words of the pivot language by imposing positional encoding on the subword level. These subword level information maps the semantically similar words very close to each other in the shared vector space. While our proposed unsupervised loss function in Chapter 5 can capture the semantic relations at the word level within the sentence embeddings obtained, for closely-related languages the structural and grammatical information of words along with word-level semantics may carry better information during training. Thus, the sentence embedding framework will be more efficient in understanding the cross-lingual sentence similarities.

- We would also like to explore unsupervised instructional learning using soft prompts for better multilingual sentence representation learning. Using soft or discrete prompts, the existing works improve the efficiency of the language models for different low-resource downstream NLP tasks. In our recent work (Goswami et al., 2023), we showcased a new prompting methodology to be adapted to new low-resourced domains without explicit pre-training. Sreedhar and Parisien, 2022 in their work highlighted that using prompts language models can be adapted to perform well on new-domain dialogue systems. Because these prompts are task-specific, we would like to explore whether using different linguistic features as prompts can improve generic multilingual sentence representation learning. In Chapter 6, we inject cognates as training datasets to further improve the efficiency of the sentence embedding models. We believe using such cross-lingual linguistic features including morphological similarities of two closely-related languages as prompts can provide information about language relatedness. We plan to conduct these experiments on a diverse set of closely-related languages including Indo-European, Celtic and South African language families.

BIBLIOGRAPHY

Abney, S., & Bird, S. (2010). The Human Language Project: Building a Universal Corpus of the World's Languages. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 88–97. https://aclanthology.org/P10-1010

Abu-Salih, B. (2021). Domain-specific knowledge graphs: A survey. *J. Netw. Comput. Appl.*, *185*, 103076. https://doi.org/10.1016/j.jnca.2021.103076

Aharoni, R., & Goldberg, Y. (2020). Unsupervised Domain Clusters in Pretrained Language Models. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7747–7763. https://doi.org/10.18653/v1/2020.acl-main.692

Alegria, I. I., Artola, X. X., De Ilarraza, A. D., & Sarasola, K. (2011). Strategies to develop Language Technologies for Less-Resourced Languages based on the case of Basque.

Almansor, E. H., Al-Ani, A., & Hussain, F. K. (2019). Transferring Informal Text in Arabic as Low Resource Languages: State-of-the-Art and Future Research Directions. In L. Barolli, F. K. Hussain, & M. Ikeda (Eds.), *Complex, Intelligent, and Software Intensive Systems - Proceedings of the 13th International Conference on Complex, Intelligent, and Software Intensive Systems, CISIS 2019, Sydney, NSW, Australia, 3-5 July 2019* (pp. 176–187). Springer. https://doi.org/10.1007/978-3-030-22354-0_17

Amine, A., Elberrichi, Z., & Simonet, M. (2010). Automatic Language Identification: An Alternative Unsupervised Approach Using a New Hybrid Algorithm. *IJCSA*, *7*(1), 94–107.

Ankerst, M., Breunig, M. M., Kriegel, H., & Sander, J. (1999). OPTICS: Ordering Points To Identify the Clustering Structure. In A. Delis, C. Faloutsos, & S. Ghandeharizadeh (Eds.), *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadelphia, Pennsylvania, USA* (pp. 49–60). ACM Press. https://doi.org/10.1145/304182.304187

Arroyo-Fernández, I., Méndez-Cruz, C., Sierra, G., Torres-Moreno, J., & Sidorov, G. (2019). Unsupervised sentence representations as word information series: Revisiting TF-IDF. *Comput. Speech Lang.*, *56*, 107–129. https://doi.org/10.1016/j.csl.2019.01.005

Artetxe, M., Labaka, G., & Agirre, E. (2016). Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In J. Su, X. Carreras, & K. Duh (Eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016* (pp. 2289–2294). The Association for Computational Linguistics. https://doi.org/10.18653/v1/d16-1250

Artetxe, M., Labaka, G., & Agirre, E. (2018). A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In I. Gurevych & Y. Miyao (Eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers* (pp. 789–798). Association for Computational Linguistics. https://doi.org/10.18653/v1/P18-1073

Artetxe, M., & Schwenk, H. (2019). Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond. *Trans. Assoc. Comput. Linguistics*, *7*, 597–610.

Assem, H., Dutta, S., & Burgin, E. (2021). DTAFA: Decoupled Training Architecture for Efficient FAQ Retrieval. *SIGDIAL*, 423–430.

Babic, K., Martincic-Ipsic, S., & Mestrovic, A. (2020). Survey of Neural Text Representation Models. *Inf.*, *11*(11), 511. https://doi.org/10.3390/info11110511

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In Y. Bengio & Y. LeCun (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Baldi, P. (2012). Autoencoders, Unsupervised Learning, and Deep Architectures. In I. Guyon, G. Dror, V. Lemaire, G. W. Taylor, & D. L. Silver (Eds.), *Unsupervised and Transfer Learning - Workshop held at ICML 2011, Bellevue, Washington, USA, July 2, 2011* (pp. 37–50). JMLR.org. http://proceedings.mlr.press/v27/baldi12a.html

Barman, U., Das, A., Wagner, J., & Foster, J. (2014). Code Mixing: A Challenge for Language Identification in the Language of Social Media. *Proceedings of the First Workshop on Computational Approaches to Code Switching*, 13–23. https://doi.org/10.3115/v1/W14-3902

Batsuren, K., Bella, G., & Giunchiglia, F. (2019). CogNet: A Large-Scale Cognate Database. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3136–3145. https://doi.org/10.18653/v1/P19-1302

Baumann, P., & Pierrehumbert, J. (2014). Using Resource-Rich Languages to Improve Morphological Analysis of Under-Resourced Languages. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, 3355–3359. http://www.lrec-conf.org/proceedings/lrec2014/pdf/1051_Paper.pdf

Baxter, J. (1997). A Bayesian/Information Theoretic Model of Learning to Learn via Multiple Task Sampling. *Mach. Learn.*, *28*(1), 7–39. https://doi.org/10.1023/A:1007327622663

Baziotis, C., Nikolaos, A., Papalampidi, P., Kolovou, A., Paraskevopoulos, G., Ellinas, N., & Potamianos, A. (2018). NTUA-SLP at SemEval-2018 Task 3: Tracking Ironic Tweets using Ensembles of Word and Character Level Attentive RNNs. *Proceedings of The 12th International Workshop on Semantic Evaluation*, 613–621. https://doi.org/10.18653/v1/S18-1100

Bergsma, S., McNamee, P., Bagdouri, M., Fink, C., & Wilson, T. (2012). Language Identification for Creating Language-Specific Twitter Collections. *Proceedings of the Second Workshop on Language in Social Media*, 65–74. https://www.aclweb.org/anthology/W12-2108

Berment, V. (2004). Methods to computerize "little equipped" languages and groups of languages. *Université Joseph-Fourier-Grenoble I dissertation*.

Bhattacharyya, P. (2010). IndoWordNet. In N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, & D. Tapias (Eds.), *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2010, 17-23 May 2010, Valletta, Malta*. European Language Resources Association. http://www.lrec-conf.org/proceedings/lrec2010/summaries/939.html

Bhavsar, P., Safro, I., Bouaynaya, N., Polikar, R., & Dera, D. (2017). Machine learning in transportation data analytics. In *Data analytics for intelligent transportation systems* (pp. 283–307). Elsevier.

Biemann, C. (2006a). Chinese whispers: an efficient graph clustering algorithm and its application to natural language processing problems. *Proceedings of the first workshop on graph based methods for natural language processing*, 73–80.

Biemann, C. (2006b). Chinese Whispers - an Efficient Graph Clustering Algorithm and its Application to Natural Language Processing Problems. *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for Natural Language Processing*, 73–80. https://aclanthology.org/W06-3812

Bisk, Y., & Tran, K. M. (2018). Inducing Grammars with and for Neural Machine Translation. In A. Birch, A. M. Finch, M. Luong, G. Neubig, & Y. Oda (Eds.), *Proceedings of the 2nd Workshop on Neu-*

ral Machine Translation and Generation, NMT@ACL 2018, Melbourne, Australia, July 20, 2018 (pp. 25–35). Association for Computational Linguistics. https://doi.org/10.18653/v1/w18-2704

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, *3*, 993–1022. http://jmlr.org/papers/v3/blei03a.html

Blitzer, J., & Zhu, X. (2008). Semi-Supervised Learning for Natural Language Processing. *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA, Tutorial Abstracts*, 3. https://aclanthology.org/P08-5003/

Blunsom, P., & Cohn, T. (2011). A Hierarchical Pitman-Yor Process HMM for Unsupervised Part of Speech Induction. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 865–874. https://aclanthology.org/P11-1087

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguistics*, *5*, 135–146. https://doi.org/10.1162/tacl_a_00051

Bollegala, D., & Bao, C. (2018). Learning word meta-embeddings by autoencoding. *COLING*, 1650–1661.

Bosch, S., & Griesel, M. (2018). African Wordnet: facilitating language learning in African languages. *Proceedings of the 9th Global Wordnet Conference*, 306–313. https://aclanthology.org/2018.gwc-1.36

Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. (2015). A large annotated corpus for learning natural language inference. *EMNLP*, 632–642.

Brew, C., McKelvie, D., & Place, B. (1996). Word-Pair Extraction for Lexicography.

Caldarini, G., Jaf, S., & McGarry, K. (2022). A Literature Survey of Recent Advances in Chatbots. *Information*, *13*(1). https://doi.org/10.3390/info13010041

Cao, S., Kitaev, N., & Klein, D. (2020). Multilingual Alignment of Contextual Word Representations. *ICLR*.

Cardenas, R., Lin, Y., Ji, H., & May, J. (2019). A Grounded Unsupervised Universal Part-of-Speech Tagger for Low-Resource Languages. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2428–2439. https://doi.org/10.18653/v1/N19-1252

Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, *4*(6), 759–771. https://doi.org/10.1016/0893-6080(91)90056-B

Casanueva, I., Temčinas, T., Gerz, D., Henderson, M., & Vulić, I. (2020). Efficient Intent Detection with Dual Sentence Encoders. *2nd Workshop on Natural Language Processing for Conversational AI*, 38–45.

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. *SemEval-2017*, 1–14.

Chakravarthi, B. R., Arcan, M., & McCrae, J. P. (2018). Improving Wordnets for Under-Resourced Languages Using Machine Translation. *Proceedings of the 9th Global WordNet Conference*.

Chakravarthi, B. R., Arcan, M., & McCrae, J. P. (2019a). Comparison of Different Orthographies for Machine Translation of Under-Resourced Dravidian Languages. *2nd Conference on Language, Data and Knowledge (LDK 2019)*, *70*, 6:1–6:14. https://doi.org/10.4230/OASIcs.LDK.2019.6
Keywords: Under-resourced languages, Machine translation, Dravidian languages, Phonetic transcription, Transliteration, International Phonetic Alphabet, IPA, M

Chakravarthi, B. R., Arcan, M., & McCrae, J. P. (2019b). Comparison of Different Orthographies for Machine Translation of Under-Resourced Dravidian Languages. *2nd Conference on Language, Data and Knowledge (LDK 2019)*, *70*, 6:1–6:14. https://doi.org/10.4230/OASIcs.LDK.2019.6

Keywords: Under-resourced languages, Machine translation, Dravidian languages, Phonetic transcription, Transliteration, International Phonetic Alphabet, IPA, M

Chakravarthi, B. R., Jose, N., Suryawanshi, S., Sherly, E., & McCrae, J. P. (2020a). A Sentiment Analysis Dataset for Code-Mixed Malayalam-English. *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, 177–184. https://www.aclweb.org/anthology/2020.sltu-1.25

Chakravarthi, B. R., Muralidaran, V., Priyadharshini, R., & McCrae, J. P. (2020b). Corpus Creation for Sentiment Analysis in Code-Mixed Tamil-English Text. *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, 202–210. https://www.aclweb.org/anthology/2020.sltu-1.28

Chen, J., Wang, Z., Tian, R., Yang, Z., & Yang, D. (2020). Local Additivity Based Data Augmentation for Semi-supervised NER. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1241–1251. https://doi.org/10.18653/v1/2020.emnlp-main.95

Chen, L., Garcia, F., & Kumar, V. (2021). Industry Scale Semi-Supervised Learning for Natural Language Understanding. *NAACL HLT*, 311–318.

Chidambaram, M., Yang, Y., Cer, D., Yuan, S., Sung, Y., Strope, B., & Kurzweil, R. (2019). Learning Cross-Lingual Sentence Representations via a Multi-task Dual-Encoder Model. *RepL4NLP@ACL 2019*, 250–259.

Christodoulopoulos, C., Goldwater, S., & Steedman, M. (2010). Two Decades of Unsupervised POS Induction: How Far Have We Come? *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 575–584. https://aclanthology.org/D10-1056

Cieri, C., Maxwell, M., Strassel, S. M., & Tracey, J. (2016). Selection Criteria for Low Resource Language Programs. In N. Calzolari, K. Choukri, T. Declerck, S. Goggi, M. Grobelnik, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, & S. Piperidis (Eds.), *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2016/summaries/1254.html

Ciobanu, A. M., & Dinu, L. P. (2015). Automatic discrimination between cognates and borrowings. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 431–437.

Ciobanu, A. M., & Dinu, L. P. (2016). A Computational Perspective on the Romanian Dialects. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 3281–3285. https://aclanthology.org/L16-1522

Ciobanu, A. M., Malmasi, S., & Dinu, L. P. (2018). German dialect identification using classifier ensembles. *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, 288–294.

Cleary, J. G., & Witten, I. H. (1984). Data Compression Using Adaptive Coding and Partial String Matching. *IEEE Trans. Commun.*, *32*(4), 396–402. https://doi.org/10.1109/TCOM.1984.1096090

Clematide, S., & Makarov, P. (2017). CLUZH at VarDial GDI 2017: Testing a Variety of Machine Learning Tools for the Classification of Swiss German Dialects. In P. Nakov, M. Zampieri, N. Ljubesic, J. Tiedemann, S. Malmasi, & A. Ali (Eds.), *Proceedings of the Fourth Workshop on NLP for Similar*

Languages, Varieties and Dialects, VarDial 2017, Valencia, Spain, April 3, 2017 (pp. 170–177). Association for Computational Linguistics. https://doi.org/10.18653/v1/w17-1221

Coates, J. N., & Bollegala, D. (2018). Frustratingly Easy Meta-Embedding – Computing Meta-Embeddings by Averaging Source Word Embeddings. *NAACL-HLT*, 194–198.

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In W. W. Cohen, A. McCallum, & S. T. Roweis (Eds.), *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008* (pp. 160–167). ACM. https://doi.org/10.1145/1390156.1390177

Comaniciu, D., & Meer, P. (2002). Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, *24*(5), 603–619. https://doi.org/10.1109/34.1000236

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2020). Unsupervised Cross-lingual Representation Learning at Scale. In D. Jurafsky, J. Chai, N. Schluter, & J. R. Tetreault (Eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020* (pp. 8440–8451). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.747

Conneau, A., & Kiela, D. (2018). SentEval: An Evaluation Toolkit for Universal Sentence Representations. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (2017). Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, 670–680.

Conneau, A., & Lample, G. (2019). Cross-lingual Language Model Pretraining. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada* (pp. 7057–7067). https://proceedings.neurips.cc/paper/2019/hash/c04c19c2c2474dbf5f7ac4372c5b9af1-Abstract.html

Conneau, A., Rinott, R., Lample, G., Williams, A., Bowman, S. R., Schwenk, H., & Stoyanov, V. (2018). XNLI: Evaluating Cross-lingual Sentence Representations. In E. Riloff, D. Chiang, J. Hockenmaier, & J. Tsujii (Eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018* (pp. 2475–2485). Association for Computational Linguistics. https://doi.org/10.18653/v1/d18-1269

Cotterell, R., & Schütze, H. (2015). Morphological Word-Embeddings. In R. Mihalcea, J. Y. Chai, & A. Sarkar (Eds.), *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015* (pp. 1287–1292). The Association for Computational Linguistics. https://doi.org/10.3115/v1/n15-1140

Covington, M. A. (1996). An algorithm to align words for historical comparison. *Computational linguistics*, *22*(4), 481–496.

Crowston, K. (2012). Amazon Mechanical Turk: A Research Tool for Organizations and Information Systems Scholars. In A. Bhattacherjee & B. Fitzgerald (Eds.), *Shaping the Future of ICT Research. Methods and Approaches - IFIP WG 8.2, Working Conference, Tampa, FL, USA, December 13-14, 2012. Proceedings* (pp. 210–221). Springer. https://doi.org/10.1007/978-3-642-35142-6_14

Crystal, D. (2011). *A dictionary of linguistics and phonetics*. John Wiley & Sons.

Cui, Y., Che, W., Liu, T., Qin, B., & Yang, Z. (2021). Pre-Training With Whole Word Masking for Chinese BERT. *IEEE ACM Trans. Audio Speech Lang. Process.*, *29*, 3504–3514. https://doi.org/10.1109/TASLP.2021.3124365

Cunliffe, D., Vlachidis, A., Williams, D., & Tudhope, D. (2022). Natural language processing for under-resourced languages: Developing a Welsh natural language toolkit. *Computer Speech Language*, *72*, 101311. https://doi.org/https://doi.org/10.1016/j.csl.2021.101311

Dahou, A., Xiong, S., Zhou, J., Haddoud, M. H., & Duan, P. (2016). Word Embeddings and Convolutional Neural Network for Arabic Sentiment Classification. In N. Calzolari, Y. Matsumoto, & R. Prasad (Eds.), *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan* (pp. 2418–2427). ACL. https://aclanthology.org/C16-1228/

Dai, A. M., Olah, C., & Le, Q. V. (2015). Document Embedding with Paragraph Vectors. *CoRR*, *abs/1507.07998*. http://arxiv.org/abs/1507.07998

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (pp. 4171–4186). Association for Computational Linguistics. https://doi.org/10.18653/v1/n19-1423

Dhillon, I. S., Guan, Y., & Kulis, B. (2007). Weighted Graph Cuts without Eigenvectors A Multilevel Approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, *29*(11), 1944–1957. https://doi.org/10.1109/TPAMI.2007.1115

Dieth, E. (1986). *Schwyzertütschi Dialäktschrift* (2nd ed.) [Edited and published by Christian Schmid-Cadalbert]. Verlag Sauerländer.

Diwersy, S., Evert, S., & Neumann, S. (2014). A weakly supervised multivariate approach to the study of language variation. *Aggregating dialectology, typology, and register analysis. linguistic variation in text and speech*, 174–204.

Dongen, N. (2017). Analysis and Prediction of Dutch-English Code-switching in Dutch Social Media Messages.

Donges, N. (2019). Gradient descent: an introduction to 1 of machine learning's most popular algorithms. *source: https://builtin. com/data-science/gradient-descent*.

Douka, S., Abdine, H., Vazirgiannis, M., Hamdani, R. E., & Amariles, D. R. (2021). JuriBERT: A Masked-Language Model Adaptation for French Legal Text. *CoRR*, *abs/2110.01485*. https://arxiv.org/abs/2110.01485

Duan, X., Ji, B., Jia, H., Tan, M., Zhang, M., Chen, B., Luo, W., & Zhang, Y. (2020). Bilingual Dictionary Based Neural Machine Translation without Using Parallel Sentences. In D. Jurafsky, J. Chai, N. Schluter, & J. R. Tetreault (Eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020* (pp. 1570–1579). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.143

Duong, L. (2017). Natural language processing for resource-poor languages. *University of Melbourne*.

Duong, L., Cohn, T., Bird, S., & Cook, P. (2015). Low Resource Dependency Parsing: Cross-lingual Parameter Sharing in a Neural Network Parser. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, 845–850. https://doi.org/10.3115/v1/p15-2139

Duvenhage, B., Ntini, M., & Ramonyai, P. (2017). Improved text language identification for the South African languages. *2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech)*, 214–218.

Elfardy, H., & Diab, M. T. (2013). Sentence Level Dialect Identification in Arabic. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers*, 456–461. https://aclanthology.org/P13-2081/

Elman, J. L. (1990). Finding Structure in Time. *Cogn. Sci.*, *14*(2), 179–211. https://doi.org/10.1207/s15516709cog1402_1

España-Bonet, C., Varga, Á. C., Barrón-Cedeño, A., & van Genabith, J. (2017). An Empirical Analysis of NMT-Derived Interlingual Embeddings and Their Use in Parallel Sentence Identification. *IEEE J. Sel. Top. Signal Process.*, *11*, 1340–1350.

Ester, M., Kriegel, H., Sander, J., & Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In E. Simoudis, J. Han, & U. M. Fayyad (Eds.), *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA* (pp. 226–231). AAAI Press. http://www.aaai.org/Library/KDD/1996/kdd96-037.php

Ezugwu, A. E., Ikotun, A. M., Oyelade, O. N., Abualigah, L. M., Agushaka, J. O., Eke, C. I., & Akinyelu, A. A. (2022). A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Eng. Appl. Artif. Intell.*, *110*, 104743. https://doi.org/10.1016/j.engappai.2022.104743

Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., & Lehmann, S. (2017). Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In M. Palmer, R. Hwa, & S. Riedel (Eds.), *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017* (pp. 1615–1625). Association for Computational Linguistics. https://doi.org/10.18653/v1/d17-1169

Feng, F., Yang, Y., Cer, D., Arivazhagan, N., & Wang, W. (2020). Language-agnostic BERT Sentence Embedding. *CoRR*, *abs/2007.01852*.

Feng, F., Yang, Y., Cer, D., Arivazhagan, N., & Wang, W. (2022). Language-agnostic BERT Sentence Embedding. In S. Muresan, P. Nakov, & A. Villavicencio (Eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022* (pp. 878–891). Association for Computational Linguistics. https://aclanthology.org/2022.acl-long.62

Ferrone, L., & Zanzotto, F. M. (2019). Symbolic, Distributed, and Distributional Representations for Natural Language Processing in the Era of Deep Learning: A Survey. *Frontiers Robotics AI*, *6*, 153. https://doi.org/10.3389/frobt.2019.00153

Fourrier, C., & Montariol, S. (2022). Caveats of Measuring Semantic Change of Cognates and Borrowings using Multilingual Word Embeddings. In N. Tahmasebi, S. Montariol, A. Kutuzov, S. Hengchen, H. Dubossarsky, & L. Borin (Eds.), *Proceedings of the 3rd Workshop on Computational Approaches to Historical Language Change, LChange@ACL 2022, Dublin, Ireland, May 26-27, 2022* (pp. 97–112). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.lchange-1.10

Gao, Y., Chen, J., & Zhu, J. (2016). Streaming Gibbs sampling for LDA model. *arXiv preprint arXiv:1601.01142*.

Gaussier, É., Renders, J., Matveeva, I., Goutte, C., & Déjean, H. (2004). A Geometric View on Bilingual Lexicon Extraction from Comparable Corpora. In D. Scott, W. Daelemans, & M. A. Walker (Eds.), *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain* (pp. 526–533). ACL. https://doi.org/10.3115/1218955.1219022

Ge, R., Kakade, S. M., Kidambi, R., & Netrapalli, P. (2019). The Step Decay Schedule: A Near Optimal, Geometrically Decaying Learning Rate Procedure For Least Squares. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada* (pp. 14951–14962). https://proceedings.neurips.cc/paper/2019/hash/2f4059ce1227f021edc5d9c6f0f17dc1-Abstract.html

Giwa, O., & Davel, M. H. (2013). N-gram based language identification of individual words.

Goldberg, Y. (2017). *Neural Network Methods for Natural Language Processing*. Morgan & Claypool Publishers. https://doi.org/10.2200/S00762ED1V01Y201703HLT037

Goldberg, Y., & Hirst, G. (2017). Neural network methods in natural language processing. morgan & claypool publishers (2017). *zitiert auf*, 69.

Goodfellow, I. J., Bengio, Y., & Courville, A. C. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org/

Goswami, K., Lange, L., Araki, J., & Adel, H. (2023). SwitchPrompt: Learning Domain-Specific Gated Soft Prompts for Classification in Low-Resource Domains. *CoRR*, *abs/2302.06868*. https://doi.org/10.48550/arXiv.2302.06868

Goswami, K., Sarkar, R., Chakravarthi, B. R., Fransen, T., & McCrae, J. P. (2020). Unsupervised Deep Language and Dialect Identification for Short Texts. In D. Scott, N. Bel, & C. Zong (Eds.), *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020* (pp. 1606–1617). International Committee on Computational Linguistics. https://doi.org/10.18653/v1/2020.coling-main.141

Graves, A., Jaitly, N., & Mohamed, A. (2013). Hybrid speech recognition with Deep Bidirectional LSTM. *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013*, 273–278. https://doi.org/10.1109/ASRU.2013.6707742

Grinstead, C. M., & Snell, J. L. (2009). Conditional probability–discrete conditional. *Grinstead & Snell's introduction to probability Orange Grove Texts*.

Guha, S., Rastogi, R., & Shim, K. (1999). ROCK: A Robust Clustering Algorithm for Categorical Attributes. In M. Kitsuregawa, M. P. Papazoglou, & C. Pu (Eds.), *Proceedings of the 15th International Conference on Data Engineering, Sydney, Australia, March 23-26, 1999* (pp. 512–521). IEEE Computer Society. https://doi.org/10.1109/ICDE.1999.754967

Guha, S., Rastogi, R., & Shim, K. (2001). Cure: An Efficient Clustering Algorithm for Large Databases. *Inf. Syst.*, *26*(1), 35–58. https://doi.org/10.1016/S0306-4379(01)00008-4

Guo, M., Shen, Q., Yang, Y., Ge, H., Cer, D., Hernandez Abrego, G., Stevens, K., Constant, N., Sung, Y.-H., Strope, B., & Kurzweil, R. (2018). Effective Parallel Corpus Mining using Bilingual Sentence Embeddings. *Proceedings of the Third Conference on Machine Translation: Research Papers*, 165–176.

Haffari, R., Cherry, C., Foster, G., Khadivi, S., & Salehi, B. (Eds.). (2018). *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*. Association for Computational Linguistics. https://aclanthology.org/W18-3400

Hammarström, H. (2007). A fine-grained model for language identification. *Proceedings of iNEWS-07 Workshop at SIGIR 2007*, 14–20.

Hauer, B., & Kondrak, G. (2011). Clustering semantically equivalent words into cognate sets in multilingual lists. *Proceedings of 5th international joint conference on natural language processing*, 865–873.

Hauksdóttir, A. (2014). An Innovative World Language Centre : Challenges for the Use of Language Technology. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, 2194–2198. http://www.lrec-conf.org/proceedings/lrec2014/pdf/795_Paper.pdf

Hedderich, M. A., Lange, L., Adel, H., Strötgen, J., & Klakow, D. (2021). A Survey on Recent Approaches for Natural Language Processing in Low-Resource Scenarios. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, & Y. Zhou (Eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021* (pp. 2545–2568). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.naacl-main.201

Hill, F., Cho, K., & Korhonen, A. (2016). Learning Distributed Representations of Sentences from Unlabelled Data. In K. Knight, A. Nenkova, & O. Rambow (Eds.), *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016* (pp. 1367–1377). The Association for Computational Linguistics. https://doi.org/10.18653/v1/n16-1162

Hirota, W., Suhara, Y., Golshan, B., & Tan, W. (2020). Emu: Enhancing Multilingual Sentence Embeddings with Semantic Specialization. *AAAI*, 7935–7943.

Hochreiter, S., & Schmidhuber, J. (1997a). Long Short-Term Memory. *Neural Comput.*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hochreiter, S., & Schmidhuber, J. (1997b). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Hofmann, T. (1999). Probabilistic latent semantic indexing. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 50–57.

House, A. S., & Neuburg, E. P. (1977). Toward automatic identification of the language of an utterance. I. Preliminary methodological considerations. *The Journal of the Acoustical Society of America*, *62*(3), 708–713.

Howard, J., & Ruder, S. (2018a). Universal Language Model Fine-tuning for Text Classification. In I. Gurevych & Y. Miyao (Eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers* (pp. 328–339). Association for Computational Linguistics. https://doi.org/10.18653/v1/P18-1031

Howard, J., & Ruder, S. (2018b). Universal Language Model Fine-tuning for Text Classification. *ACL*, 328–339.

Hua, T., Lu, C., Choo, J., & Reddy, C. K. (2020). Probabilistic Topic Modeling for Comparative Analysis of Document Collections. *ACM Trans. Knowl. Discov. Data*, *14*(2), 24:1–24:27. https://doi.org/10.1145/3369873

Huang, C.-R., & Lee, L.-H. (2008). Contrastive approach towards text source classification based on top-bag-of-word similarity. *Proceedings of the 22nd pacific asia conference on language, information and computation*, 404–410.

Huang, J., Tang, D., Zhong, W., Lu, S., Shou, L., Gong, M., Jiang, D., & Duan, N. (2021). WhiteningBERT: An Easy Unsupervised Sentence Embedding Approach. In M. Moens, X. Huang, L. Specia, & S. W. Yih (Eds.), *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021* (pp. 238–244). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.findings-emnlp.23

Hughes, B., Baldwin, T., Bird, S., Nicholson, J., & MacKinlay, A. (2006a). Reconsidering Language Identification for Written Language Resources. In N. Calzolari, K. Choukri, A. Gangemi, B. Maegaard,

J. Mariani, J. Odijk, & D. Tapias (Eds.), *Proceedings of the Fifth International Conference on Language Resources and Evaluation, LREC 2006, Genoa, Italy, May 22-28, 2006* (pp. 485–488). European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2006/summaries/459.html

Hughes, B., Baldwin, T., Bird, S., Nicholson, J., & MacKinlay, A. (2006b). Reconsidering Language Identification for Written Language Resources. *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. http://www.lrec-conf.org/proceedings/lrec2006/pdf/459_pdf.pdf

Hull, D. A. (1999). Xerox TREC-8 Question Answering Track Report. In E. M. Voorhees & D. K. Harman (Eds.), *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999*. National Institute of Standards; Technology (NIST). http://trec.nist.gov/pubs/trec8/papers/xerox-QA.pdf

Hussein, A., & Hajj, H. M. (2022). Domain Adaptation with Representation Learning and Nonlinear Relation for Time Series. *ACM Trans. Internet Things*, *3*(2), 12:1–12:26. https://doi.org/10.1145/3502905

Jäger, G., List, J.-M., & Sofroniev, P. (2017). Using support vector machines and state-of-the-art algorithms for phonetic alignment to identify cognates in multi-lingual wordlists. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 1205–1216. https://aclanthology.org/E17-1113

Jauhiainen, T. S., Lui, M., Zampieri, M., Baldwin, T., & Lindén, K. (2019). Automatic language identification in texts: A survey. *Journal of Artificial Intelligence Research*, *65*, 675–782.

Jayagopal, A., Aiswarya, A. M., Garg, A., & Nandakumar, S. K. (2022). Multimodal Representation Learning With Text and Images. *CoRR*, *abs/2205.00142*. https://doi.org/10.48550/arXiv.2205.00142

Ji, Y., Cohn, T., Kong, L., Dyer, C., & Eisenstein, J. (2015). Document Context Language Models. *CoRR*, *abs/1511.03962*. http://arxiv.org/abs/1511.03962

Jimerson, R., & Prud'hommeaux, E. (2018). ASR for documenting acutely under-resourced indigenous languages. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Jose, N., Chakravarthi, B. R., Suryawanshi, S., Sherly, E., & McCrae, J. P. (2020). A Survey of Current Datasets for Code-Switching Research. *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 136–141.

Jurafsky, D., & Martin, J. H. (2009). *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd Edition*. Prentice Hall, Pearson Education International. https://www.worldcat.org/oclc/315913020

Jurgens, D. (2021). Learning about Word Vector Representations and Deep Learning through Implementing Word2vec. *Proceedings of the Fifth Workshop on Teaching NLP*, 108–111. https://doi.org/10.18653/v1/2021.teachingnlp-1.19

Kakwani, D., Kunchukuttan, A., Golla, S., N.C., G., Bhattacharyya, A., Khapra, M. M., & Kumar, P. (2020). IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. *Findings of the Association for Computational Linguistics: EMNLP 2020*, 4948–4961. https://doi.org/10.18653/v1/2020.findings-emnlp.445

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A Convolutional Neural Network for Modelling Sentences. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, 655–665. https://doi.org/10.3115/v1/p14-1062

Kanakaraddi, S. G., & Nandyal, S. S. (2018). Survey on Parts of Speech Tagger Techniques. *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*, 1–6. https://doi.org/10.1109/ICCTCT.2018.8550884

Kanojia, D., Dabre, R., Dewangan, S., Bhattacharyya, P., Haffari, G., & Kulkarni, M. (2020a). Harnessing Cross-lingual Features to Improve Cognate Detection for Low-resource Languages. In D. Scott, N. Bel, & C. Zong (Eds.), *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020* (pp. 1384–1395). International Committee on Computational Linguistics. https://doi.org/10.18653/v1/2020.coling-main.119

Kanojia, D., Dabre, R., Dewangan, S., Bhattacharyya, P., Haffari, G., & Kulkarni, M. (2020b). Harnessing Cross-lingual Features to Improve Cognate Detection for Low-resource Languages. *Proceedings of the 28th International Conference on Computational Linguistics*, 1384–1395. https://doi.org/10.18653/v1/2020.coling-main.119

Kanojia, D., Kulkarni, M., Bhattacharyya, P., & Haffari, G. (2020c). Challenge Dataset of Cognates and False Friend Pairs from Indian Languages. In N. Calzolari, F. Béchet, P. Blache, K. Choukri, C. Cieri, T. Declerck, S. Goggi, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, & S. Piperidis (Eds.), *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020* (pp. 3096–3102). European Language Resources Association. https://aclanthology.org/2020.lrec-1.378/

Kanojia, D., Kulkarni, M., Bhattacharyya, P., & Haffari, G. (2019a). Cognate Identification to Improve Phylogenetic Trees for Indian Languages. *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, 297–300. https://doi.org/10.1145/3297001.3297045

Kanojia, D., Patel, K., & Bhattacharyya, P. (2018). Indian Language Wordnets and their Linkages with Princeton WordNet. In N. Calzolari, K. Choukri, C. Cieri, T. Declerck, S. Goggi, K. Hasida, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, S. Piperidis, & T. Tokunaga (Eds.), *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2018/summaries/936.html

Kanojia, D., Patel, K., Kulkarni, M., Bhattacharyya, P., & Haffari, G. (2019b). Utilizing Wordnets for Cognate Detection among Indian Languages. *Proceedings of the 10th Global Wordnet Conference*, 404–412.

Kanojia, D., Sharma, P., Ghodekar, S., Bhattacharyya, P., Haffari, G., & Kulkarni, M. (2021). Cognition-aware Cognate Detection. *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 3281–3292. https://doi.org/10.18653/v1/2021.eacl-main.288

Kenter, T., Borisov, A., & de Rijke, M. (2016). Siamese CBOW: Optimizing Word Embeddings for Sentence Representations. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. https://doi.org/10.18653/v1/p16-1089

Kiela, D., Conneau, A., Jabri, A., & Nickel, M. (2018). Learning Visually Grounded Sentence Representations. In M. A. Walker, H. Ji, & A. Stent (Eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)* (pp. 408–418). Association for Computational Linguistics. https://doi.org/10.18653/v1/n18-1038

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. In A. Moschitti, B. Pang, & W. Daelemans (Eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL* (pp. 1746–1751). ACL. https://doi.org/10.3115/v1/d14-1181

King, B., & Abney, S. (2013). Labeling the languages of words in mixed-language documents using weakly supervised methods. *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1110–1119.

Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R. S., Urtasun, R., Torralba, A., & Fidler, S. (2015a). Skip-Thought Vectors. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada* (pp. 3294–3302). https://proceedings.neurips.cc/paper/2015/hash/f442d33fa06832082290ad8544a8da27-Abstract.html

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., & Fidler, S. (2015b). Skip-thought vectors. *Advances in neural information processing systems*, 3294–3302.

Klema, V., & Laub, A. (1980). The singular value decomposition: Its computation and some applications. *IEEE Transactions on Automatic Control*, *25*(2), 164–176. https://doi.org/10.1109/TAC.1980.1102314

Kocmi, T., & Bojar, O. (2017). LanideNN: Multilingual Language Identification on Character Window. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 927–936. https://www.aclweb.org/anthology/E17-1087

Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. *Proceedings of Machine Translation Summit X: Papers*, 79–86. https://aclanthology.org/2005.mtsummit-papers.11

Koehn, P., Och, F. J., & Marcu, D. (2003). Statistical Phrase-Based Translation. In M. A. Hearst & M. Ostendorf (Eds.), *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003*. The Association for Computational Linguistics. https://aclanthology.org/N03-1017/

Kondrak, G. (2000). A new algorithm for the alignment of phonetic sequences. *1st Meeting of the North American Chapter of the Association for Computational Linguistics*.

Kondrak, G. (2005). Cognates and word alignment in bitexts. *Proceedings of Machine Translation Summit X: Papers*, 305–312.

Kondrak, G., Marcu, D., & Knight, K. (2003). Cognates Can Improve Statistical Translation Models. In M. A. Hearst & M. Ostendorf (Eds.), *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003*. The Association for Computational Linguistics. https://aclanthology.org/N03-2016/

Konstantopoulos, S. (2007). What's in a Name? *CoRR*, *abs/0710.1481*. http://arxiv.org/abs/0710.1481

Kumar, C. S. (1975). Origin and Development of the Bengali Language.

Kusner, M. J., Sun, Y., Kolkin, N. I., & Weinberger, K. Q. (2015). From Word Embeddings To Document Distances. In F. R. Bach & D. M. Blei (Eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015* (pp. 957–966). JMLR.org. http://proceedings.mlr.press/v37/kusnerb15.html

Kvapilíková, I., Artetxe, M., Labaka, G., Agirre, E., & Bojar, O. (2020). Unsupervised Multilingual Sentence Embeddings for Parallel Corpus Mining. In S. Rijhwani, J. Liu, Y. Wang, & R. Dror (Eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student*

Research Workshop, ACL 2020, Online, July 5-10, 2020 (pp. 255–262). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-srw.34

Labat, S., & Lefever, E. (2019). A Classification-Based Approach to Cognate Detection Combining Orthographic and Semantic Similarity Information. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, 602–610. https://doi.org/10.26615/978-954-452-056-4_071

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In C. E. Brodley & A. P. Danyluk (Eds.), *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001* (pp. 282–289). Morgan Kaufmann.

Le, Q. V., & Mikolov, T. (2014). Distributed Representations of Sentences and Documents. *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, *32*, 1188–1196. http://proceedings.mlr.press/v32/le14.html

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Lee, S.-w., Lee, R., Seo, M.-s., Park, J.-c., Noh, H.-c., Ju, J.-g., Jang, R.-y., Lee, G.-w., Choi, M.-s., & Choi, D.-g. (2021). Multi-Task Learning with Task-Specific Feature Filtering in Low-Data Condition. *Electronics*, *10*(21), 2691.

Li, X., & Roth, D. (2002). Learning Question Classifiers. *19th International Conference on Computational Linguistics, COLING 2002, Howard International House and Academia Sinica, Taipei, Taiwan, August 24 - September 1, 2002*.

Liang, Y., Duan, N., Gong, Y., Wu, N., Guo, F., Qi, W., Gong, M., Shou, L., Jiang, D., Cao, G., Fan, X., Zhang, R., Agrawal, R., Cui, E., Wei, S., Bharti, T., Qiao, Y., Chen, J.-H., Wu, W., … Zhou, M. (2020). XGLUE: A New Benchmark Datasetfor Cross-lingual Pre-training, Understanding and Generation. *EMNLP*.

Lin, C.-C., Ammar, W., Levin, L., & Dyer, C. (2014). The CMU Submission for the Shared Task on Language Identification in Code-Switched Data. *Proceedings of the First Workshop on Computational Approaches to Code Switching*, 80–86. https://doi.org/10.3115/v1/W14-3909

Lin, Z., Feng, M., dos Santos, C. N., Yu, M., Xiang, B., Zhou, B., & Bengio, Y. (2017). A Structured Self-Attentive Sentence Embedding. *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. https://openreview.net/forum?id=BJC%5C_jUqxe

Liu, X., Eshghi, A., Swietojanski, P., & Rieser, V. (2019). Benchmarking Natural Language Understanding Services for Building Conversational Agents. *IWSDS*, 165–183.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, *abs/1907.11692*. http://arxiv.org/abs/1907.11692

Liu, Z., Wang, X., Chen, Q., & Tang, B. (2018). Chinese Clinical Entity Recognition via Attention-Based CNN-LSTM-CRF. *2018 IEEE International Conference on Healthcare Informatics Workshop (ICHI-W)*, 68–69.

Liu, Z., Lin, Y., & Sun, M. (2020). *Representation Learning for Natural Language Processing*. Springer. https://doi.org/10.1007/978-981-15-5573-2

Lui, M., & Baldwin, T. (2012). langid.py: An Off-the-shelf Language Identification Tool. *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the System Demonstrations, July 10, 2012, Jeju Island, Korea*, 25–30. https://aclanthology.org/P12-3005/

Lui, M., & Baldwin, T. (2014). Accurate Language Identification of Twitter Messages. *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, 17–25. https://doi.org/10.3115/v1/W14-1303

Maaten, L. v. d., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, *9*(Nov), 2579–2605.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, 281–297. https://projecteuclid.org/euclid.bsmsp/1200512992

Makin, R., Pandey, N., Pingali, P., & Varma, V. (2007). Experiments in cross-lingual ir among indian languages. *Proceedings of the International Workshop on Cross Language Information Processing (CLIP)*.

Malmasi, S., & Zampieri, M. (2017). German dialect identification in interview transcriptions. *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, 164–169.

Mann, G. S., & Yarowsky, D. (2001). Multipath Translation Lexicon Induction via Bridge Languages. *Second Meeting of the North American Chapter of the Association for Computational Linguistics*. https://aclanthology.org/N01-1020

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press. https://doi.org/10.1017/CBO9780511809071

Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Comput. Linguistics*, *19*(2), 313–330.

Martins, B., & Silva, M. J. (2005). Language identification in web pages. In H. Haddad, L. M. Liebrock, A. Omicini, & R. L. Wainwright (Eds.), *Proceedings of the 2005 ACM Symposium on Applied Computing (SAC), Santa Fe, New Mexico, USA, March 13-17, 2005* (pp. 764–768). ACM. https://doi.org/10.1145/1066677.1066852

Mather, L. A. (1998). A Linear Algebra Approach to Language Identification. *Proceedings of the 4th International Workshop on Principles of Digital Document Processing*, 92–103.

Mati, D. N., Hamiti, M., Susuri, A., Selimi, B., & Ajdari, J. (2021). Building Dictionaries for Low Resource Languages: Challenges of Unsupervised Learning. *Annals of Emerging Technologies in Computing (AETiC)*, *5*(3), 52–58.

Maxwell, M., & Hughes, B. (2006). Frontiers in linguistic annotation for lower-density languages. *Proceedings of the workshop on frontiers in linguistically annotated corpora 2006*, 29–37.

McCarthy, A. D., Kirov, C., Grella, M., Nidhi, A., Xia, P., Gorman, K., Vylomova, E., Mielke, S. J., Nicolai, G., Silfverberg, M., Arkhangelskiy, T., Krizhanovsky, N., Krizhanovsky, A., Klyachko, E., Sorokin, A., Mansfield, J., Ernštreits, V., Pinter, Y., Jacobs, C. L., … Yarowsky, D. (2020). UniMorph 3.0: Universal Morphology. *Proceedings of the 12th Language Resources and Evaluation Conference*, 3922–3931. https://aclanthology.org/2020.lrec-1.483

McNamee, P. (2005). Language identification: a solved problem suitable for undergraduate instruction. *Journal of computing sciences in colleges*, *20*(3), 94–101.

Melamed, I. D. (1999). Bitext Maps and Alignment via Pattern Recognition. *Computational Linguistics*, *25*(1), 107–130. https://aclanthology.org/J99-1003

Meng, H., Lo, W.-K., Chen, B., & Tang, K. (2001). Generating phonetic cognates to handle named entities in English-Chinese cross-language spoken document retrieval. *IEEE Workshop on Automatic Speech Recognition and Understanding, 2001. ASRU '01.*, 311–314. https://doi.org/10.1109/ASRU.2001.1034649

Merlo, P., & Rodriguez, M. A. (2019a). Cross-Lingual Word Embeddings and the Structure of the Human Bilingual Lexicon. In M. Bansal & A. Villavicencio (Eds.), *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019* (pp. 110–120). Association for Computational Linguistics. https://doi.org/10.18653/v1/K19-1011

Merlo, P., & Rodriguez, M. A. (2019b). Cross-lingual word embeddings and the structure of the human bilingual lexicon. *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 110–120.

Meurers, W. D. (2021). Natural Language Processing and Language Learning. *The Encyclopedia of Applied Linguistics*.

Miao, X., Qi, H., Zhang, Z., Cao, G., Lin, R., & Lin, W. (2021). Dual Neural Network Classification Based on BERT Feature Extraction for Authorship Verification. In G. Faggioli, N. Ferro, A. Joly, M. Maistro, & F. Piroi (Eds.), *Proceedings of the Working Notes of CLEF 2021 - Conference and Labs of the Evaluation Forum, Bucharest, Romania, September 21st - to - 24th, 2021* (pp. 2069–2072). CEUR-WS.org. http://ceur-ws.org/Vol-2936/paper-182.pdf

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space. In Y. Bengio & Y. LeCun (Eds.), *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. http://arxiv.org/abs/1301.3781

Mikolov, T., Le, Q. V., & Sutskever, I. (2013b). Exploiting Similarities among Languages for Machine Translation. *CoRR*, *abs/1309.4168*. http://arxiv.org/abs/1309.4168

Mikolov, T., Yih, W., & Zweig, G. (2013c). Linguistic Regularities in Continuous Space Word Representations. In L. Vanderwende, H. D. III, & K. Kirchhoff (Eds.), *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA* (pp. 746–751). The Association for Computational Linguistics. https://aclanthology.org/N13-1090/

Min, S., Chen, D., Hajishirzi, H., & Zettlemoyer, L. (2019). A Discrete Hard EM Approach for Weakly Supervised Question Answering. In K. Inui, J. Jiang, V. Ng, & X. Wan (Eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019* (pp. 2851–2864). Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1284

Mintz, M., Bills, S., Snow, R., & Jurafsky, D. (2009). Distant supervision for relation extraction without labeled data. In K. Su, J. Su, & J. Wiebe (Eds.), *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore* (pp. 1003–1011). The Association for Computer Linguistics. https://aclanthology.org/P09-1113/

Mulloni, A., & Pekar, V. (2006). Automatic Detection of Orthographics Cues for Cognate Recognition. *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. http://www.lrec-conf.org/proceedings/lrec2006/pdf/676_pdf.pdf

Navigli, R. (2018). Natural Language Understanding: Instructions for (Present and Future) Use. *IJCAI*, 5697–5702.

Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, *48*(3), 443–453. https://doi.org/https://doi.org/10.1016/0022-2836(70)90057-4

Nguyen, D., & Doğruöz, A. S. (2013). Word Level Language Identification in Online Multilingual Communication. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 857–862. https://aclanthology.org/D13-1084

Nguyen, T. Q., & Chiang, D. (2017). Transfer Learning across Low-Resource, Related Languages for Neural Machine Translation. *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 296–301. https://aclanthology.org/I17-2050

Nigam, K., & Ghani, R. (2000). Analyzing the effectiveness and applicability of co-training. *Proceedings of the ninth international conference on Information and knowledge management*, 86–93.

Noble, W. S. (2006). What is a support vector machine? *Nature biotechnology*, *24*(12), 1565–1567.

Pagliardini, M., Gupta, P., & Jaggi, M. (2018). Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features. *NAACL-HLT*, 528–540.

Palangi, H., Deng, L., Shen, Y., Gao, J., He, X., Chen, J., Song, X., & Ward, R. K. (2016). Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval. *IEEE ACM Trans. Audio Speech Lang. Process.*, *24*(4), 694–707. https://doi.org/10.1109/TASLP.2016.2520371

Pang, B., & Lee, L. (2004). A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain*, 271–278.

Pang, B., & Lee, L. (2005). Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, 115–124.

Patwa, P., Aguilar, G., Kar, S., Pandey, S., PYKL, S., Gambäck, B., Chakraborty, T., Solorio, T., & Das, A. (2020). SemEval-2020 Task 9: Overview of Sentiment Analysis of Code-Mixed Tweets. *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global Vectors for Word Representation. In A. Moschitti, B. Pang, & W. Daelemans (Eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL* (pp. 1532–1543). ACL. https://doi.org/10.3115/v1/d14-1162

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In M. A. Walker, H. Ji, & A. Stent (Eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)* (pp. 2227–2237). Association for Computational Linguistics. https://doi.org/10.18653/v1/n18-1202

Peters, M. E., Ruder, S., & Smith, N. A. (2019). To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks. In I. Augenstein, S. Gella, S. Ruder, K. Kann, B. Can, J. Welbl, A. Conneau, X. Ren, & M. Rei (Eds.), *Proceedings of the 4th Workshop on Representation Learning for NLP, RepL4NLP@ACL 2019, Florence, Italy, August 2, 2019* (pp. 7–14). Association for Computational Linguistics. https://doi.org/10.18653/v1/w19-4302

Pires, T., Schlinger, E., & Garrette, D. (2019). How Multilingual is Multilingual BERT? *ACL*, 4996–5001.

Poulston, A., Waseem, Z., & Stevenson, M. (2017). Using TF-IDF n-gram and word embedding cluster ensembles for author profiling: Notebook for PAN at CLEF 2017. *CEUR Workshop Proceedings*, *1866*.

Prager, J. M. (2000). Linguini: Language Identification for Multilingual Documents. *J. Manag. Inf. Syst.*, *16*(3), 71–102. https://doi.org/10.1080/07421222.1999.11518257

Prager, J. M., Radev, D. R., Brown, E. W., Coden, A., & Samn, V. (1999). The Use of Predictive Annotation for Question Answering in TREC8. In E. M. Voorhees & D. K. Harman (Eds.), *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999*. National Institute of Standards; Technology (NIST). http://trec.nist.gov/pubs/trec8/papers/IBMTrec8QA.ps

Prasad, G., van Schijndel, M., & Linzen, T. (2019). Using Priming to Uncover the Organization of Syntactic Representations in Neural Language Models. In M. Bansal & A. Villavicencio (Eds.), *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019* (pp. 66–76). Association for Computational Linguistics. https://doi.org/10.18653/v1/K19-1007

Qazi, A., Raj, R. G., Tahir, M., Cambria, E., & Syed, K. B. S. (2014). Enhancing Business Intelligence by Means of Suggestive Reviews. *The Scientific World Journal*, *2014*, 1–11.

Qin, H., Zhu, J., & Shen, B. (2021). Weakly-Supervised Question Answering with Effective Rank and Weighted Loss over Candidates. In J. Leskovec, M. Grobelnik, M. Najork, J. Tang, & L. Zia (Eds.), *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021* (pp. 1959–1967). ACM / IW3C2. https://doi.org/10.1145/3442381.3449919

Rahit, K. T. H., Hasan, K. T., Amin, M. A., & Ahmed, Z. (2018). BanglaNet: Towards a WordNet for Bengali Language. *Proceedings of the 9th Global Wordnet Conference*, 1–9. https://aclanthology.org/2018.gwc-1.1

Ram, A., Prasad, R., Khatri, C., Venkatesh, A., Gabriel, R., Liu, Q., Nunn, J., Hedayatnia, B., Cheng, M., Nagar, A., King, E., Bland, K., Wartick, A., Pan, Y., Song, H., Jayadevan, S., Hwang, G., & Pettigrue, A. (2018). Conversational AI: The Science Behind the Alexa Prize. *CoRR*, *abs/1801.03604*. http://arxiv.org/abs/1801.03604

Rama, T. (2016). Siamese Convolutional Networks for Cognate Identification. In N. Calzolari, Y. Matsumoto, & R. Prasad (Eds.), *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan* (pp. 1018–1027). ACL. https://aclanthology.org/C16-1097/

Rani, P., McCrae, J. P., & Fransen, T. (2022). MHE: Code-Mixed Corpora for Similar Language Identification. *Proceedings of the Language Resources and Evaluation Conference*, 3425–3433. https://aclanthology.org/2022.lrec-1.366

Rani, P., Suryawanshi, S., Goswami, K., Chakravarthi, B. R., Fransen, T., & McCrae, J. P. (2020). A Comparative Study of Different State-of-the-Art Hate Speech Detection Methods in Hindi-English Code-Mixed Data. *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, 42–48. https://www.aclweb.org/anthology/2020.trac-1.7

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In K. Inui, J. Jiang, V. Ng, & X. Wan (Eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019* (pp. 3980–3990). Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1410

Reimers, N., & Gurevych, I. (2020). Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. In B. Webber, T. Cohn, Y. He, & Y. Liu (Eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November*

16-20, 2020 (pp. 4512–4525). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.365

Rijhwani, S., Sequiera, R., Choudhury, M., Bali, K., & Maddila, C. S. (2017). Estimating code-switching on Twitter with a novel generalized word-level language detection technique. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1971–1982.

Rohit, G., Dharamshi, E. G., & Subramanyam, N. (2017). Approaches to Question Answering Using LSTM and Memory Networks. In J. C. Bansal, K. N. Das, A. Nagar, K. Deep, & A. K. Ojha (Eds.), *Soft Computing for Problem Solving - SocProS 2017, Volume 1, Bhubaneswar, India, December 23-24, 2017* (pp. 199–209). Springer. https://doi.org/10.1007/978-981-13-1592-3_15

Ruder, S. (2017). An Overview of Multi-Task Learning in Deep Neural Networks. *CoRR*, *abs/1706.05098*. http://arxiv.org/abs/1706.05098

Ruder, S. (2019). *Neural transfer learning for natural language processing* (Doctoral dissertation). NUI Galway.

Ruder, S., Vulic, I., & Søgaard, A. (2019). A Survey of Cross-lingual Word Embedding Models. *J. Artif. Intell. Res.*, *65*, 569–631. https://doi.org/10.1613/jair.1.11640

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533–536.

Sahni, T., Chandak, C., Chedeti, N. R., & Singh, M. (2017). Efficient Twitter sentiment classification using subjective distant supervision. *9th International Conference on Communication Systems and Networks, COMSNETS 2017, Bengaluru, India, January 4-8, 2017*, 548–553. https://doi.org/10.1109/COMSNETS.2017.7945451

Salinca, A. (2016). Business reviews classification using sentiment analysis. *SYNASC*, 247–250.

Salton, G., Wong, A., & Yang, C. (1975). A Vector Space Model for Automatic Indexing. *Commun. ACM*, *18*(11), 613–620. https://doi.org/10.1145/361219.361220

Samardžić, T., Scherrer, Y., & Glaser, E. (2016). ArchiMob - A Corpus of Spoken Swiss German. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 4061–4066. https://www.aclweb.org/anthology/L16-1641

Sánchez-Pi, N., Martí, L., & Garcia, A. C. (2013). Text Classification Techniques in Oil Industry Applications. *International Joint Conference SOCO-CISIS-ICEUTE*, 211–220.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019a). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019b). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, *abs/1910.01108*. http://arxiv.org/abs/1910.01108

Sarkar, D., Zampieri, M., Ranasinghe, T., & Ororbia, A. G. (2021). fBERT: A Neural Transformer for Identifying Offensive Content. In M. Moens, X. Huang, L. Specia, & S. W. Yih (Eds.), *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021* (pp. 1792–1798). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.findings-emnlp.154

Scherrer, Y., Samardžić, T., & Glaser, E. (2019). Digitising Swiss German: how to process and study a polycentric spoken language. *Language Resources and Evaluation*, *53*(4), 735–769.

Schwenk, H. (2018). Filtering and Mining Parallel Data in a Joint Multilingual Space. In I. Gurevych & Y. Miyao (Eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers* (pp. 228–234). Association for Computational Linguistics. https://doi.org/10.18653/v1/P18-2037

Schwenk, H., & Douze, M. (2017). Learning Joint Multilingual Sentence Representations with Neural Machine Translation. In P. Blunsom, A. Bordes, K. Cho, S. B. Cohen, C. Dyer, E. Grefenstette, K. M. Hermann, L. Rimell, J. Weston, & S. Yih (Eds.), *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP@ACL 2017, Vancouver, Canada, August 3, 2017* (pp. 157–167). Association for Computational Linguistics. https://doi.org/10.18653/v1/w17-2619

Schwenk, H., Kiela, D., & Douze, M. (2019). Analysis of Joint Multilingual Sentence Representations and Semantic K-Nearest Neighbor Graphs. *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 6982–6990. https://doi.org/10.1609/aaai.v33i01.33016982

Segonne, V., Candito, M., & Crabbé, B. (2019). Using Wiktionary as a resource for WSD : the case of French verbs. In S. Dobnik, S. Chatzikyriakidis, & V. Demberg (Eds.), *Proceedings of the 13th International Conference on Computational Semantics, IWCS 2019, Long Papers, Gothenburg, Sweden, May 23-27 May, 2019* (pp. 259–270). Association for Computational Linguistics. https://doi.org/10.18653/v1/w19-0422

Selamat, A., & Ching, N. C. (2008). Arabic script documents language identifications using Fuzzy ART. *2008 Second Asia International Conference on Modelling & Simulation (AMS)*, 528–533.

Sennrich, R., Haddow, B., & Birch, A. (2016). Neural Machine Translation of Rare Words with Subword Units. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. https://doi.org/10.18653/v1/p16-1162

Severini, S., Hangya, V., Fraser, A. M., & Schütze, H. (2020). Combining Word Embeddings with Bilingual Orthography Embeddings for Bilingual Dictionary Induction. In D. Scott, N. Bel, & C. Zong (Eds.), *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020* (pp. 6044–6055). International Committee on Computational Linguistics. https://doi.org/10.18653/v1/2020.coling-main.531

Sharma, S., Santra, B., Jana, A., Tokala, S., Ganguly, N., & Goyal, P. (2019). Incorporating Domain Knowledge into Medical NLI using Knowledge Graphs. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 6092–6097. https://doi.org/10.18653/v1/D19-1631

Shiells, K., & Pham, P. (2010). Unsupervised Clustering for Language Identification.

Singh, A. K., & Gorla, J. (2007). Identification of languages and encodings in a multilingual document. *Cahiers du Cental*, *5*, 1.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, 1631–1642.

Søgaard, A., & Goldberg, Y. (2016). Deep multi-task learning with low level tasks supervised at lower layers. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. https://doi.org/10.18653/v1/p16-2038

Sreedhar, M. N., & Parisien, C. (2022). Prompt Learning for Domain Adaptation in Task-Oriented Dialogue. *CoRR*, *abs/2211.05596*. https://doi.org/10.48550/arXiv.2211.05596

Srinivasan, S., Huang, Z., & Kirchhoff, K. (2022). Representation Learning Through Cross-Modal Conditional Teacher-Student Training For Speech Emotion Recognition. *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, 6442–6446. https://doi.org/10.1109/ICASSP43922.2022.9747754

Takase, S., & Kobayashi, S. (2020). All Word Embeddings from One Embedding. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. https://proceedings.neurips.cc/paper/2020/hash/275d7fb2fd45098ad5c3ece2ed4a2824-Abstract.html

Tamura, A., Watanabe, T., & Sumita, E. (2012). Bilingual Lexicon Extraction from Comparable Corpora Using Label Propagation. In J. Tsujii, J. Henderson, & M. Pasca (Eds.), *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea* (pp. 24–36). ACL. https://aclanthology.org/D12-1003/

Tang, H., Sun, X., Jin, B., Wang, J., Zhang, F., & Wu, W. (2021). Improving Document Representations by Generating Pseudo Query Embeddings for Dense Retrieval. In C. Zong, F. Xia, W. Li, & R. Navigli (Eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021* (pp. 5054–5064). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.acl-long.392

Tang, Y., Blincoe, K., & Kempa-Liehr, A. W. (2020). Enriching feature engineering for short text samples by language time series analysis. *EPJ Data Science*, *9*(1), 26.

Tanvir, H., Kittask, C., Eiche, S., & Sirts, K. (2021). EstBERT: A Pretrained Language-Specific BERT for Estonian. In S. Dobnik & L. Øvrelid (Eds.), *Proceedings of the 23rd Nordic Conference on Computational Linguistics, NoDaLiDa 2021, Reykjavik, Iceland (Online), May 31 - June 2, 2021* (pp. 11–19). Linköping University Electronic Press, Sweden. https://aclanthology.org/2021.nodalida-main.2/

Tauchmann, C. (2021). *Advanced Corpus Annotation Strategies for NLP. Applications in Automatic Summarization and Text Classification* (Doctoral dissertation). Technical University of Darmstadt, Germany. http://tuprints.ulb.tu-darmstadt.de/17576/

Tiedemann, J. (1999). Automatic Construction of Weighted String Similarity Measures. *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*. https://aclanthology.org/W99-0626

Tiedemann, J., & Ljubesic, N. (2012). Efficient Discrimination Between Closely Related Languages. In M. Kay & C. Boitet (Eds.), *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 8-15 December 2012, Mumbai, India* (pp. 2619–2634). Indian Institute of Technology Bombay. https://aclanthology.org/C12-1160/

Tillmann, C., Mansour, S., & Al-Onaizan, Y. (2014). Improved Sentence-Level Arabic Dialect Classification. In M. Zampieri, L. Tan, N. Ljubesic, & J. Tiedemann (Eds.), *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects, VarDial@COLING 2014, Dublin, Ireland, August 23, 2014* (pp. 110–119). Association for Computational Linguistics; Dublin City University. https://doi.org/10.3115/v1/W14-5313

Trieschnigg, D., Hiemstra, D., Theune, M., Jong, F., & Meder, T. (2012). An exploration of language identification techniques in the dutch folktale database. *Proceedings of the Workshop on Adaptation of Language Resources and Tools for Processing Cultural Heritage (LREC 2012)*.

Turney, P. D., & Pantel, P. (2010). From Frequency to Meaning: Vector Space Models of Semantics. *J. Artif. Intell. Res.*, *37*, 141–188. https://doi.org/10.1613/jair.2934

Van-Tu, N., & Anh-Cuong, L. (2016). Improving question classification by feature extraction and selection. *Indian Journal of Science and Technology*, *9*(17), 1–8.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017a). Attention is All you Need. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA* (pp. 5998–6008). https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017b). Attention is all you need. *NIPS*, 6000–6010.

Vatanen, T., Väyrynen, J. J., & Virpioja, S. (2010). Language Identification of Short Text Segments with N-gram Models. In N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, & D. Tapias (Eds.), *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2010, 17-23 May 2010, Valletta, Malta*. European Language Resources Association. http://www.lrec-conf.org/proceedings/lrec2010/summaries/279.html

Voss, C., Tratz, S., Laoudi, J., & Briesch, D. (2014). Finding Romanized Arabic Dialect in Code-Mixed Tweets. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, 2249–2253. http://www.lrec-conf.org/proceedings/lrec2014/pdf/1116_Paper.pdf

Wan, A. (2016). Leveraging Data-Driven Methods in Word-Level Language Identification for a Multilingual Alpine Heritage Corpus. *Proceedings of the Workshop on Multilingual and Cross-lingual Methods in NLP*, 45–54. https://doi.org/10.18653/v1/W16-1206

Wang, D., Hu, W., Cao, E., & Sun, W. (2020). Global-to-Local Neural Networks for Document-Level Relation Extraction. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 3711–3721. https://doi.org/10.18653/v1/2020.emnlp-main.303

Wang, D., & Zheng, T. F. (2015). Transfer learning for speech and language processing. *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2015, Hong Kong, December 16-19, 2015*, 1225–1237. https://doi.org/10.1109/APSIPA.2015.7415532

Wang, H., Henderson, J., & Merlo, P. (2019). Weakly-Supervised Concept-based Adversarial Learning for Cross-lingual Word Embeddings. In K. Inui, J. Jiang, V. Ng, & X. Wan (Eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019* (pp. 4418–4429). Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1450

Wang, W., Hosseini, S., Awadallah, A. H., Bennett, P. N., & Quirk, C. (2019). Context-Aware Intent Identification in Email Conversations. *SIGIR*, 585–594.

Werbos, P. J. (1988). Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, *1*(4), 339–356. https://doi.org/10.1016/0893-6080(88)90007-X

Wiese, G., Weissenborn, D., & Neves, M. L. (2017). Neural Domain Adaptation for Biomedical Question Answering. In R. Levy & L. Specia (Eds.), *Proceedings of the 21st Conference on Computational*

Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017 (pp. 281–289). Association for Computational Linguistics. https://doi.org/10.18653/v1/K17-1029

Wieting, J., Bansal, M., Gimpel, K., & Livescu, K. (2016). Towards Universal Paraphrastic Sentence Embeddings. In Y. Bengio & Y. LeCun (Eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. http://arxiv.org/abs/1511.08198

Wray, S. (2018). Classification of Closely Related Sub-dialects of Arabic Using Support-Vector Machines. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. https://aclanthology.org/L18-1580

Xie, J., Girshick, R., & Farhadi, A. (2016). Unsupervised deep embedding for clustering analysis. *International conference on machine learning*, 478–487.

Xie, Q., Huang, J., Du, P., Peng, M., & Nie, J.-Y. (2021). Graph Topic Neural Network for Document Representation. *Proceedings of the Web Conference 2021*, 3055–3065. https://doi.org/10.1145/3442381.3450045

Xu, W., & Rudnicky, A. (2000). Can artificial neural networks learn language models? *Sixth International Conference on Spoken Language Processing, ICSLP 2000 / INTERSPEECH 2000, Beijing, China, October 16-20, 2000*, 202–205. http://www.isca-speech.org/archive/icslp%5C_2000/i00%5C_1202.html

Yang, J., Zhang, Y., & Dong, F. (2017). Neural Word Segmentation with Rich Pretraining. In R. Barzilay & M. Kan (Eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers* (pp. 839–849). Association for Computational Linguistics. https://doi.org/10.18653/v1/P17-1078

Yang, Y., Ábrego, G. H., Yuan, S., Guo, M., Shen, Q., Cer, D., Sung, Y., Strope, B., & Kurzweil, R. (2019). Improving Multilingual Sentence Embedding using Bi-directional Dual Encoder with Additive Margin Softmax. In S. Kraus (Ed.), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019* (pp. 5370–5378). ijcai.org. https://doi.org/10.24963/ijcai.2019/746

Yang, Y., Cer, D., Ahmad, A., Guo, M., Law, J., Constant, N., Ábrego, G. H., Yuan, S., Tar, C., Sung, Y., Strope, B., & Kurzweil, R. (2020). Multilingual Universal Sentence Encoder for Semantic Retrieval. In A. Celikyilmaz & T. Wen (Eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020* (pp. 87–94). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-demos.12

Yang, Y., & Hospedales, T. M. (2017). Trace Norm Regularised Deep Multi-Task Learning. *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. https://openreview.net/forum?id=rknkNR7Ke

Yang, Z., Yang, Y., Cer, D., Law, J., & Darve, E. (2020). Universal Sentence Representation Learning with Conditional Masked Language Model. *CoRR*, *abs/2012.14388*. https://arxiv.org/abs/2012.14388

Yin, W., & Schütze, H. (2016). Learning word meta-embeddings. *ACL*, 1351–1360.

Yin, X., Zhang, W., Zhu, W., Liu, S., & Yao, T. (2020). Improving sentence representations via component focusing. *Applied Sciences*, *10*(3), 958.

Zaidan, O., & Callison-Burch, C. (2014). Arabic Dialect Identification. *Comput. Linguistics*, *40*(1), 171–202. https://doi.org/10.1162/COLI_a_00169

Zaidan, O. F., & Callison-Burch, C. (2014). Arabic dialect identification. *Computational Linguistics*, *40*(1), 171–202.

Zampieri, M., & Gebre, B. G. (2012). Automatic identification of language varieties: The case of Portuguese. *KONVENS2012-The 11th Conference on Natural Language Processing*, 233–237.

Zampieri, M., Gebre, B. G., & Diwersy, S. (2013). N-gram Language Models and POS Distribution for the Identification of Spanish Varieties (Ngrammes et Traits Morphosyntaxiques pour la Identification de Variétés de l'Espagnol) [in French]. *Proceedings of TALN 2013 (Volume 2: Short Papers)*, 580–587. https://aclanthology.org/F13-2010

Zampieri, M., Malmasi, S., Scherrer, Y., Samardzic, T., Tyers, F., Silfverberg, M., Klyueva, N., Pan, T.-L., Huang, C.-R., Ionescu, R. T., et al. (2019). A report on the third VarDial evaluation campaign. *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, 1–16.

Zampieri, M., Malmasi, S., Sulea, O.-M., & P. Dinu, L. A Computational Approach to the Study of Portuguese Newspapers Published in Macau. In: In *Proceedings of the Workshop on Natural Language Processing Meets Journalism (NLPMJ)*. New York, NY, USA, 2016, 47–51.

Zayats, V., Toutanova, K., & Ostendorf, M. (2021). Representations for Question Answering from Documents with Tables and Text. In P. Merlo, J. Tiedemann, & R. Tsarfaty (Eds.), *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021* (pp. 2895–2906). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.eacl-main.253

Zečević, A., & Vujicic-Stankovic, S. (2013). The Mysterious Letter J. *Proceedings of the Workshop on Adaptation of Language Resources and Tools for Closely Related Languages and Language Variants*, 40–44.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An Efficient Data Clustering Method for Very Large Databases. In H. V. Jagadish & I. S. Mumick (Eds.), *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996* (pp. 103–114). ACM Press. https://doi.org/10.1145/233269.233324

Zhang, W., Clark, R. A. J., Wang, Y., & Li, W. (2016a). Unsupervised language identification based on Latent Dirichlet Allocation. *Comput. Speech Lang.*, *39*, 47–66. https://doi.org/10.1016/j.csl.2016.02.001

Zhang, W., Clark, R. A., Wang, Y., & Li, W. (2016b). Unsupervised language identification based on Latent Dirichlet Allocation. *Computer Speech & Language*, *39*, 47–66.

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 649–657.

Zhang, Y., He, R., Liu, Z., Lim, K. H., & Bing, L. (2020). An Unsupervised Sentence Embedding Method by Mutual Information Maximization. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, 1601–1610.

Zhang, Y., Baldridge, J., & He, L. (2019). PAWS: Paraphrase Adversaries from Word Scrambling. *NAACL-HLT*, 1298–1308.

Zhao, W., Peyrard, M., Liu, F., Gao, Y., Meyer, C. M., & Eger, S. (2019). MoverScore: Text Generation Evaluating with Contextualized Embeddings and Earth Mover Distance. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, 563–578.

Zhou, C., Sun, C., Liu, Z., & Lau, F. (2015). A C-LSTM neural network for text classification. *arXiv preprint arXiv:1511.08630*.

Ziser, Y., & Reichart, R. (2018). Pivot Based Language Modeling for Improved Neural Domain Adaptation. In M. A. Walker, H. Ji, & A. Stent (Eds.), *Proceedings of the 2018 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers) (pp. 1241–1251). Association for Computational Linguistics. https://doi.org/10.18653/v1/n18-1112

Zoph, B., Yuret, D., May, J., & Knight, K. (2016). Transfer Learning for Low-Resource Neural Machine Translation. In J. Su, X. Carreras, & K. Duh (Eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016* (pp. 1568–1575). The Association for Computational Linguistics. https://doi.org/10.18653/v1/d16-1163

Zweigenbaum, P., Sharoff, S., & Rapp, R. (2017). Overview of the Second BUCC Shared Task: Spotting Parallel Sentences in Comparable Corpora. *Proceedings of the 10th Workshop on Building and Using Comparable Corpora*, 60–67.