Title	Enhancing knowledge graph completion models and selected biological applications
Author(s)	Mohamed, Sameh K.
Publication Date	2020-06-18
Publisher	NUI Galway
Item record	http://hdl.handle.net/10379/16048

NATIONAL UNIVERSITY OF IRELAND GALWAY

ENHANCING KNOWLEDGE GRAPH COMPLETION MODELS AND SELECTED BIOLOGICAL APPLICATIONS

by
Sameh K. Mohamed

A thesis submitted in partial fulfilment for the degree of Doctor of Philosophy

in the

College of Engineering and Informatics
Data Science Institute

Supervisors: Dr. Vít Nováček, Prof. Mathieu d'Aquin

Galway, Ireland, June 2020

Wings are a constraint that makes
it possible to fly.
— Robert Bringhurst

To family, friends and colleagues . . .

Acknowledgements

I would like to thank my supervisor Dr. Vít Nováček and my former colleague Dr. Pierre-Yves Vandenbussche for their support throughout my PhD. They continuously guided me to enhance my research and writing skills. I would like also to thank Dr. Pasquale Minervini and Dr. Luca Costabello for their enormous support and guidance during the early stages of my PhD studies.

I also want to express my sincere gratitude to my friends: Dr. Waleed Yousef, Dr. Mohamed Alaggan, Dr. Deyaaeldeen Almahallawi, Ahmed M. Farrag for their valuable support and advices. I believe I would have never reached this point without their valuable encouragement and support.

Last but not the least, I would like to thank my family: my parents, brother, sisters, my wife and my son for being by my side all the time.

The work described in this thesis has been supported by he TOMOE project funded by Fujitsu Laboratories Ltd., Japan and Insight Centre for Data Analytics at the National University of Ireland Galway, Ireland (supported by the Science Foundation Ireland grants (12/RC/2289 and 12/RC/2289_P2).

Galway, 9 January 2020 S. K. Mohamed

Preface

It was my pleasure to watch Sameh grow during the quest for his PhD, a quest that has now come to fruition with this thesis. Building on the solid grounds of comprehensive state of the art research in knowledge graphs, knowledge base completion and associated machine learning fields, Sameh has pushed the boundaries of knowledge graph embedding models and their applications to knowledge base completion. Yet what I see as perhaps the most significant accomplishment of his PhD research is a systematic exploration of current challenges in biomedical knowledge discovery and a thorough study of corresponding opportunities in applying the knowledge graph technology to these important problems with many practical implications. I believe Sameh's PhD research has brought state of the art results in these areas, as demonstrated by a good few high-profile publications already. I feel privileged that I could contribute a little to Sameh's successful journey through the often daunting waters of establishing himself as an independent researcher. I am looking forward to him making the results presented in this document obsolete by still more exciting and valuable research in the years to come.

Galway, 12 January 2020 Vít Nováček

Abstract

Knowledge completion is the task of extending knowledge graphs to enhance the quality of systems relying on them. In recent years, various knowledge completion techniques were developed to model knowledge graphs using different of features such as graph features and embeddings. These models showed complementary capabilities where graph feature model excelled in terms of interpretability and knowledge graph embedding models excelled in terms of accuracy and scalability. Despite the advances achieve by these models in extending knowledge graphs, they still have predictive accuracy. The evaluation of the capabilities of these models was also limited to standard benchmarks with no real use case scenarios especially. In this thesis, study both graph feature models and knowledge graph embedding models and their use in extending knowledge graph and we propose new models for both approaches. We also present and evaluation of the capabilities of knowledge graph embedding models in multiple real life biological use cases.

First, we examine the current limitation of the poor feature representations in graph feature models and we propose a new graph feature model, the DSP model, which offers richer feature representations. We show by experimental evaluation that our new proposed model outperforms the current state-of-the-art models on a standard NELL based benchmark with no extra added computational cost. Secondly, We study knowledge graph embedding models where we investigate their training pipeline and examine its different paths and their effects on the models accuracy and scalability. We then propose a new tensor factorisation based knowledge graph embedding model, the TriVec model, which models embedding using multiple vectors. We show that this representation allows our model to dynamical encode embedding interactions of different types of symmetric and asymmetric relationships which results in accuracy improvements. We show by experimental evaluation on different standard benchmarks that our model outperforms other state-of-the-art methods in terms of accuracy.

We also study the potential uses of knowledge graph embedding models in biological uses cases where we demonstrate their different capabilities in predicting links in biological networks, measure similarity between biological concepts and clustering biological entities. We then present three use case scenarios of the use of knowledge graph embedding models in predicting drug protein targets, polypharmacy side-effect and tissue-specific protein functions where we show that they knowledge graph embedding models represented by our newly proposed model, the TriVec model, outperform state-of-the art techniques in these use cases.

Key words: knowledge completion, knowledge graph embeddings, bioinformatics

Contents

List of Figures

List of Tables

Introduction Part I

1 Introduction

Knowledge graphs are data modelling techniques which are used to model relational data. This technique of graph-based modelling of information has a long history in logic and artificial intelligence (Davis et al. 1993). Early approaches introduced the idea as a framework for general knowledge representation (Minsky 1974), or a means for representing semantic networks (Sowa 2006). Later, the technique has been adopted in different fields with multiple given names like Knowledge Graphs (KGs) in the field of artificial intelligence (Berg 1993), Heterogeneous Information Networks (HINs) in the field of information retrieval (Xiang et al. 2009), Resource Description Framework Graphs (RDF graphs) in the semantic web community (Brachman & Levesque, W3C 2004, 1997) and Semantic Networks in the community of cognitive sciences (Sowa 2006). In recent years, graph-structured knowledge bases (*i.e.* knowledge graphs) became a popular means for modelling linked data at scale where they were adopted in different industrial and academic settings. They were used to model data from different fields such as biological networks (Dumontier et al. 2014), lexical information (Miller 1995) and general human knowledge (Mitchell et al. 2015). They were also used to support different types of applications such as question answering systems (Ferrucci et al. 2010), biological discoveries (Mohamed, Nováček & Nounu 2019) and digital assistants (Qian 2013).

1.1 The problem

The wide adoption of knowledge graphs for modelling relational web data enabled the creation of multiple knowledge bases such as Wikidata (Vrandecic & Krötzsch 2014), DBpedia (Lehmann & et. al. 2014) and YAGO (Mahdisoltani et al. 2015). These knowledge bases were then used in multiple applications as a source of information as they are easily readable for both humans and machines. Despite the usefulness of such knowledge bases, they are incomplete (Razniewski et al. 2016). This incompleteness of knowledge bases reduces the accuracy of predictive models that depend on them. This encouraged researchers to develop different approaches to perform knowledge graph completion such as graph feature models and knowledge graph embedding models (Nickel, Murphy, Tresp & Gabrilovich 2016*a*).

Despite the recent advances in the development of knowledge completion methods (Wang et al. 2017), they still suffer from limited accuracy (Kadlec et al. 2017). The use of knowledge graph embedding models is also limited to general human knowledge prediction tasks on knowledge graph benchmarks such as YAGO (Nickel et al. 2012) and NELL (Mohamed & Novácek 2019), and a study of the capabilities of these models on other real life use cases in different domain *e.g.* biological domain, is missing.

In this thesis, we propose new techniques to enhance the accuracy of the state-of-the-art knowledge graph embedding and graph feature models. We then show that our proposed models can provide better predictive accuracy with no extra added computational cost. We also provide a set of real life use cases for applications of knowledge graph embeddings in the biological domain where we use our proposed knowledge graph embedding model to perform different predictive and analytical tasks on biological data.

1.2 Methods

In this thesis, we investigate the use of graph feature models and embedding models for predicting new facts in knowledge graphs. First, we study the graph feature models such as the path ranking algorithm (PRA) (Lao & Cohen 2010*a*) and the subgraph feature extraction model (Gardner & Mitchell 2015) which use connected paths as features to predict links in knowledge graphs. We show that these models can not then operate in the absence of these connecting paths features. We then propose a new method which uses both connected paths and subgraph paths to allow predictions in sparse knowledge graphs.

Despite the accuracy enhancements achieved by our new graph features model, it suffered from limited scalability as other graph feature models due to dependency on time-consuming path exploration procedures. Other studies (Toutanova & Chen, Nickel, Murphy, Tresp & Gabrilovich 2015, 2016*b*) have also suggested that these models have inferior accuracy compared to knowledge graph embedding based models in knowledge graph predictive tasks. We therefore study the knowledge graph embedding models where we analyse their training pipeline and investigate the effects of the different training components on the models' accuracy and scalability. We then propose a new knowledge graph embedding model, the TriVec model (Mohamed & Novácek 2019), which uses multiple vectors to model embedding interactions and we compare it to other state-of-the-art knowledge graph embedding methods. Finally, we showcase the predictive capabilities of our proposed multi-vector approach in selected biological applications such as predicting drug targets, predictive polypharmacy side-effects and predicting tissue-specific protein functions.

1.3 Experiments and Results

We first perform an experimental evaluation to compare our proposed graph feature model, the DSP model, with the state-of-the-art models such as the PRA and SFE models. We execute

the evaluation on a NELL based benchmark (Gardner & Mitchell 2015) where we use two different evaluation configurations in the presence and absence of connecting paths. We then show that our proposed method outperforms all other approaches in terms of both the mean average precision (MAP) and mean reciprocal rank (MRR) metrics.

Secondly, we study the training of knowledge graph embedding (KGE) methods where we execute different experiments to assess the effects of the different training components on the models' accuracy and scalability. We then show that the different types of training objectives and negative sampling strategies have a significant effect on both the accuracy and scalability of the models. We also show that the accuracy of KGE models is sensitive to the training hyperparameters such as the embedding size and number of training iterations.

Thirdly, we perform an experimental evaluation of our proposed multi-vector knowledge graph embedding model where we compare it to other state-of-the-art method in a link prediction task on different standard benchmarking datasets. Our results show that our proposed approach achieves better predictive accuracy (in terms of MRR and Hits@10) compared to other KGE methods on 5 out of 6 of the used benchmarking datasets.

Finally, we assess the predictive accuracy of our proposed multi-vector KGE model on selected biological applications such as predicting drug targets, predictive polypharmacy side-effects and predicting tissue-specific protein functions. In these experiments, we compare our method to other KGE models and state-of-the-art biological predictive models corresponding to each problem. In all of our experiments, the results show that our proposed method, the TriVec model, outperform all other approaches in terms of the are under the ROC and precision recall curves.

1.4 Contributions

The work discussed in this thesis provides enhancements to the design of knowledge graph embedding models and their potential applications in the field of bioinformatics. The contribution described in this thesis are listed as follows:

- **A new graph feature model for knowledge completion** We propose a new graph feature model which uses a combination of subgraph paths and connecting paths in knowledge graphs to learn new links between entity pairs in the knowledge graph. We show with experimental evaluation that our new model outperforms the current state-of-the-art approaches in terms of both MAP and MRR on a standard benchmarking dataset.

- **Analysis of the KGE training pipeline** We analyse the effects of the different components of KGE models on their accuracy and scalability. We then suggest changes to the current default training components of KGE models that can enhance their predictive accuracy while preserving their high scalability.

- **A new knowledge graph embedding model** We provide a new knowledge graph embedding model which uses tensor factorization on multiple vectors to model facts in knowledge graphs. Our proposed model uses a combination of symmetric and asymmetric interactions to model embedding interactions. We show using experimental evaluation that our proposed model outperforms current state-of-the-art methods in terms of the predictive accuracy on multiple standard benchmarks.

- **Showcasing the capabilities of KGE models in modelling biological systems** We provide a detailed study of the potential predictive and analytical capabilities of KGE models in modelling complex biological systems. We also discuss possible limitations of the models and suggest different strategies for tackling these limitations.

- **Use cases of KGE model in biological applications** We provide three example use cases for utilizing KGE models to predict infer biological knowledge and complete knowledge about biological systems. These use cases are predicting protein drug target, predictive tissue-specific side-effects and predicting polypharmacy side-effects where we show that KGE models can outperform current state-of-the-art methods in these different predictive tasks.

1.5 Outline

The rest of this thesis is structured as follows:

- **Chapter 2** presents the different concepts and notation used throughput this document where it provides a brief introduction to knowledge graphs their construction process and their applications. It also provides a brief introduction to the knowledge graph completion task and the different methods proposed to tackle it such as graph feature models and knowledge graph embedding models.

- **Chapter 3** presents the current graph feature models and discusses their limitations. It also presents our newly proposed graph feature model, the DSP model and discuss it design and utilised path feature types. Finally, it provides a comparative experimental evaluation of the DSP model and other graph feature models such as the PRA and SFE models on a NELL based benchmark (Mohamed et al. 2018).

- **Chapter 4** presents a study of the training strategies of KGE models and the effects of different KGE training components on the models' accuracy and scalability. First, it discusses knowledge graph embedding models and their different training objectives. It then examines the loss functions, negative sampling and training hyperparameters of KGE models and their effects on the performance of KGE models (Mohamed, Novácek, Vandenbussche & Muñoz 2019).

- **Chapter 5** presents our new proposed knowledge graph embedding models, the TriVec model. First, it discusses the evolution of tensor factorisation based knowledge graph embed-

ding models. It then presents our new technique and shows its new approach for modelling embeddings. Finally, it provides a comparative experimental evaluation of our approach and other KGE models on different standard benchmarking datasets using the standard link prediction evaluation pipeline (Mohamed & Novácek 2019).

- **Chapter 6** presents a study of the potential uses of KGE models in biological applications. First, it discusses the evolution of network based predictive models in the biological domain. It then presents three example case studies: predicting drug targets, predictive polypharmacy side-effects and predicting tissue-specific protein functions. It then uses these case studies to demonstrate the different predictive and analytical capabilities of KGE models. The study also discusses the challenges and limitation that face the adoption of KGE models in biological applications.

- **Chapters 7 and 8** present detailed studies of the application of our TriVec model and other KGE models in the prediction of drug targets and tissue-specific protein functions respectively (Mohamed, Nováček & Nounu, Mohamed 2019, 2020).

- **Chapter 9** presents the conclusions of our studies and the future research directions that we intend to pursue to extend our works.

1.6 Publications

In the following, we list the set of publications produced in relation to the research topics discussed in this thesis.

Publications:

1) Biological applications of knowledge graph embedding models.
Sameh K. Mohamed, Vít Nováček and Aayah Nounu
Briefings in Bioinformatics Journal | 2020

2) On the influence of training objectives and hyperparmaters tuning on knowledge graph embeddings accuracy and scalability.
Sameh K. Mohamed and Vít Nováček
Future Generation Computer Systems Journal | In review

3) Discovering protein drug targets using knowledge graph embeddings.
Sameh K. Mohamed, Vít Nováček and Aayah Nounu
Bioinformatics Journal | 2020

4) Predicting tissue-specific protein functions using multi-part tensor decomposition.
Sameh K. Mohamed
Information Sciences Journal | 2020

5) Link prediction using multi part embeddings.
Sameh K. Mohamed and Vít Nováček
European Semantic Web Conference (ESWC) | 2019

6) Loss Functions in Knowledge Graph Embedding Models.
Sameh K. Mohamed, Emir Muñoz, Pierre-Yves Vandenbussche and Vít Nováček
Deep Learning for Knowledge Graphs Workshop (DL4KG@ESWC) | 2019

7) Unsupervised Hierarchical Grouping of Knowledge Graph Entities.
Sameh K. Mohamed
Large Scale RDF Analytics Workshop (LASCAR@ESWC) | 2019

8) Knowledge base completion using distinct subgraph paths.
Sameh K. Mohamed, Pierre-Yves Vandenbussche and Vít Nováček
Annual ACM Symposium on Applied Computing (ACM-SAC) | 2018

9) Identifying equivalent relation paths in knowledge graphs.
Sameh K. Mohamed, Emir Muñoz, Pierre-Yves Vandenbussche and Vít Nováček
nternational Conference on Language, Data and Knowledge (LDK) | 2017

2 Background

In this chapter, we review the core concepts and terminologies used in this thesis. We study knowledge graphs, their applications and how they are built. We also discuss the different types of predictive tasks which are applied on knowledge graphs. Finally, we discuss the design and properties of both graph feature models and knowledge graph embedding models.

2.1 Knowledge Graphs

A knowledge graph is a data modelling technique that models linked data as a graph, where the graph's nodes represent data entities and its edges represent the relations between these entities. In recent years, knowledge graphs became a popular means for modelling relational data where they were adopted in various industrial and academic applications such as semantic search engines (Qian 2013), question answering systems (Ferrucci et al. 2010) and general knowledge repositories (Mitchell et al. 2015). They were also used to model data from different types of domains such as general human knowledge (Mitchell et al. 2015), lexical information (Miller et al. 1990) and biological systems (Dumontier et al. 2014).

Knowledge graphs model facts as subject, predicate and object (SPO) triplets/triples, where subjects and objects are the knowledge entities and predicates are the knowledge relations. In this context, the subject entity is associated to the object entity with the predicate relation *e.g.* (*Paris*, *capital-of*, *France*).

2.1.1 Construction

Knowledge graphs are generated using different techniques, which affects their quality, accuracy and completeness. These techniques can be categorized as follows:

- *Manual curation*, a process where a group of experts generate the set of knowledge graph facts (triplets), which lead to knowledge graphs with high accuracy. However, the process of manually curating knowledge graphs does not scale as it depends on human

experts.

- *Collaborative content*, a process where facts are generated by an open group of volunteers, which is a faster process than the curated approach as it depends on a wider group of volunteers. However, this process is still slow and it can introduce some inaccurate information as volunteers may not be experts in the specific domain of generated facts.

- *Automatic semi-structured data processing*, a process where facts are extracted automatically from semi-structured data such as Wikipedia info-boxes. This technique outperforms previous techniques in terms of scalability, as it depends on automated procedures that can be scaled by extending computational resources. However, efficiency, accuracy and completeness of such a technique depend on the quality of the used data sources and the efficiency of the used algorithms.

- *Automatic unstructured data processing*, a process where facts are extracted automatically from unstructured textual data using natural languages processing approaches. This also depends on the efficiency of the used textual sources as in the previous approach.

Knowledge graphs can also be constructed using combined manual and automatic techniques to benefit from the features of both techniques.

2.1.2 Examples and Applications

Recently, knowledge graphs are adopted to model information from different fields, including general human knowledge (Vrandecic & Krötzsch 2014), biological systems (Dumontier et al. 2014) and language lexical information (Miller 1995). In the following, we present a set of examples of the currently popular knowledge graph datasets like:

- *Wikidata* (Vrandecic & Krötzsch 2014) (successor of *Freebase* (Bollacker et al. 2008)), a knowledge graph dataset containing facts about general human knowledge. Wikidata is created by a group of collaborative volunteers[1], and it is based on information from Wikipedia article from different languages.

- *WordNet* (Miller 1995), a knowledge graph dataset that contains facts about language lexical information, which is manually created by a group of experts[2], and it was build to model facts about English language.

[1] Wikidata currently has a group of 2,808,142 register users according to statistics at https://www.wikidata.org/wiki/Special:Statistics

[2] Wordnet dataset project started in 1985 by George A. Miller in the Princeton University Department of Psychology, and is currently housed in the Department of Computer Science, according to http://wordnet.princeton.edu/wordnet/about-wordnet/

- *YAGO* (Mahdisoltani et al. 2015), a general knowledge graph dataset derived from Wikipedia, WordNet (Miller 1995) and GeoNames[3], where facts are automatically extracted from sources.
- *NELL* (Mitchell et al. 2015), a general knowledge graph dataset constructed by automatically extracting information from Web pages. NELL enriches its facts using two steps, first it extracts facts by applying natural language processing techniques over content of Web pages. It then applies knowledge base inference to extend its facts based on the existing knowledge.
- *Bio2RDF* (Dumontier et al. 2014) a biological knowledge base which includes information from different medical and biological database in an RDF format.

Knowledge graphs have been used by different systems and applications as they provide rich semantics for structured information. They have been used in popular search engines such as Google, to enhance semantics of search results, where its search engine benefits from facts in the Google Knowledge Graph (Singhal 2012). Microsoft also used its knowledge graph *i.e.* Satori (Qian 2013), to provide richer semantics of results of its Bing search engine. Furthermore, knowledge graphs were used to support question answering systems like IBM's Watson (Ferrucci et al. 2010), which has used popular knowledge graphs such as YAGO and Freebase to win the game of *Jeopardy* against human experts. They also play an essential rule in providing information to digital assistants like Google Now, Siri, Cortana, and Alexa.

Further discussion of the properties and uses of knowledge graphs can be found in (Nickel, Murphy, Tresp & Gabrilovich 2016*b*).

2.2 Knowledge Graph Incompleteness

Despite the large volume of information stored in currently available knowledge graphs, they are still incomplete, and this incompleteness affects the accuracy of predictive models which rely on them (Razniewski et al. 2016). This has encouraged research into completing these knowledge graphs by predicting new facts using the currently available knowledge. In the following, we discuss the objective of extending knowledge graphs and its related task where we discuss two example tasks: the knowledge graph completion and link prediction tasks. We also discuss the metrics we use in the evaluation of predictive models in the following chapters of this thesis.

2.2.1 Extending Knowledge Graphs

A complete knowledge graph *i.e.* a knowledge clique, in its optimum form, contains all possible connections between nodes for all relations and the true label of each of these connections.

[3]GeoNames is geographical database that contain information about places, cities and countries. http://www.geonames.org/

Table 2.1 – Statistics of popular benchmarking knowledge graph datasets. The sparsity column represent the ratio between the existing triplets count and the clique size.

Dataset	Entities	Relations	Triplets	Clique Size	Sparsity
WN18	41k	18	151k	30258M	0.0000050
WN18RR	41k	11	93k	18491M	0.0000050
FB15k	15k	1k	610k	225000M	0.0000030
FB15k-237	15k	237	300k	1066500M	0.0000003
YAGO10	123k	37	1.1M	5006100M	0.0000002

This version, the knowledge clique, however, is impractical due to the density and complexity of relations in knowledge graphs. For example, Table 2.1 provides a list of popular knowledge graph benchmarking datasets and statistics about their triplet count, potential clique size and the sparsity ratio of its graph. Although these benchmarks represent only a part of their respective original knowledge graphs, their clique sizes are huge - up to 5000 billion triplets. In real-life applications and commercial knowledge graphs such a clique is infeasible; therefore, extending knowledge graphs is formalized in different tasks which aim to enhance the current versions of knowledge graphs in different ways. In this thesis, we focus on two main tasks: knowledge graph completion and link prediction which we discuss in the following subsections.

2.2.2 Knowledge Graph Completion Task

The task of knowledge graph completion is the process of ranking possible graph assertions (triplets) according to their true nature (true or false). Given a knowledge graph G which consists of the set of entities $\mathcal{E}$, the set of relations $\mathcal{R}$ and the set of known positive triplets T_G, the task of knowledge graph completion is defined such that given a set of triplets T_U which does not intersect with the graph triplets ($T_G \cap T_U = \phi$), the objective is then to rank the set of triplets T_U such that for each triplet $t \in T_U$, if t is true, then t is ranked higher than all other false triplets. In practice, this objective is commonly evaluated independently per relation in the set of relations R_U which represents all the relation instances included in the triplets in the set T_U, where the final evaluation score represents the average predictive accuracy of the model on all relations in the investigated set triplets T_U. The evaluation is executed independently for each group of triplets T_r which consists of all triplets with the relation r and $T_r \subset T_U$.

The time complexity of the evaluation of predictive models is $\mathcal{O}(N)$ where N is the size of the investigated set of triplets T_U. Further details and examples on the evaluation of predictive models on the knowledge graph completion task is found in Chapter 3. In this chapter, we discuss a new model for extending knowledge graphs where we evaluate this model compared to other state-of-the-art approaches on the task of knowledge graph completion using a NELL-based benchmarking knowledge graph.

2.2.3 Link Prediction Task

Extending knowledge graphs is achieved using the general form of the knowledge completion task as previously discussed. It can also be achieved using the link prediction task. Unlike the knowledge graph completion task, the objective of link prediction task is similar to question answering. The link prediction task is defined as follows: given a knowledge graph G which consists of a set of known positive triplets T_G, set of relations R_G, set of entities E_G and another set of known positive triplets T_U which consists of entities and relations from the sets E_G and R_G respectively, and does not intersect with the graph triplets ($T_G \cap T_U = \phi$), the objective is then to provide a set of $2N$ ranks where N is the size of the investigated set of triplets T_U. These ranks are generated such that for each triplet $t \in T_U, t = (s, p, o)$ two ranks are generated for the two sets of triplets: $\cup_{o' \neq o \wedge o' \in E_G \wedge l(s,p,o')=0}(s, p, o')$ and $\cup_{s' \neq s \wedge s' \in E_G \wedge l(s',p,o)=0}(s', p, o)$, where $l(s, p, o) = 1$ if the triplet (s, p, o) is true and 0 otherwise. These two ranks can be viewed as two question as follows:
1) Which object entities are associated with the subject entity s through the relation p?
2) Which subject entities are associated with the object entity o through the relation p?

The predictive models are then expected to provide two ranks of the possible answers for the two queries where each subject answer (s') and object answer (o') are given a high score if they are true and a low score otherwise. This task is traditionally used for question answering where it can resemble questions such as "Who are Alice's friends?" and "Where was Bob born?". These questions can be translated to the queries $(?, \text{FriendOf}, \text{Alice})$ and $(\text{Bob}, \text{BornIn}, ?)$ where a link prediction model processes them.

The evaluation of the link prediction task is a time-consuming task as it has a time complexity of $\mathcal{O}(NM)$ where N is the size of the investigated set of triplets T_U and M is the number of entities in the set E_G. However, with advances of GPU computing and matrix operation modules, this process can be executed within $\mathcal{O}(N)$ time complexity where the evaluation of each rank is executed once using vectorization capabilities in matrix operation modules. Further details and examples on the link prediction task and its evaluation protocol is found in Chapter 5. In this chapter, we introduce a new predictive model for extending knowledge graphs and we compare it to other state-of-the-art models in terms of predictive accuracy on the task of link prediction on knowledge graphs using the benchmarking datasets included in table 2.1.

2.2.4 Evaluation Metrics

The different tasks for extending knowledge graphs are evaluated as ranking problems and solved using ranking models. Learning to rank models are evaluated using different ranking measures including Mean Average Precision (MAP), Normalised Discounted Cumulative Gain (NDCG), and Mean Reciprocal Rank (MRR) (Liu 2011). Below we discuss MAP, MRR and the Hits@k metrics that we use in this thesis where the objective is to rank entries for a set of

queries $Q = \{q_1, q_2, \ldots, q_n\}$.

Mean Average Precision (MAP): MAP is a ranking measure that evaluates the quality of a rank depending on the whole rank of its true (relevant) elements. First, we need to define Precision at position k ($P@k$):

$$P@k(q, l) = \frac{\sum_{i \leq k} \mathbf{I}(l, x_i)}{k},$$

where $x \in q$ and $\mathbf{I}(l, x)$ is an indicator function that is equal to 1 when x is a relevant element and 0 otherwise.

The Average Precision (AP) is defined by:

$$AP(q, l) = \frac{\sum_{i=1}^{n} P@k(q, l) \cdot \mathbf{I}(l, x_i)}{n_1},$$

where n is the total number of objects associated with query q, and n_1 is the number of objects with label one. The MAP is then defined as the mean of AP over all queries Q:

$$\text{MAP}(Q, l) = \frac{\sum_{i=1}^{n} AP(q, l)}{n}. \tag{2.1}$$

Mean Reciprocal Rank (MRR): The Reciprocal Rank (RR) is a statistical measure used to evaluate the response of ranking models depending on the rank of the first correct answer. The MRR is the average of the reciprocal ranks of results for different queries in Q. Formally, MRR is defined as:

$$\text{MRR} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\mathcal{R}(\mathbf{x}_i, f)},$$

where x_i is the highest ranked relevant item for query q_i. Values of RR and MRR have a maximum of 1 for queries with true items ranked first, and get closer to 0 when the first true item is ranked in lower positions. Therefore, we can define the MRR error as $1 - \text{MRR}$. This error starts from 0 when the first true item is ranked first, and increases towards 1 for less successful rankings.

Hits@k: This metric represents the number of correct elements predicted among the top-k elements in a rank, where we commonly use Hits@1, Hits@3 and Hits@10. This metric is defined as follows:

$$\text{Hits@k}(q, l, k) = \sum_{i=0}^{i=k} \mathbf{I}(l, x_i).$$

2.3 Graph Feature Models

Graph feature models are knowledge graph completion models which use graph patterns *e.g.* paths, subgraphs and neighbour nodes, as features. These models were specialised in knowledge graph completion where they learn to score graph facts using different graph patterns as features.

One of the earliest graph feature models is the Path Ranking Algorithm (PRA) (Lao & Cohen 2010*b*), which uses paths connecting pairs of nodes as indicating features to predicting new direct links between these nodes. It first extracts connecting paths between two nodes using random walks, it then uses these paths as features to predict new links between nodes in the graph. Despite the novelty of such an approach, it had limited representation of path features between nodes since it only used one-directional paths from source to target nodes. Further extensions to the PRA models introduced the use of backward random walks (Lao et al. 2015) which provided richer feature representation of node pairs. This resulted in enhancement in the predictive accuracy of the PRA model, however, the new approach still suffered from high rates of false positives (Gardner et al. 2013). Later approaches introduced the use of node embeddings (Gardner et al. 2013) and richer connecting path representations (Gardner & Mitchell 2015) for better modelling of node pairs' features.

In the following we discuss the popular graph feature types used in state of the art graph feature models.

2.3.1 Path feature types

Graph feature model use different types of path-based features which can be represented in different ways. In the following, we describe the set of path feature types used by popular graph feature models.

- **Connecting paths** Paths which connect pairs of nodes. These paths are used as a feature representation of these pairs in order to learn new links between them. A path in this feature type is represented as set of relations which traverse from one node to the other in node pair. This feature type was introduced by the PRA model and later used by its different variations.

- **ANYREL paths** A path feature type used in the SFE model (Gardner & Mitchell 2015), which represents a transformation of connecting paths and acts as a relaxations of the path representation to allow intersection of similar connecting paths. For example, given a connecting path ⟨ a, b ⟩, the ANYREL version of this path is a set of two path ⟨ ANYREL, b ⟩ and ⟨ a, ANYREL ⟩, which allows more intersections between other different paths with the relation from the ANYREL path. This supports a richer representation of connecting paths, allowing for more intersections through the ANYREL wild card that replaces relation instances in connecting paths.

Further discussion on examples of graph feature models and their path features is included in Chapter 3 where we discuss a new graph feature model, the DSP model, for extending knowledge graphs which uses a combination of ANYREL paths and subgraph based paths as features. In this chapter, we also evaluate our proposed model compared to other state-of-the-art approaches on the task of knowledge graph completion using a NELL based benchmarking knowledge graph.

2.4 Knowledge Graph Embeddings

Knowledge graph embedding models learn low rank vector representation *i.e.* embeddings, for graph entities and relations. In the link prediction task, they learn embeddings in order to rank knowledge graph facts according to their factuality. The process of learning these embeddings consists of different phases. First, they initialise the embeddings with random noise. These embeddings are then used to produce scores for a set of true and corrupted facts, where a score of a fact is generated by computing the interaction between the fact's subject, predicate and object embeddings using a model dependent scoring function. Finally, embeddings are updated by a training loss that usually represents a min-max loss, where the objective is to maximise true facts scores and minimise false facts scores.

We define our knowledge graph embedding notations as follows: for any given knowledge graph, E is the set of all entities, R is the set of all relations *i.e.* predicates, N_e and N_r are the numbers of entities and relations respectively, T is the set of all known true facts, e and w are matrices of sizes $N_e \times K$ and $N_r \times K$ respectively that represent entities and relations embeddings of rank K, ϕ_{spo} is the score of the triple (s, p, o), and $\mathcal{L}$ is the model's training loss.

In the following, we discus the different components of the training pipeline of knowledge graph embedding models including scoring function, negative sampling techniques and loss function types.

2.4.1 Scoring Functions

Knowledge graph embedding models generate scores for facts using model-dependent scoring functions that compute interactions between facts' components embeddings. These functions use different approaches to compute these embeddings interactions such as distance between embeddings (Bordes et al. 2013), tensor factorisation (Trouillon et al. 2016) and embeddings convolutional filters (Dettmers et al. 2018).

In the following, we present some of these approaches and we specify some examples of knowledge graph embedding models that use them.

• *Distance-based embeddings interactions*: The Translating Embedding model (TransE) (Bordes et al. 2013) is one of the early models that use distance between embeddings to generate triple

scores. It interprets triple's embeddings interactions as a linear translation of the subject to the object such that $e_s + w_p = e_o$, and generates a score for a triple as follows:

$$\phi_{spo}^{\text{TransE}} = \| e_s + w_p - e_o \|_{l1/l2}, \tag{2.2}$$

where true facts have zero score and false facts have higher scores. This approach provides scalable and efficient embeddings learning as it has linear time and space complexity. However, it fails to provide efficient representation for interactions in one-to-many, many-to-many and many-to-one predicates as its design assumes one object per each subject-predicate combination.

• *Factorisation-based embedding interactions*: Interactions based on embedding factorisation provide better representation for predicates with high cardinality. They have been adopted in models like DistMult (Yang et al. 2015*b*) and ComplEx (Trouillon et al. 2016). Older models such as the Canonical PARAFAC (CP) model (Hitchcock 1927) which use vector interactions as a sum of products which is defined as follows:

$$\phi_{spo}^{\text{CP}} = \sum_{k=1}^{K} e1_{s_k} w_{p_k} e2_{o_k} \tag{2.3}$$

where $e1$ and $e2$ represent entities matrices in the subject and objects forms, where the model have two different representations of each entities in its subject and object forms.

On the other hand, the DistMult model uses the bilinear product of embeddings of the subject, the predicate, and the object as their interaction, and its scoring function is defined as follows:

$$\phi_{spo}^{\text{DistMult}} = \sum_{k=1}^{K} e_{s_k} w_{p_k} e_{o_k} \tag{2.4}$$

where e_{s_k} is the k-th component of subject entity s embedding vector e_s. DistMult achieved a significant improvement in accuracy in the task of link prediction over models like TransE. However, the symmetry of embedding scoring functions affects its predictive power on asymmetric predicates as it cannot capture the direction of the predicate. On the other hand, the ComplEx model uses embedding in a complex form to model data with asymmetry. It models embeddings interactions using the the product of complex embeddings, and its scores are defined as follows:

$$\phi_{spo}^{\text{ComplEx}} = \text{Re}(\sum_{k=1}^{K} e_{s_k} w_{p_k} \overline{e}_{o_k}) = \sum_{k=1}^{K} e_{s_k}^r w_{p_k}^r e_{o_k}^r + e_{s_k}^i w_{p_k}^r e_{o_k}^i + e_{s_k}^r w_{p_k}^i e_{o_k}^i - e_{s_k}^i w_{p_k}^i e_{o_k}^r \tag{2.5}$$

where $Re(x)$ represents the real part of complex number x and all embeddings are in complex form such that $e, w \in C$, e^r and e^i are respectively the real and imaginary parts of e, and $\overline{e}_o$ is the complex conjugate of the object embeddings e_o such that $\overline{e}_o = e_o^r - i e_o^i$ and this introduces

asymmetry to the scoring function. Using this notation, ComplEx can handle data with asymmetric predicates, and to keep scores in the real spaces it only uses the real part of embeddings product outcome. ComplEx preserves both linear time and linear space complexities as in TransE and DistMult, however, it surpasses their accuracies in the task of link prediction due to its ability to model a wider set of predicate types.

• *Convolution-based embeddings interactions*: Following the success of convolutional neural networks image processing tasks, models like R-GCN (Schlichtkrull et al. 2018) and ConvE (Dettmers et al. 2018) utilized convolutional networks to learn knowledge graph embeddings. The R-GCN model learns entity embeddings using a combination of convolutional filters of its neighbours, where each predicate represent a convolution filter and each neighbour entity represents an input for the corresponding predicate filter. This approach is combined with the DistMult model to perform link prediction. Meanwhile, the ConvE model concatenates subject and predicate embeddings vectors into an image (a matrix form), then it uses a 2D convolutional pipeline to transform this matrix into a vector and computes its interaction with the object entity embeddings to generate a corresponding score as follows:

$$\phi_{spo}^{\text{ConvE}} = f(vec(f([\overline{e_s}; \overline{w_p}] * \omega))W)e_o \tag{2.6}$$

where $\overline{e_s}$ and $\overline{w_p}$ denotes a 2D reshaping of e_s and w_p, ω is a convolution filter, f denotes a non-linear function, $vec(x)$ is a transformation function that reshape matrix x of size $m \times n$ into a vector of size $mn \times 1$.

2.4.2 Negative Sampling

KGE models generate negative instances during training using two different techniques: 1-versus-n and 1-vs-all negative samples. In the following we describe both these two different sampling techniques.

1-versus-n negative sampling

In this approach, a KGE model samples n negative instance for each positive instance in the training data using uniform random sampling. For example, given a true training triplet (s, p, o) a knowledge graph embedding model will generate n training instance by corrupting either the subject or the object entities as described in the following notation:

$$\mathcal{N}_{1-\text{vs}-\text{n}}[(s, p, o), n] = \bigcup^{n/2}(s', p, o) \cup \bigcup^{n/2}(s, p, o'),$$

were $s' \in E$ and $o' \in E$ denote the corruptions of the subject and object respectively.

1-versus-all negative sampling

In this approach the KGE model of generates $2|E|$ corruptions of each training instance where $|E|$ is the number of entities in the knowledge graph. This is achieved by corrupting the subject and object with all possible corruptions as follows:

$$\mathcal{N}_{1-\text{vs}-\text{all}}[(s,p,o)] = \bigcup_{s' \in E} (s',p,o) \cup \bigcup_{o' \in E} (s,p,o').$$

Some of the used corruptions in this negative sampling approach will be true specially for 1-to-many and many-to-many relations which may seem problematic. However, due to the traditionally large entities vocabularies in knowledge graphs, the effect of these instances in the computation of the training loss and computing gradients in negligible and this method is known to yield high predictive accuracy when applied in the training of KGE models.

Further details and discussion on negative sampling in knowledge graph embedding models and its influence on predictive accuracy and scalability of knowledge graph embedding models is included in Chapter 4.

2.4.3 Knowledge Graph Embedding Loss Functions

Knowledge graph embedding models can be categorized as ranking models, and in learning to rank, models typically use loss functions to learn the optimal scoring function parameters from training data. The learning process then involves minimising a loss function defined on the basis of the objects, their labels, and the scoring function. In learning to rank models, the objective is to score a set of labelled objects such that: for each two objects with different label values, the object with greater label value also has greater model score. Next, we present the several approaches proposed in learning to rank to learn optimal scoring functions.

Let $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ be a set of objects (triples) to be ranked. Let $l : X \rightarrow \mathbb{N}$ be a labelling function where $l(\mathbf{x})$ is the label of object $\mathbf{x}$ (*e.g.* , true/false or an arbitrary integer in case of multi-label problems). By default, we assume a binary labelling for triples, $l : X \rightarrow \{0,1\}$, where 0 and 1 represent false and true labels, respectively.

Let $\mathcal{F} = \{f_1, f_2, ..., f_n\}$ be the set of possible scoring functions. Given a KGE model, $f : X \rightarrow \mathbb{R}$ is its scoring function, where f aims to score triples in X such that positive triples are scored higher than negative ones, formally, $\forall \mathbf{x}_i, \mathbf{x}_j \in X \; l(\mathbf{x}_i) > l(\mathbf{x}_j) \implies f(\mathbf{x}_i) > f(\mathbf{x}_j)$. Finally, $\mathcal{R}(\mathbf{x}_i, f)$ denotes the rank position of element $\mathbf{x}_i$ according to scoring function f, or the position of score $f(\mathbf{x}_i)$ in a descending order of all scores $f(\mathbf{x})$ for all $\mathbf{x} \in X$.

Pointwise approach: The loss function in this approach is defined in terms of the difference between the elements' predicted score and its actual label value. The objective of this loss is to independently maximize the scores of true instances and minimize the scores of negative

instances. This is achieved by minimizing the difference between the scores of true instances and the true label, and minimizing the difference between the scores of negative instances and the false label. The formula is as follows:

$$\mathcal{L}_{\text{pointwise}}(f; X, l) = \sum_{i=1}^{n} \phi(f(\mathbf{x}_i) - l(\mathbf{x}_i)),$$

where ϕ is a transformation function, *e.g.* , square function $\phi(x) = x^2$ as in Bayes optimal subset ranking (Cossock & Zhang 2008) and RESCAL KGE model (Nickel et al. 2011).

Pairwise approach: The loss is defined as the summation of the differences between the predicted score of an element and the scores of all negative elements. The objective is then to minimize the difference between negative and positive instances such that the scores of negative instances are always less than the scores of positive instances with a specific margin. The formula is as follows:

$$\mathcal{L}_{\text{pairwise}}(f; X, l) = \sum_{i=1}^{n-1} \sum_{j=1, l(\mathbf{x}_j) < l(\mathbf{x}_i)}^{n} \phi(f(\mathbf{x}_i) - f(\mathbf{x}_j)),$$

where the function ϕ can be the hinge function as in the Translating Embeddings model (Bordes et al. 2013) or the exponential function as in RankBoost (Freund et al. 2003).

Listwise approach: The loss is defined as a comparison between the rank permutation probabilities of model scores and values of actual labels (Cao et al. 2007). Let $\phi(x)$ be an increasing and strictly positive function. We define the probability of an object being ranked on the top (*a.k.a.* top one probability), given the scores of all the objects as:

$$P_f(\mathbf{x}_i) = \frac{\phi(f(\mathbf{x}_i))}{\sum_{j=i}^{n} \phi(f(\mathbf{x}_j))},$$

where $f(\mathbf{x}_i)$ is the score of object i, $i = 1, 2, \ldots, n$. The listwise loss can then be defined as:

$$\mathcal{L}_{\text{listwise}}(f; X, l) = \sum_{i=1}^{n} \mathcal{L}_m(P_f(\mathbf{x}_i), P_l(\mathbf{x}_i)),$$

where $\mathcal{L}_m$ is a model-dependent loss. Possible examples include cross entropy in ListNet (Cao et al. 2007) or likelihood loss as in ListMLE (Xia et al. 2008).

Further details and discussion on the examples of loss functions of knowledge graph embedding models and their influence on predictive accuracy and scalability of knowledge graph

embedding models is included in Chapter 4.

2.5 Summary

In this chapter, we have discussed the basic concepts, terminologies and techniques that we use in this thesis. We have introduced knowledge graphs, their current examples and applications and the problem of knowledge graph incompleteness. We then provided a brief discussion on the objective of knowledge graph extension and its related tasks such as the knowledge graph completion and link prediction tasks, where we defined each task and discussed its evaluation protocol. Further details and discussions on these tasks follows in Chapters 3 and 4 respectively.

We have also discussed the different types of predictive models for extending knowledge graphs such as graph feature models and knowledge graph embedding models where we have discussed their training pipeline and their associated features and limitations. Further details and discussions on graph feature models is included in Chapter 3 while knowledge graph embedding models are discussed in Chapters 4 and 5 respectively. A set of biological use cases of knowledge graph embedding models are also included in Chapters 6, 7 and 8.

Technical Contributions Part II

3 Knowledge Graph Completion Using Distinct Subgraph Paths

3.1 Overview

Large scale knowledge graphs (*i.e.* graph-structured knowledge bases) have been used as convenient means for modelling information in many different domains, including general human knowledge (Lehmann & et. al. 2014), biomedical information (Dumontier et al. 2014) and language lexical information (Miller et al. 1990). Knowledge graphs are now used by different applications such as enhancing semantics of search engine results (Singhal, Qian 2012, 2013), biomedical discoveries (Muñoz et al. 2016), or powering question answering and decision support systems (Ferrucci et al. 2010). Despite the huge volume of information stored in knowledge graphs, they are still incomplete (Razniewski et al. 2016). For example, 69% to 99% of entities in most of the popular knowledge bases like Freebase (Bollacker et al. 2008) and YAGO3 (Mahdisoltani et al. 2015) lack at least one property possessed by all other entities in the same class (Razniewski et al. 2016). Incompleteness of knowledge bases can substantially affect the efficiency of systems relying on them, which has motivated research in knowledge base completion via automatic prediction of new, implicit facts.

This work addresses a family of knowledge base completion models known as graph feature models, which use graph patterns as features. One of the early models in this family is Path Ranking Algorithm (PRA) (Lao & Cohen 2010*b*), which uses paths connecting pairs of nodes as indicating features for predicting new direct links between nodes. For example, in Fig. 3.1, PRA can predict the fact that *Tedd* is playing for *TeamX* using the path ⟨ colleague, plays_for ⟩ along with the path ⟨ practise, practise^{-1}, plays_for ⟩ where *practise*$^{-1}$ is the inverse of relation *practise*. PRA extracts these connecting path features using random walks linking the subject and object nodes. Then, it uses each random walk probability as a value in a feature vector corresponding to the subject and object. This technique is able to provide expressive prediction for new facts. However, it suffers from low efficiency, and high computational cost of computing random walk probabilities. An extension of PRA uses backward random walks (Lao et al. 2015) to extract paths originating from object and reaching the subject node like the path ⟨ plays_for^{-1}, colleague^{-1} ⟩ in Fig. 3.1, which resulted in an efficiency

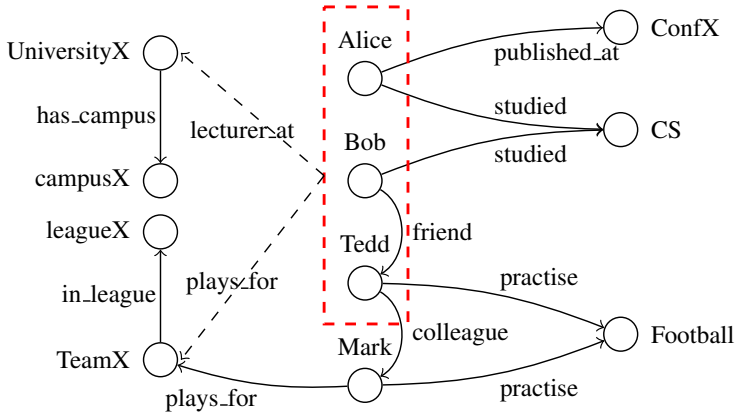

Figure 3.1 – A sample of a graph about people and their professions.

improvement over traditional PRA path features. Other extensions suggest using latent feature representations as a support, like latent syntactic cues (Gardner et al. 2013) which introduces a latent feature representation of combination of relations to infer new ones. Another approach suggests incorporating similarity between latent representation of relations as support features for knowledge base completion (Gardner et al. 2014), leading to significant efficiency improvements for models based on random walk inference. However, they suffer from the same computational problems as for PRA. Later, Neelakantan et al. introduce path bigram features (Neelakantan et al. 2015) for connecting paths. This work leads to significant improvement in the efficiency of PRA. Furthermore, Subgraph Feature Extraction (SFE) (Gardner & Mitchell 2015) – the state of the art model – drops random walk probabilities, and uses a binary representation of paths. SFE also proposes using ANYREL features, which is a set of subgraph paths built from connecting paths by replacing relation instances with a wild card. For example, given a natural path ⟨ a, b ⟩, the ANYREL version of this path is a set of two paths ⟨ ANYREL, b ⟩ and ⟨ a, ANYREL ⟩, which is a relaxation of the path representation to allow more intersections between different paths. This supports a richer representation of connecting paths, allowing for more intersections of similar connecting paths through the ANYREL wild card that replaces relation instances in connecting paths.

Despite the improvements achieved by SFE in path extraction and representation, all reviewed methods still suffer from two interrelated problems. Firstly, they represent facts using a limited feature set, *i.e.* connecting paths. This only captures interaction between subject and object entities, and neglects information describing entities themselves (like other subgraph paths to neighbour nodes that can capture entity attributes and properties). This can lead to suboptimal predictions. The second problem is that the current methods totally disregard entity pairs with no connecting paths in between as a consequence of using a limited feature set. This means that the methods cannot make certain predictions at all.

We propose a new model called Distinct Subgraph Paths (DSP) model. The model uses a

Table 3.1 – Properties of current graph feature models.

	PRA	**SFE**	**DSP**
Negatives	Failed RW.	PPR	PPR
F. types	CP	CP & ANYREL	DSP & ANYREL
F. weights	RW prob.	Binary	Binary
Model	LogReg	LogReg	LogReg
Scoring	$\sum_{f_i \in X} A_i$	$\sum_{f_i \in X} A_i$	$\frac{1}{1+\exp(-(X \cdot A+b))}$
Scope	Connected	Connected	All

new set of features that describe distinct properties of entities using disjoint sets of subgraph paths for both subject and object entities. For example, in Fig. 3.1, while investigating the fact ⟨*Alice*, *lecturer_at*, *UniversityX* ⟩, we propose using subgraph paths to express properties of subject and object entities *Alice* and *UniversityX*. *Alice* can be expressed using the path set: [⟨ SUB:published_at ⟩, ⟨ SUB:studied ⟩, ⟨ SUB:studied ⟩studied^{-1}], and *UniversityX* using: [⟨ OBJ:has_campus ⟩]. These disjoint path sets provide distinct description of both subject and object entities for a given fact. This makes our model capable of:

1. Employing a richer set of features that can describe properties of candidate fact entities.
2. Providing ranking scores for node pair candidates in the absence of connecting paths.
3. Providing better results than PRA and SFE (ANYREL) in terms of mean average precision (MAP), mean reciprocal rank (MRR) and Hits@5, 10, 20 with no extra computational cost.

3.2 Background

Many relational learning models were developed to predict new facts in knowledge bases. In recent years, latent feature models witnessed a rapid development providing a variety of models using methods such as tensor factorization (Nickel et al. 2011) or latent distance embeddings (Bordes et al. 2011). Although these models excel in the task of link prediction in knowledge graphs, their predictions are hard to interpret. They act as a black box relying on latent representation of features that are hard to trace back to original knowledge (Toutanova & Chen 2015).

In contrast, graph feature models provide more expressive predictions. They use graph-based features like subgraphs, connecting paths and neighbourhood information, which corresponds to intelligible parts of prior knowledge. This makes these techniques more suited to use cases where interpretability of the predictions matters (*e.g.* in life sciences).

Recent development of graph feature models encompasses Path Ranking Algorithms (PRA) (Lao & Cohen 2010*b*) and its variations that used backward random walks (Lao et al. 2015), latent

syntactic cues (Gardner et al. 2013), incorporating vector similarity in inference (Gardner et al. 2014) or bigram feature path (Neelakantan et al. 2015). The most recent improvement of the PRA-based techniques is SFE (Gardner & Mitchell 2015) that uses ANYREL path features. This set of models relies exclusively on connecting paths between nodes as features. They provide expressive and interpretable predictions, but they still lack in terms of efficiency and ability to predict scores for relationship between non-connected nodes (Gardner & Mitchell 2015).

In their work, Gardner & Mitchell (2015) investigated the use of non-connected subgraph paths called "One-Sided features." In their experiment, the authors only considered these features with higher expressivity on data with connected entity pairs and concluded that such features yield inferior results compared to PRA and ANYREL features. In our work we consider the effect of non-connected subgraph paths to better model relationships even in the absence of connecting paths. Our experiments have demonstrated the relevance of this contribution.

Despite the recent focus on latent feature models, it has been observed experimentally that neither latent feature models nor graph models are superior for learning over knowledge graphs; they are complementary (Nickel, Murphy, Tresp & Gabrilovich 2016*b*). The former models harness global graph patterns, while the latter capture local and quasi-local graph patterns (Toutanova & Chen 2015).

In this work, we focus on enhancing the predictive capabilities of graph feature models and utilising their expressiveness in the task of knowledge base completion, where DSP model extends the work accomplished by PRA and SFE. Table 3.1 shows a comparison between properties of DSP model compared to previous models like SFE and PRA, where *Negatives* represent negative generation technique, *F.Types* represent feature types, *F. weights* represent representation of feature weights, and *Scoring* represent model's scoring function, where A is learnt coefficients, and b is learnt intercept.

SFE and DSP model use PPR for generating negative instances while PRA uses failed random walks. Also, they use binary representation of feature weights in the feature matrix while PRA uses random walk probabilities. On the other hand, DSP model uses DSP & ANYREL features while PRA and SFE use connected path and ANYREL path features. Also, its prediction scope target all candidate triples while PRA and SFE target triples with connected subject and object entities. These are our main technical contributions over the state of the art, and our experiments have shown these contributions are reflected in tangible gains in performance, without sacrificing computational efficiency.

3.3 Distinct Subgraph Paths Model

In this section, we focus on the technical description of DSP model. We present how DSP model extracts feature paths from knowledge graphs, how the model is learned, and how DSP model predict a score for new absent facts.

Table 3.2 – Example of candidates' ranking for the relation *lecturer_at* as per knowledge graph from Fig. 3.1 and their relevance (Rel.) to the relation.

Candidates of relation: *lecturer_at*				
Rank	**Subject**		**Object**	
#	**Entity**	**Rel.**	**Entity**	**Rel.**
1	Alice	High	UniversityX	High
2	Bob	Med.	UniversityX	High
3	Tedd	Low	UniversityX	High
4	Alice	High	TeamX	Low
5	Bob	Med.	TeamX	Low
6	Tedd	Low	TeamX	Low

3.3.1 Motivating Example

The task of knowledge base completion is a ranking task by nature, since the aim is to find the most probable absent true facts in a knowledge base (Socher et al. 2013). For example, in Fig. 3.1, a knowledge base completion task would aim at ranking possible facts about people and their workplaces. In the absence of connecting paths between entities, such as between {*Alice*, *Bob*, *Tedd*} and *UniversityX*, approaches like SFE, PRA and its variants would not be able to provide a corresponding relation score. Indeed these methods rely on the assumption that since non-connected nodes have no connecting paths, they have no direct relationship. However, absence of connecting paths can be a result of knowledge incompleteness.

In our approach we consider using a set of features that we call distinct subgraph paths (DSP) as support features for ranking candidate absent facts. Distinct subgraph paths are the union of the two sets of subgraph paths originating from subject and object nodes of a triple in a knowledge graph, each prefixed with a distinct label corresponding to its origin ("*SUB:*" for subject or "*OBJ:*" for object). For example, when investigating the relation *lecturer_at* between persons group of entities and corresponding workplace entities, our model will be able to use the presence of path features like ⟨ SUB:published_at ⟩ and ⟨ SUB:studied ⟩ to predict that *Alice* has a high probability of being a subject for the relation. Similarly for other entities in our example, subgraph paths can be used to rank candidate facts as in Table 3.2. This provides a rank of most relevant entities to be connected with a specific relation. As per our example, *Alice* who has *studied*, and *published_at* a conference, is more likely to be lecturing than *Bob* who only has *studied* or *Tedd* who has none of these relative attributes. Also, the fact that each one of them can be lecturing at *UniversityX* is more probable than lecturing at *TeamX*, as attributes of *UniversityX* like *has_campus* is more relevant to objects of relation *lecturer_at* than those for *TeamX*. Therefore, this approach of using distinct sets of subgraph paths[1] for subject and object entities for a specific relation can support the construction of ranking scores for node pair candidates even in the absence of connecting paths as shown in Table 3.2.

[1] In the context of feature extraction for knowledge graph triples, we define a subgraph path as any path originating from candidate fact subject or object nodes.

Algorithm 1 EXTRACT SUBGRAPH PATHS

Input: v node, depth d, knowledge graph **KG**, root path R
Output: v^{sg} Subgraph paths

```
1: if d = 0 then
2:     return ⟨v⟩
3: else
4:     v^sg = [ ]
5:     for (r, u) ∈ Γ(v, KG) do
6:         R_u = R + ⟨r_k, u_k⟩
7:         v_u^sg = ExtractSubgraphPaths(u, d-1, KG, R_u)
8:         v^sg = v^sg ∪ v_u^sg ∪ R_u
9:     return v^sg
```

However, unlike connecting paths features used in PRA or SFE, subgraph paths do not capture the interactions (connecting paths) between entity pairs. Therefore, we use a combination of both DSP and ANYREL connecting paths features to support ranking node pair candidates whether they are connected or not in the knowledge graph.

DSP model training operates in two phases. First, it extracts path features for each node pair instance consisting of both connecting path (ANYREL) and subgraph path (DSP) features. Both of these features sets capture different properties of candidate facts. Subgraph paths of subject and object entities capture the relevance between these two entities and the considered relation, while connecting paths between subject and object entities capture their interactions. DSP model uses these two types of features to build a feature matrix with binary representation of path features for each relation type. In the second phase, DSP model trains a binary classifier for each feature matrix and uses this model for later predictions. Further description of the how it works follows in the next subsections.

3.3.2 Feature Extraction

Let **KG**be a knowledge graph, $\Gamma(e, \mathbf{KG})$ be the set of neighbour links of entity e in a knowledge graph **KG**, where a link is a combination of a neighbour relation r and neighbour node u reached by this relation, $(l_1 + l_2)$ be the concatenation of two lists l_1 and l_2, and p_u be a path p going through node u.

First, DSP model extracts subgraph paths of both subject and object nodes using Depth-First Search as in Algorithm 1, then it labels these paths with distinct prefixes corresponding to their origin node ("*SUB:*" for subject or "*OBJ:*" for object) using a labelling function $\gamma(p, label)$. Then, DSP model combines subgraph path originating from subject and object entities that share a common target node to build connecting paths. Let $\tau(p)$ be the target node of path p that if p starts from node v to node u, then $\tau(p) = u$, $P_{s \rightsquigarrow t}$ be a path from node s to node t, and p^{-1} be the inverse of path p. An inverse of a path is obtained by inversing the order of path relations and changing their direction, that is if $p = \langle r_1, r_2^{-1}, r_3 \rangle$, then $p^{-1} = \langle r_3^{-1}, r_2, r_1^{-1} \rangle$. DSP model combines subgraph paths originating from subject node with the inverse of

Algorithm 2 EXTRACT FEATURE PATHS

Input: (s,t) node pair, path length l, knowledge graph **KG**
Output: $P^{ANYREL}_{s\rightsquigarrow t}$, $P^{DSP}_{s\rightsquigarrow t}$ feature paths

```
1:  s^sg = ExtractSubgraphPaths(s, ⌈l/2⌉, KG, [])
2:  t^sg = ExtractSubgraphPaths(t, ⌈l/2⌉, KG, [])
3:  P^DSP_{s⇝t} = γ(s^sg, "sub") ∪ γ(t^sg, "obj")
4:  T_s = { τ(p) | p ∈ s^sg }
5:  T_t = { τ(p) | p ∈ t^sg }
6:  T_c = T_s ∩ T_t
7:  P^cp_{s⇝t} = [ ]
8:  P^ANYREL_{s⇝t} = [ ]
9:  for t ∈ T_c do
10:     for p_s ∈ s^sg ∧ τ(p_s) = t do
11:         for p_t ∈ t^sg ∧ τ(p_t) = t do
12:             P^cp_{s⇝t} = P^cp_{s⇝t} ∪ (p_s ⊕ p_t^{-1})
13: P^ANYREL_{s⇝t} = ⋃_{p∈P^cp_{s⇝t}} AnyrelPaths(p)
14: return P^ANYREL_{s⇝t} ∪ P^DSP_{s⇝t}
```

path originating from object node (providing they share a common target node) to build a connecting path from subject to object. For example, a subject subgraph path $p_{s\rightsquigarrow t} = \langle r_1, r_2^{-1}\rangle$ and object subgraph path $p_{o\rightsquigarrow t} = \langle r_5, r_3^{-1}\rangle$ are combined to generate a connecting path $p^{cp}_{s\rightsquigarrow o} = p_{s\rightsquigarrow t} \oplus p^{-1}_{o\rightsquigarrow t} = \langle r_1, r_2^{-1}, r_3, r_5^{-1}\rangle$. After, DSP model builds ANYREL paths corresponding to extracted connecting paths and label them with prefix label "*ANYREL:*". This procedure of extracting DSP and ANYREL path features is described in Algorithm 2.

For example, when DSP model extracts features for the fact (*Tedd, plays_for, TeamX*) from Fig. 3.1, it extracts subgraph paths around subject and object entities *Tedd* and *TeamX*. The union of these two sets of subgraph paths constitutes DSP feature paths. After, DSP model uses common target nodes to subject and object entities *Tedd* and *TeamX* such as *Mark* and *Football* to extract connecting paths. For example, DSP model combines paths to common target node *Football*: ⟨ practise ⟩ that originates from subject node and ⟨ plays_for, practise ⟩ that originates from object node, by appending the subgraph path from subject entity ⟨ practise ⟩ to the inverse of object entity path ⟨ practise^{-1}, plays_for^{-1} ⟩. This results in a connecting path ⟨ practise, practise^{-1}, plays_for^{-1} ⟩.

DSP model uses the same connecting paths extraction process as SFE, and extends it with distinct subgraph paths for subject and object nodes as discussed. Despite its high complexity, the feature extraction process is embarrassingly parallel and can therefore be distributed to minimise computational cost. It is important to note that feature extraction phase of DSP model requires no extra computational overhead compared to SFE since proposed DSP subgraph features are already extracted while generating connecting paths as shown in Algorithm 2.

3.3.3 Model Learning

For each candidate fact, the DSP model extracts distinct subgraph and ANYREL paths features to build a feature matrix based on the union of the two paths feature sets. These features represent the column names of the feature matrix, and for each fact, DSP model populates corresponding feature column with 1 when the feature is extracted for the fact and 0 otherwise.

DSP generates a separate feature matrix for each relation, where feature matrices are generated from a set of both positive and negative facts. Then, for each feature matrix DSP model trains a logistic regression model to learn a binary classification model for each relation. This model is then used to predict scores for candidate facts such that learned path feature weights differ for each relation type corresponding to its extracted path feature matrix.

Logistic regression is a binary classifier *i.e.* it can discriminate between *true* and *false* facts in case of knowledge base completion. It is used to predict scores of candidate facts corresponding to both classes. DSP model uses the difference between these scores corresponding to *true* and *false* facts to generate a single score for candidate facts in the following manner:

$$s(f)_{DSP} = s(f)_{lr}^{true} - s(f)_{lr}^{false}$$

where $s(f)_{DSP}$ is DSP model's score of candidate fact f, $s(f)_{lr}^{true}$ is logistic regression score of class "*true facts*" and $s(f)_{lr}^{false}$ is logistic regression score of class "*false facts*", that:

$$s(f)_{lr}^{true} = \frac{1}{1 + \exp(-(X \cdot A + b))}$$

for feature row X and learnt coefficients A, and intercept b and $s(f)_{lr}^{false} = 1 - s(f)_{lr}^{true}$, that DSP model scoring function can be defined as:

$$s(f)_{DSP} = 2 * s(f)_{lr}^{true} - 1$$

Since the output is an ordered rank, we can simplify the scoring function to be

$$s(f)_{DSP} = s(f)_{lr}^{true} = \frac{1}{1 + \exp(-(X \cdot A + b))}$$

Using the difference of both logistic regression scores associated to true fact and false fact classes the DSP model is able to transform the output of the classification into a rank. In case of using a different learning model than logistic regression, classes score difference can have different interpretation. We use logistic regression following previous state-of-the-art path feature models, however, we aim to investigate the performance of different learning models *e.g.* SVM classifier or decision trees in future work.

High ranked elements reflect high positive difference of class scores in favour of the true facts' class, and low ranked elements reflect a high negative difference of class scores in favour of

Table 3.3 – Example of DSP and SFE model interpretation of a prediction score of fact (*Tedd, plays_for, TeamX*).

#	**Path feature**	**DSP Weight**	**SFE Weight**
1	⟨ SUB:practise ⟩	0.32	N/A
2	⟨ SUB:friend^{-1} ⟩	-0.21	N/A
3	⟨ OBJ:in_league ⟩	0.45	N/A
4	⟨ OBJ:plays_for^{-1} ⟩	0.53	N/A
5	⟨ ANYREL:colleague, ANYREL ⟩	0.75	0.8
6	⟨ ANYREL:ANYREL, plays_for ⟩	1.45	1.2

the false facts' class.

3.3.4 Model Interpretability

Expressiveness of machine learning models is a key aspect in their evaluation, as understanding the behaviour of a model empowers both users and designers of the model, and it can help assessing the trust in it (Ribeiro et al. 2016). Graph feature models use graph components as features, and these components can be used as a meaningful explanation of their prediction. Usually, predictions of graph feature models are expressed by features they extract, that represent prior knowledge parts *e.g.* subgraph paths, connecting paths or neighbour nodes. While other approaches *e.g.* association rule mining (Galárraga et al. 2015), and relation path pattern mining (Mohamed et al. 2017) extract rules and patterns from knowledge graphs and use them as evidence for existence of candidate paths and triples.

DSP model expresses its predictions using the set of features it uses: distinct subgraph paths and ANYREL paths. It uses weights of the learned path features coefficients as an explanation. Each feature coefficient in DSP learned model represents how important is the corresponding path feature for the predicted score. I For example, predicting a score for candidate fact (*Tedd, plays_for, TeamX*) from Figure 3.1 can be expressed in a series of path features and associated weights as shown in Table 3.3. Among the features that contributed the most to the prediction is ⟨ ANYREL:ANYREL, plays_for ⟩ which can correspond in our example to the path: ⟨ colleague, plays_for ⟩.

Table 3.3 presents a set of possible path features that can be extracted by DSP model for candidate fact (*Tedd, plays_for, TeamX*) with their possible learnt weights for DSP and SFE. The table shows that DSP model can extract and learn coefficients for richer set of path features. For example, it can learn weights for first four path feature types, distinct subgraph paths, while SFE model only extracts and learns ANYREL path feature types.

3.4 Experiments

In this section, we present the benchmarking dataset, NELL, and we discuss its curation sources, its properties, and its method for generating negative instances. Then, we discuss our evaluation protocol and ranking metrics. We also discuss setup and implementation details of our experiments.

3.4.1 Benchmark dataset

NELL benchmark dataset To evaluate DSP model and compare it with prior art, we reuse the NELL benchmark dataset[2] proposed by (Gardner & Mitchell 2015) and used to compare SFE, PRA and its variants. NELL dataset was automatically created by scraping the Web then extracting general knowledge information from web pages (Mitchell et al. 2015). The dataset itself uses knowledge base completion models for assessment of new candidate facts that can be learned from present facts. The NELL benchmark dataset consists of three elements: graph triples, the knowledge graph including all entities and relations; training triples and testing triples, sets of positive and negative instances of 10 relations used for evaluation purpose. Statistics of NELL benchmark dataset used in experiments are detailed in Table 3.4.

Negative sample generation Generating negative example facts is an important issue for training the model. In the NELL benchmark dataset, negative facts are generated using constrained version of closed world assumption. Typically, knowledge graphs contain only true facts. However, under the open world assumption, all absent facts can not be assumed to be false as this absence can happen due to knowledge graph incompleteness. Using a constrained version of closed world assumption allows using facts that does not exist in the knowledge graph as negative examples while using heuristics to minimise the chances of those newly asserted negative facts to be true. Gardner & Mitchell (2015) applied the following strategy to generate negative facts as part of the NELL benchmark dataset: for each subject and object nodes in the set of positive facts, a score is computed for similar nodes of same class (NELL meta information) using personalised page rank (Scarselli et al. 2004). Then, negative examples are generated by creating absent facts from most similar nodes from the personalised page rank with a ratio of 1:10 positives to negatives. We use the NELL benchmark dataset which consists of both positive and negative fact instances with 1:10 positives to negatives ratio to be able to compare to previous works.

3.4.2 Evaluation

Although using the same NELL benchmark dataset, PRA and SFE models make use of a limited instance set corresponding to limitations of their systems. In the following, we discuss the

[2] Description for the NELL benchmark dataset can be found at http://rtw.ml.cmu.edu/emnlp2015_sfe/

Table 3.4 – Statistics of the NELL benchmark dataset used in experiments.

NELL benchmark dataset	
Element	**# Triples**
Graph relations	≈ 110K
Graph entities	≈ 1.2M
Graph fact triples	≈ 3.8M
Training triple instances	≈ 54K
Testing triple instances	≈ 13.5K

evaluation configuration and evaluation metrics with regard to approaches used by prior art.

Configuration Since PRA and SFE based models are only taking into account connecting paths as features, they only consider a subset of the evaluation dataset for which node pairs have a connecting paths between them. DSP model can handle both instance node pairs with and without connecting paths. Ergo, we run our experiments in two different configurations: set of instances with connecting paths (referred to as "connected nodes") and set of all instances (referred to as "all nodes"). We evaluate DSP model, PRA (Lao & Cohen 2010*b*), SFE with different features like plain connecting paths *i.e.* PRA feature paths, bigram feature paths introduced by Neelakantan et al. (2015), and combination of PRA and ANYREL feature paths (Gardner & Mitchell 2015) over these two different configurations. Further discussion of these approaches and their path feature types is presented by Gardner & Mitchell (2015).

Since PRA, its variants and SFE do not handle non-connecting paths, we assume 0 as a score for instance node pairs with no connecting paths. The scoring function of these models depends on the accumulation of weights corresponding to connecting path features. Therefore, 0 score of non-connecting paths which do not belong to their feature set will not impact the scoring function.

Metrics Similarly to prior-art, the evaluation metrics we use are the mean average precision (MAP) and mean reciprocal rank (MRR). We introduce as well the use of the Hits@k metric which is the number of correct elements predicted among the top-k elements. MAP is the mean of a set of average precision (AP) scores, and average precision is the average of Precision@k scores for positive elements in the rank (Liu 2011). While the Hits@k metric reports the count of positive instances retrieved at position k, the Precision@k metric reports this count normalised by the number of true instances found at position (k) such that Precision@k is defined as:

$$P@k(\pi, l) = \frac{\sum_{t \leq k} I_{\{l_{\pi^{-1}(t)}=1\}}}{k}$$

where π is a list, l is label function, $I_{\{.\}}$ is an indicator function of a element which equals to 1 when the element is relevant and 0 otherwise, and $\pi^{-1}(j)$ denotes the element ranked at position j of the list π. Let n be the total number of rank elements and m be the total number

of true elements, we can define Average Precision as:

$$AP(\pi, l) = \frac{\sum_{k=1}^{n} P@K(\pi, l) \cdot I_{\{l_{\pi^{-1}(t)}=1\}}}{m}$$

Mean average precision is the mean of a set of average precision scores[3]. Mean reciprocal rank is the harmonic mean of the rank position of the first relevant element defined as:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$$

where $rank_i$ refers to the rank position of the first relevant element for the i-th query.

3.4.3 Pre-processing

For comparison purposes, we apply the same pre-processing strategy as for SFE (Gardner & Mitchell 2015). NELL benchmark dataset includes semantic information about relations such as inverse relation property. For example, relation *concept:riverflowsthroughcity* is declared as an inverse relation of *concept:cityliesonriver*, and both exist in the dataset. Following the pre-processing applied in SFE work, we discard these inverses while extracting path features so to disable direct inference form inverse relations. For instance, if there is a candidate fact (*river:wye*, *concept:riverflowsthroughcity*, *city:hay*) in the knowledge graph, we discard the corresponding inverse relation fact (*city:hay*, *concept:cityliesonriver*, *river:wye*).

Nonetheless, to allow for bidirectional exploration of the knowledge graph, we append inverses of all facts in the knowledge graph used. That is if (e_1, r_1, e_2) is a fact in the dataset, we also append (e_2, r_1^{-1}, e_1). While predicting a given fact using the model, we disregard candidate facts' inverse.

3.4.4 Implementation

In our experiments we use Python3 as a language. Version 0.17.1 of the scikit-learn python library is used for the implementation of logistic regression (Pedregosa et al. 2011). We use logistic regression with default configuration of L1 regularization, where inverse of regularization strength $C = 1.0$. We choose adjacency matrices as a data structure to represent a knowledge graph. During feature extraction, we extract subgraph paths of depth 2 where the depth is the number of relations in the path. We only use 50 neighbour instances per relation in order to avoid dense neighbourhood of nodes and keep a sample of all neighbour nodes.

[3]During our experiments, we have found out that the published code of PRA and SFE uses an inaccurate implementation of the AP metrics. After confirming this with the authors in a private communication, we have reimplemented the metric using the presented formula before computing the values discussed in this chapter. Note that the reimplemented metric does not introduce any dramatic changes when comparing the existing techniques among themselves or with our results. We only wanted to make sure the results we present are as accurate as possible.

Table 3.5 – Evaluation of DSP model over set of connected/all node pair instances.

Connected Nodes		
Model (features)	**MAP**	**MRR**
PRA (PRA)	0.447	0.792
SFE (PRA)	0.557	0.806
SFE (Bigrams)	0.638	**1.000**
SFE (PRA+ANYREL)	0.675	0.933
DSP(ANYREL+DSP)*	**0.690**	0.950
All Nodes		
Model (features)	**MAP**	**MRR**
PRA (PRA)	0.569	0.783
SFE (PRA)	0.540	0.806
SFE (Bigrams)	0.654	**1.000**
SFE (PRA+ANYREL)	0.655	0.933
DSP(ANYREL+DSP)*	**0.698**	0.950

Table 3.6 – Average Hits@k of DSP and other models.

Model	**MRR**	**Hits@k**		
		@5	**@10**	**@20**
SFE (Bigrams)	**1.000**	4.5	8.5	16.9
SFE (PRA+AR)	0.933	**4.6**	8.9	17.0
DSP (DSP+AR)*	0.950	**4.6**	**9.0**	**17.5**

This effectively mean that when a node has more than 50 instances of one relation, only 50 are considered. We do not sample the set of neighbour all together, as this may result in discarding neighbour relations with fewer instances in dense nodes. We therefore sample neighbour nodes per relation instance. All experiments run over a machine with 40 Gb of RAM and 10 CPU processing cores of 2.2 GHz.

3.5 Results and Discussion

The outcome of our experiments in both connected nodes and all nodes configurations is presented in Table 3.5. DSP model achieves a mean average precision of 0.692 and 0.698 for connected nodes configuration and all nodes respectively, outperforming SFE with MAP of 0.675 and 0.655, and PRA with 0.557 and 0.54. Considering also non-connected paths (all nodes configuration), the mean average precision of DSP model shows a slight improvement of 1%, while other approaches like SFE with different features shows a decrease of mean average precision. On the contrary, PRA shows an improvement of 12% in all node configuration, as in connected nodes configuration PRA has high percentage of discarded positive candidate facts due to absence of connecting path as it uses random walks, where scoring non connected instances with 0 enables PRA to gain this improvement.

Table 3.7 – Evaluation of DSP and SFE over NELL 10 relations using all nodes pairs, with percentage of non-connected instances for each relation.

Relation	NCI	DSP (DSP + ANYREL)					SFE (PRA + ANYREL)				
		AP	RR	H@5	H@10	H@20	AP	RR	H@5	H@10	H@20
concept:riverflowsthroughcity	1%	**0.503**	**1.00**	5	9	19	0.480	**1.00**	5	9	19
concept:sportsteampositionforsport	13%	**0.768**	**1.00**	5	10	14	0.558	**1.00**	5	9	9
concept:citylocatedincountry	1%	**0.548**	**1.00**	3	8	18	0.495	0.333	3	8	17
concept:athleteplaysforteam	1%	**0.776**	**1.00**	5	10	20	**0.776**	**1.00**	5	10	20
concept:writerwrotebook	10%	**0.828**	**1.00**	5	10	20	0.783	**1.00**	5	10	20
concept:actorstarredinmovie	9%	**0.869**	**1.00**	5	10	20	0.786	**1.00**	5	10	20
concept:journalistwritesforpublication	8%	**0.879**	**1.00**	5	10	20	0.838	**1.00**	5	10	20
concept:stadiumlocatedincity	1%	**0.628**	**1.00**	5	10	19	0.624	**1.00**	5	10	20
concept:statehaslake	0%	0.237	0.50	3	3	5	**0.278**	**1.00**	3	3	5
concept:teamplaysinleague	1%	**0.942**	**1.00**	5	10	20	0.936	**1.00**	5	10	20
Average	-	**0.698**	**0.950**	4.60	9.00	17.50	0.655	0.933	4.60	8.90	17.00

In terms of mean reciprocal rank (MRR), SFE with bigrams features provides best results with a score of 1.0. While MRR provides a useful information when a user only wishes to see one relevant element, it may be more suited in the context of knowledge graph completion to look at the number of top-k relevant elements. We present in Table 3.6 the measure of Hits at k for $k = 5, 10, 20$. For that measure, DSP model outperforms all other models demonstrating a better ability to rank higher relevant elements within top-$5, 10, 20$.

Table 3.7 details models' performance in the "all nodes" configuration per relation. Column "NCI" represents the percentage of non-connecting pair nodes instances for a given relation[4]. The results show that models like SFE or PRA are affected negatively by the absence of non-connected node pair instances. On the contrary and following our intuition, DSP model improvement is greater for relations with high percentage of non-connected instances. DSP model's highest improvement $\Delta_{MAP} DSPox 0.21$ is observed for relation *concept:sportsteampositionforsport* which has the highest percentage of non-connected instances of 13%. The lowest DSP model's relative performance $\Delta_{MAP} DSPox - 0.04$ is observed for relation *concept:statehaslake* with the lowest percentage of non-connected instances of 0%.

In our experiments, we have found that DSP model is able to provide a high rank[5] for true candidate facts even in the absence of connecting paths, hence not in the result set of any previous graph feature models. For example, considering the relation *concept:citylocatedincountry*, DSP model is able to predict that *city:abu_dhabi* is located within *country:the_united_arab_emirates* (ranked at top 2.87%) while there are no connecting paths between them. Similarly, DSP model was able to provide a high rank (top 2.19%) for the fact that *river:wye* and *city:hay* are connected with *concept:riverflowsthroughcity* relation, despite they have no connecting paths between them.

[4]Depending on the sampling method used (random walk, DFS), pair nodes can be considered connected or non-connected.

[5]Positive candidate facts with high rank are in the top 10% elements of the rank, where positive to negative ratio is 1:10.

4 Training Knowledge Graph Embedding Models

4.1 Overview

The recent advent of knowledge graph embedding (KGE) models has allowed for scalable and efficient manipulation of large knowledge graphs (KGs) such as RDF Graphs, improving the results of a wide range of tasks such as link prediction (Bordes et al., Wang et al., Nie & Sun 2013, 2014, 2019), entity resolution (Nickel et al., Bordes et al. 2011, 2014) and entity classification (Nickel et al. 2012). KGE models operate by learning embeddings in a low-dimensional continuous space from the relational information contained in the KG while preserving its inherent structure. Specifically, their objective is to rank knowledge facts—relational triples (s, p, o) connecting subject and object entities s and o by a relation type p—based on their relevance. Various interactions between their entity and relation embeddings are used for computing the knowledge fact ranking. These interactions are typically reflected in a model-specific scoring function.

For instance, TransE (Bordes et al. 2013) uses a scoring function defined as the distance between the o embedding and the translation of the embedding associated to s by the relation type p embedding. DistMult (Yang et al. 2015*a*), ComplEx (Trouillon et al. 2016) and HolE (Nickel, Rosasco & Poggio 2016) use multiplicative composition of the entity embeddings and the relation type embeddings. This leads to a better reflection of the relational semantics and leads to state-of-the-art performance results (refer to (Wang et al. 2017) for a review). Although there is a growing body of literature proposing different KG models (mostly focusing on the design of new scoring functions), other parts of the knowledge graph embedding learning process, *e.g.* loss functions, negative sampling strategies, etc, have not received much attention to date (Mohamed, Novácek, Vandenbussche & Muñoz 2019).

This has already been shown to influence the behaviour of the KGE models. For instance, (Hayashi & Shimbo 2017) observed that despite the different motivations behind HolE and CompleEx models, they have equivalent scoring functions. Yet their performance still differs. Trouillon et. al. (Trouillon & Nickel 2017) have concluded that this difference is caused by

the fact that HolE uses a max-margin loss while ComplEx a log-likelihood loss. This shows that cost functions are important for thorough understanding, and even improvement of the performance of different KGE models.

Other than loss function selection, the studies of Dettmers et. al. (Dettmers et al. 2018) and Lacroix et. al. (Lacroix et al. 2018) have shown that using 1-vs-all negative sampling strategy can significantly enhance the accuracy of KGE models. Furthermore, Kadlec et. al. (Kadlec et al. 2017) have also shown that the accuracy of KGE models is sensitive to the training parameters where minor changes to the parameters can significantly change the models' resulting accuracy. Despite the importance of all the previously mentioned parts of the KGE learning process, a comprehensive study is still missing. This study is the a step towards improving our understanding of the influence of the different parts of the training pipeline on the behaviour of KGE models. Our analysis specifically focuses on investigating the effects of training parts on both the scalability and accuracy of KGE models. We first investigate KGE loss functions and their different approaches, and we assess the effects of the loss function choice on different KGE models. We then study KGE negative sampling strategies and effects on both accuracy and scalability. We finally discuss the effects of training parameters such as the embedding size, batch size, etc, on the scalability and accuracy of different KGE models.

Despite the growing number of KGE models and their different new approaches, we limit our study to a basic set of models: the TransE (Bordes et al. 2013), DistMult (Yang et al. 2015*a*), TriModel (Mohamed & Novácek 2019) and Complex (Trouillon et al. 2016) models which represent the most popular and publicly available methods. We use these methods as simple examples to examine and showcase the different parts of the KGE learning process.

The summary of our contributions is as follows:

1. We provide a comprehensive analysis of training loss functions as used in several representative state-of-the-art KGE models. We also preform an empirical evaluation of different KGE models with different loss functions and we show the effect of these losses on the KGE models predictive accuracy.

2. We study negative sampling strategies and we examine their effects on the accuracy and scalability of KGE models.

3. We study the effects of changes in the different hyperparameters and their effects on the accuracy and scalability of KGE models during the training process.

4.2 Background

In this section, we discuss knowledge graph embedding models and training pipeline.

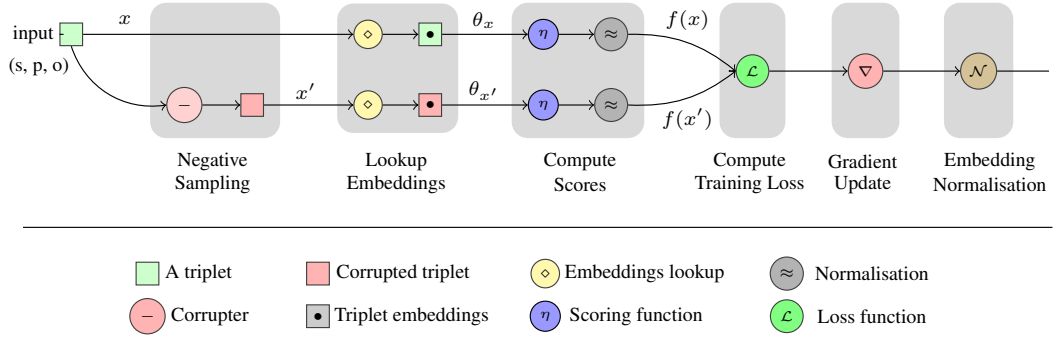

Figure 4.1 – An illustration of the process of training a knowledge graph embedding model over an example (s, p, o) triplet.

4.2.1 Knowledge Graph Embedding Process

Knowledge graph embedding models learn low rank vector representation *i.e.* embeddings for graph entities and relations. In the link prediction task, they learn embeddings in order to rank knowledge graph facts according to their factuality. The process of learning these embeddings consists of different phases as shown in Fig. 4.1. First, they initialise the embeddings of both relations and entities using random noise. These embeddings are then used to score a set of true and false facts, where the scores of facts are generated by computing the interaction between their subject, predicate and object embeddings using a model dependent scoring function. Finally, embeddings are updated through a gradient decent routine which minimises a training loss that usually represents a min-max loss over the scored facts. The objective is then to maximise the scores of true facts and minimise the scores of other facts.

Negative Sampling

In learning to rank approaches, models use a ranking loss *e.g.* pointwise or pairwise loss to rank a set of true and negative instances (Chen et al. 2009), where negative instances are generated by corrupting true training facts with a ratio of negative to positive instances (Bordes et al. 2013). This corruption happens by changing either the subject or object of the true triple instance. In this configuration, the ratio of negative to positive instances is traditionally learnt using a grid search, where models compromise between the accuracy achieved by increasing the ratio and the runtime required for training.

On the other hand, multi-class based models train to rank positive triples against their all possible corruptions as a multi-class problem where the range of classes is the set of all entities. For example, training on a triple (s, p, o) is achieved by learning the right classes "s" and "o" for the pairs (?, p, o) and (s, p, ?) respectively, where the set of possible class is E of size N_e. Despite the enhancements of predictions accuracy achieved by such approaches (Dettmers et al., Lacroix et al. 2018, 2018), their negative sampling procedure is exhaustive and require high space complexity due to the usage of the whole entity vocabulary per each triple.

Embedding Interactions

After KGE models produce negative samples from input triplets, they then generate scores for both the true and negative (corrupted) triplets. These scores are generated using embedding interaction function *i.e.* scoring functions. First, the model looks up the embeddings of the triplets subject, predicate and object. The model then process uses an embedding interactions function to learn a score for each triplet using its embeddings.

The embedding interaction functions are model-dependent and they operate using different approaches such as embedding translation (Bordes et al. 2013), linear products (Yang et al. 2015*a*) and convolutional filters (Dettmers et al. 2018). For example, the TransE model uses a translation based scoring function which encode embedding interactions as a translation from the subject embedding vector to the object embedding vector through the predicate vector (Bordes et al. 2013). Such an approach allowed highly scalable knowledge graph embedding with linear time and space complexity. However, it suffered from limited ability to encode 1-to-many predicates in knowledge graph due to dependence on direct additive translations (Yang et al. 2015*a*). On the other hand, the DistMult model used a linear product based scoring functions which allowed better encoding of 1-to-many predicates while preserving the linear time and space complexity. However, the DistMult model's scoring function suffered limited ability to preserve the predicate direction due to dependence on a symmetric operation (Yang et al. 2015*a*).

Further approaches such as the ComplEx (Trouillon et al. 2016), ConvE (Dettmers et al. 2018), TriModel (Mohamed & Novácek 2019), etc, proposed new scoring mechanisms which allowed encode both 1-to-many relation and preserve the predicate directionality within linear time and space complexity. Since these scoring functions are well covered in previous studies, we will not discuss the details of their mechanisms-of-action in our study. Further information and technical details about the knowledge graph embedding scoring functions can be found in the studies of Nickel et. al. (Nickel, Rosasco & Poggio 2016) and Wang et. al. (Wang et al. 2017).

4.2.2 Experimental Evaluation

In this part, we describe the setup of the experiments which we conducted ins this chapter on the TransE (Bordes et al. 2013), DistMult (Yang et al. 2015*a*) and ComplEx (Trouillon et al. 2016) KGE models. We present the benchmarking datasets, experiments setup, and implementation details.

Benchmarking Datasets In our experiments we use six knowledge graph benchmarking datasets:

- NELL239: a subsets of the NELL dataset (Gardner & Mitchell 2015) which contains general knowledge about people, places, teams, universities, etc (Mohamed & Novácek

Dataset	Entity count	Relation count	Train	Valid	Test
NELL239	48k	239	74k	3k	3k
WN18RR	41k	11	87k	3k	3k
FB15k-237	15k	237	272k	18k	20k
YAGO10	123k	37	1M	5k	5k
PSE	32K	967	3.7M	459K	459K

Table 4.1 – Statistics of entities, relations, and triples count per split of the benchmarking datasets which we use in this chapter.

2019). We use this dataset as it is small in size which enables fast prototyping and analysis of models. It also contains rich set of relations and a large number of entities which allow for assessing models' performance on highly diverse data.

- WN18: a subset of the Wordnet dataset (Miller et al. 1990) which contains lexical information of the English language (Bordes et al., Dettmers et al. 2013, 2018).

- FB15k-237: a subset of the Freebase dataset (Bollacker et al. 2008) that contains information about general human knowledge (Toutanova et al. 2015).

- YAGO10: a subset of the YAGO3 dataset (Mahdisoltani et al. 2015) that contains information mostly about people and their citizenship, gender, and professions knowledge (Bouchard et al. 2015).

- PSE: polypharmacy side-effects dataset (Zitnik et al. 2018) contains facts about drug combinations and their related side-effects. The dataset was introduced by Zitnik et. al. (Zitnik et al. 2018) to study modelling polypharmacy side-effects using knowledge graph embedding models. Since the dataset is significantly larger than the available standard benchmark we use it to study the effects of hyperparameters and accuracy of the knowledge graph embedding models.

Table 4.1 contains statistics about our experiments' benchmarking datasets [1]. Statistics about the two datasets used are presented in Table 4.1. Note that datasets derived from Freebase would have also been an option, but in this chapter, we opted to leave these out due to their highly demanding nature in terms of computational time and space required.

Evaluation Protocol The three KGE models are evaluated using a unified protocol that assesses their performance in the task of link prediction. Let X be the set of facts, *i.e.* triples, Θ_E be the embeddings of entities E, and Θ_R be the embeddings of relations R. The KGE evaluation protocol works in three steps:

[1] All the benchmarking datasets can be downloaded using the following url: https://figshare.com/s/8c2f1e1f98aff44b5b71

(1) *Corruption*: For each $\mathbf{x} = (s, p, o) \in X$, $\mathbf{x}$ is corrupted $2|E| - 1$ times by replacing its subject and object entities with all the other entities in E. The corrupted triples can be defined as:

$$\mathbf{x}_{\text{corr}} = \bigcup_{s' \in E} (s', p, o) \cup \bigcup_{o' \in E} (s, p, o')$$

where $s' \neq s$ and $o' \neq o$. These corruptions effectively provide negative examples for the supervised training and testing process due to the Local Closed World Assumption (Nickel, Murphy, Tresp & Gabrilovich 2016*b*).

(2) *Scoring*: Both original triples and corrupted instances are evaluated using a model-dependent scoring function. This process involves looking up embeddings of entities and relations, and computing scores depending on these embeddings.

(3) *Evaluation*: Each triple and its corresponding corruption triples are evaluated using the RR ranking metric as a separate query, where the original triples represent true objects and their corruptions false ones. It is possible that corruptions of triples may contain positive instances that exist among training or validation triples. In our experiments, we alleviate this problem by filtering out positive instances in the triple corruptions. Therefore, MRR and Hits@k are computed using the knowledge graph original triples and non-positive corruptions only (Bordes et al. 2013).

4.3 Loss Functions in KGE Models

Generally, KGE models are cast as learning to rank problems. They employ multiple training loss functions that comply with the ranking loss approaches. In the state-of-the-art KGE models, loss functions were designed according to various pointwise and pairwise approaches that we review next.

4.3.1 KGE pointwise losses

In the following, we discuss current pointwise loss functions for KGE models including SE, hinge, and logistic losses.

Pointwise square error loss (SE) It is a pointwise ranking loss function used in RESCAL (Nickel et al. 2011). It models training losses with the objective of minimising the squared difference between model predicted scores for triples and their true labels:

$$\mathcal{L}_{\text{SE}_{Pt}} = \frac{1}{2} \sum_{i=1}^{n} (f(\mathbf{x}_i) - l(\mathbf{x}_i))^2. \tag{4.1}$$

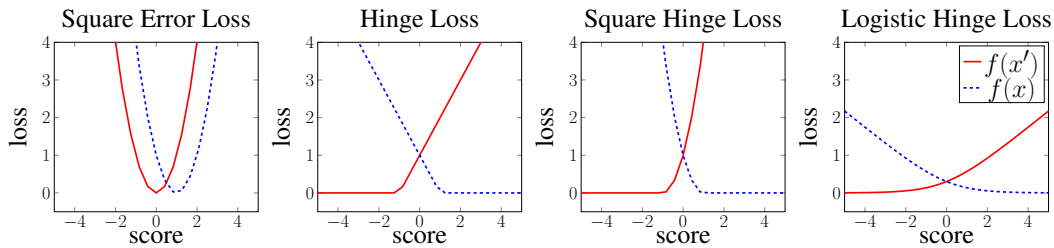

Figure 4.2 – Plot of the loss growth of different types of pointwise knowledge graph embedding loss functions.

The optimal score for true and false facts is 1 and 0, respectively. A nice to have characteristic of SE loss is that it does not require configurable training parameters, shrinking the search space of hyper parameters compared to other losses (*e.g.* , the margin parameter of the hinge loss).

Pointwise hinge loss: Hinge loss can be interpreted as a pointwise loss, where the objective is to generally minimise the scores of negative facts and maximise the scores of positive facts to a specific configurable value. This approach is used in HolE (Nickel, Rosasco & Poggio 2016), and it is defined as:

$$\mathcal{L}_{\text{hinge}_{Pt}} = \sum_{\mathbf{x} \in X} [\lambda - l(\mathbf{x}) \cdot f(\mathbf{x})]_+, \tag{4.2}$$

where $l(\mathbf{x}) = 1$ if $\mathbf{x}$ is true and -1 otherwise, and $[x]_+$ denotes $max(x, 0)$. This effectively generates two different loss slopes for positive and negative scores as shown in Fig. 4.2. Thus, the objective resembles a pointwise loss that minimises negative scores to reach $-\lambda$, and maximises positives scores to reach λ.

Pointwise logistic loss The ComplEx (Trouillon et al. 2016) model uses a logistic loss, which is a smoother version of pointwise hinge loss without the configurable margin parameter. Logistic loss uses a logistic function to minimise the negative triples score and maximise the positive triples score. This is similar to hinge loss, but uses a smoother linear loss slope defined as:

$$\mathcal{L}_{\text{logistic}_{Pt}} = \sum_{\mathbf{x} \in X} \log(1 + \exp(-l(\mathbf{x}) \cdot f(\mathbf{x}))), \tag{4.3}$$

where $l(\mathbf{x})$ is the true label of fact $\mathbf{x}$ where it is equal to 1 for positive facts and is equal to -1 otherwise.

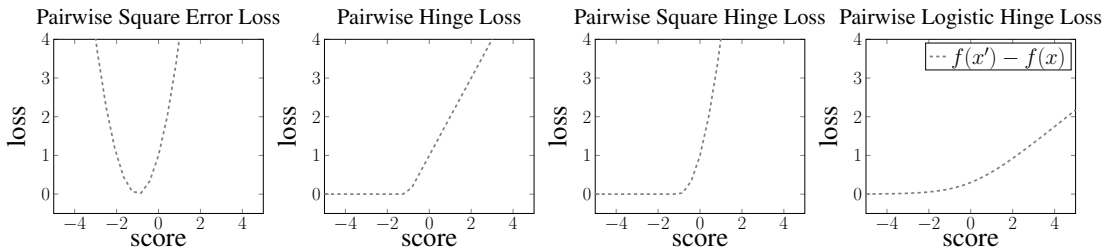

Figure 4.3 – Plot of the loss growth of different types of pairwise knowledge graph embedding loss functions.

4.3.2 KGE Pairwise Losses

Here, we discuss established pairwise loss functions in KGE model which are summarised in Fig. 4.3.

Pairwise hinge loss Hinge loss is a linear learning to rank loss that can be implemented in both a pointwise or pairwise loss settings. In both the TransE (Bordes et al. 2013) and DistMult (Yang et al. 2015*a*) models the hinge loss is used in its pairwise form, where it is defined as follows:

$$\mathcal{L}_{\text{hinge}_{Pr}} = \sum_{\mathbf{x} \in X^+} \sum_{\mathbf{x}' \in X^-} [\lambda + f(\mathbf{x}') - f(\mathbf{x})]_+, \tag{4.4}$$

where X^+ is the set of true facts, X^- is the set of false facts, and λ is a configurable margin.

In this case, the objective is to minimise the marginal difference (difference of scores with the added margin) between the scores of negative and positive instances. This approach optimises towards having embeddings that satisfy $\forall_{\mathbf{x} \in X^+} \forall_{\mathbf{x}' \in X^-} f(\mathbf{x}) > f(\mathbf{x}')$ as in Fig. 4.3.

Pairwise logistic loss Logistic loss can also be interpreted as pairwise margin based loss following the same approach as in hinge loss. The loss is then defined as:

$$\mathcal{L}_{\text{logistic}_{Pr}} = \sum_{\mathbf{x} \in X^+} \sum_{\mathbf{x}' \in X^-} \log(1 + \exp(f(\mathbf{x}') - f(\mathbf{x}))), \tag{4.5}$$

where the objective is to minimise marginal difference between negative and positive scores with a smoother linear slope than hinge loss as shown in Fig. 4.3.

4.3.3 KGE multi-class losses

In the following, we discuss KGE loss functions that are used to cast the KGE process into a multi-class classification problem.

Binary cross entropy loss (BCE) The ConvE model's (Dettmers et al. 2018) study proposed a new binary cross entropy multi-class loss to model the training error of KGE models in link prediction. In this setting, the whole vocabulary of entities is used to train each positive fact such that for a triple (s, p, o), all facts (s, p, o') with $o' \in E$ and $o' \neq o$ are considered false. Despite the extra computational cost of this approach, it allowed ConvE to generalise over a larger sample of negative assistances, therefore surpassing other approaches in accuracy (Dettmers et al. 2018).

Negative-log softmax loss (NLS) In a recent work, Lacroix et. al. (Lacroix et al. 2018) introduced a softmax regression loss to model training error of the ComplEx model as a multi-class problem. In this approach, the objective for each (s, p, o) triple is to minimise the following loss:

$$\begin{aligned}
&\mathcal{L}^{\text{NLS}}_{spo} = \mathcal{L}^{o'}_{spo} + \mathcal{L}^{s'}_{spo}, \\
&\mathcal{L}^{o'}_{spo} = -\phi_{spo} + \log(\textstyle\sum_{o'} \exp(\phi_{spo'}) \\
&\mathcal{L}^{s'}_{spo} = -\phi_{spo} + \log(\textstyle\sum_{s'} \exp(\phi_{s'po})
\end{aligned} \tag{4.6}$$

where ϕ_{spo} is the model score for the triple (s,p,o), $s' \in E$, $s' \neq s$, $o' \in E$ and $o' \neq o$. This resembles a log-loss of the softmax value of the positive triple compared to all possible object and subject corruptions where the objective is to maximise positive facts scores and minimise all other scores. This approach achieved significant improvement to the prediction accuracy of ComplEx model over all benchmark datasets when used with the 3-nuclear norm regularisation of embeddings (Lacroix et al. 2018).

4.3.4 Effects of training objectives on accuracy

We performed an experimental evaluation for the effect of loss function on the accuracy of KGE models in the link prediction task in terms of MRR and Hits at 10. For simplicity, We have only experimented with three KGE models: the TransE, DistMult and Complex. These models are used as examples where we assess their performance on different benchmarks where they are coupled with different loss functions configurations.

Table 4.2 shows the outcome of our experiments where it compares the accuracy of the examined models on both ranking and multi-class loss approaches in terms of MRR and Hits@10. In

	Model	Loss			NELL239		WN18RR		Fb15k-237	
					MRR	H10	MRR	H10	MRR	H10
Ranking Loss	TransE	Pr	* Hinge		**0.28**	**0.43**	0.20	0.47	**0.27**	**0.43**
			Logistic		0.27	**0.43**	**0.21**	**0.48**	0.26	**0.43**
		Pt	Hinge		0.19	0.32	0.12	0.34	0.12	0.25
			Logistic		0.17	0.31	0.11	0.31	0.01	0.23
			SE		0.01	0.02	0.00	0.00	0.01	0.01
	DistMult	Pr	Hinge		0.20	0.32	0.40	0.45	0.10	0.16
			Logistic		0.26	0.40	0.39	0.45	0.19	0.36
		Pt	* Hinge		0.25	0.41	**0.43**	0.49	0.21	0.39
			Logistic		0.28	0.43	**0.43**	**0.50**	0.20	0.39
			SE		**0.31**	**0.48**	**0.43**	**0.50**	**0.22**	**0.40**
	ComplEx	Pr	Hinge		0.24	0.38	0.39	0.45	0.20	0.35
			Logistic		0.27	0.43	0.41	0.47	0.19	0.35
		Pt	Hinge		0.21	0.36	0.41	0.47	0.20	0.39
			* Logistic		0.14	0.24	0.36	0.39	0.13	0.28
			SE		**0.35**	**0.52**	**0.47**	**0.53**	**0.22**	**0.41**
Multi-class losses	CP (Hitchcock 1927)	MC	BCE		-	-	-	-	-	-
			NLS		-	-	0.08	0.12	0.22	0.42
	DistMult	MC	BCE		-	-	0.43	0.49	0.24	0.42
			NLS		0.39	0.55	0.43	0.50	0.34	**0.53**
	ComplEx	MC	BCE		-	-	**0.44**	0.51	0.25	0.43
			NLS		**0.40**	**0.58**	**0.44**	**0.52**	**0.35**	**0.53**

Table 4.2 – Link prediction results for KGE models with different loss functions on standard benchmarking datasets. The abbreviations MC, Pr, Pt stand for multi-class, pairwise and pointwise respectively. The * mark is assigned to the model's default loss function. In the ranking losses, best results are computed per model where bold results represent model's best result and underlined results represent the best result in each respective loss approach.

the ranking loss configuration, the results show that the default loss functions of the examined models does not always yield the best results. On the contrary, The DistMult and ComplEx models which by default use the pointwise hinge and logistic losses respectively obtain their best result using the pointwise square error loss with all the examined benchmarks. This shows that changing the default loss function of these models can help enhance their predictive accuracy. The results of the TransE model also show that its default loss (pairwise hinge loss) achieves best result on 4 out of 6 examined evaluation metrics. On the other hand, Its pairwise logistic loss configuration achieves the best result in 3 out of 6 of the examined evaluation metrics. Given that the pairwise logistic loss is non-parameters compared the margin-based hinge loss, the pairwise logistic loss can be a preferred configuration to the TransE model as it can significantly reduce the grid search time.

The results also show that the multi-class loss versions of the CP (Hitchcock 1927), DistMult, ComplEx models have significantly better results than their ranking based losses. For example, on the NELL239 data set, the best performing ranking-based approach, the complex model with the pointwise square error loss, achieves 0.35 and 0.52 scores in terms of MRR and Hits@10 respectively compared to it NLS loss version which achieves 0.40 and 0.58 scores respectively. The results also show that the best multi-class loss results are obtained using the negative log softmax loss. For example, both the multi-class based versions of the DistMult and ComplEx achieve their best results using the negative softmax loss.

4.3.5 Effects of training objectives on scalability

We have shown that different training objectives yield significantly different results for the same KGE models. Our experimental results also suggested that the multi-class loss functions achieve the best results in terms of MRR and Hits@10 on all the investigated dataset. However, this approach uses a 1-vs-all negative sampling strategy which is time-consuming due to its higher time and space complexity compared to usual 1-vs-n sampling. In the following, we compare the ranking losses and multi-class losses in terms of the runtime required for training a KGE on different dataset size to study the scalability of both approaches.

We execute an experiment where we use the YAGO10 benchmark where we train KGE models on different percentages of the dataset and study the relation between the growth in the dataset size and the required training runtime. We the compare the training runtime of different KGE models with the negative softmax loss (NLS) and the pointwise square error loss as representatives of their respective loss class.

Fig. 4.4 shows the outcomes of our experiments where it presents a series of plots which describe the relation between the growth of the dataset size and the growth of training runtime of both ranking and multi-class loss approaches. The results show that the multi-class losses have significantly higher training runtime than ranking loss function version of all the KGE models. The runtime of the ranking loss functions in plots appear to be constant, however, it is growing with a constant increase related to the growth of the training data. On the other hand, the multi-class loss functions have a linear growth which corelates to the growth of the training data. This shows the significant difference between both training loss approaches in terms of the scalability of the training process.

The results also show that the multi-class loss have different growth slopes if the DistMult, ComplEx and TriModel approaches. These different results from there different techniques in modelling embedding interactions which have a significant effect on the training time with 1-vs-all negative sampling (multi-class losses).

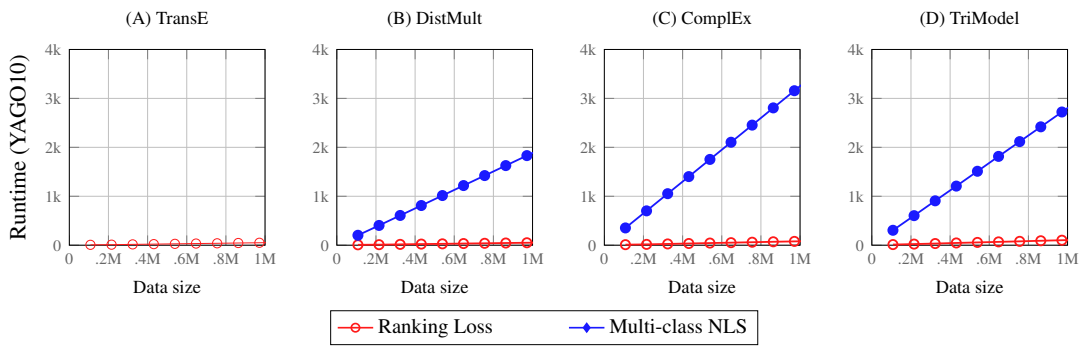

Figure 4.4 – A set of plots which describe the relation between the training runtime and the dataset size for the multi-class and ranking losses for different models on the YAGO10 dataset. The results reported in this figure are acquired by training KGE models with a small embedding size (10) for 20 iterations only. The TransE model's plot reports only results for ranking loss functions.

4.4 KGE training hyperparameters

In this section we discuss the effects of training hyperparameters on both the accuracy and scalability of knowledge graph embedding models. We first discuss the effects in terms of scalability and we then discuss the implications of changes of hyperparameters on the accuracy of KGE models in the task of link prediction.

4.4.1 Training hyperparameters effects on KGE scalability

Knowledge graph embedding models are famous for their high quality predictions with high scalability (Nickel, Murphy, Tresp & Gabrilovich 2016*b*). Most of the KGE methods employ linear transformations such as vector translations and vector diagonal products to learn interactions between embeddings, therefore, they operate within linear time and space complexity (Trouillon et al., Lacroix et al., Mohamed & Novácek 2016, 2018, 2019).

Despite the high scalability of the training process KGE models, they require hyperparameters training routine which is time consuming due to the large hyperparameters search space. Traditional, the hyperparameters search is executed using grid search for the best hyperparameters for each model on each new dataset. The training hyperparameters of KGE models include embedding size, negative samples per positive, batch size, etc. Kadlec et. al. (Kadlec et al. 2017) have shown that minor changes to these hyperparameters can yield significantly different results in terms the models' resulting accuracy. The changes in these hyperparameters have also have an impact on their runtime where changes in hyperparameters such as the embedding size and the number of sampled negatives can affect the memory space used during training.

We performed an experimental evaluation for four different KGE models: TransE, DistMult,

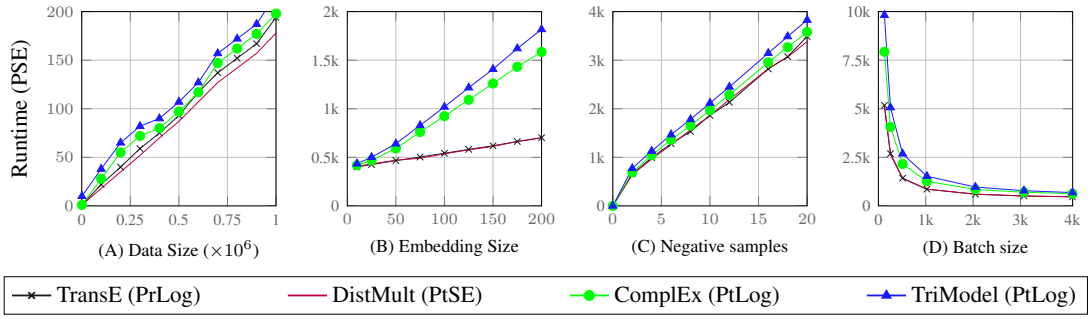

Figure 4.5 – A set of line plots which describe the changes of training data sizes and training hyperparameters and their effects on the trainign runtime of the TransE, DistMult, TriModel and Complex models on the PSE dataset. The runtime is reported in second for all the plots.

Complex and TriModel, where we examine the effects of changes of the training hyperparameters and data size on their training runtime. We use the PSE benchmarking dataset (Zitnik et al. 2018) —our largest benchmarking dataset— to show the effect of hyperparameters on training runtimes of KGE models.

Fig. 4.5 shows the outcome results of our experiments across the different investigated training hyperparameters. The plot "A" shows the relation between the training runtime and the size of the processed data. The plot shows that all the four investigated have a linear relation between their training runtime and the investigated data size. The plot also shows that the investigated models have a consistent growth in terms of their runtime across all the data sizes. The DistMult model consistency achieves the smallest runtime followed by the TransE, DistMult, TriModel and ComplEx models respectively.

Plot "B" shows the relationship between the training runtime and the model embedding size. The plot shows that all the investigated models have a linear growth of their training runtime corresponding to the growth of the embeddings size. However, the growth rate of the TransE and DistMult models is considerably smaller than the growth of both the ComplEx and TriModel models. This occurs as both the TransE and DistMult models use a single vector to represent each of their embeddings while the ComplEx and TriModel models use two and three vectors respectively. Despite the better scalability of both the TransE and DistMult models, the ComplEx and TriModel models generally achieve better predictive accuracy than the TransE and DistMult models (Mohamed & Novácek 2019).

The plot "C" shows the relation between the runtime of KGE models and the number of negative samples they use during training. The plot shows that there is a positive linear relation between training runtime and the number of negative samples–where all the KGE models have similar results across all the investigated sampling sizes. The TriModel, however, consistently have the highest runtime compared to other models.

Plot "D" shows the effects of the size of the batch on the training runtime. The plot shows an exponential decay of the training runtime with the linear growth of the data batch size. The KGE models process all the training data for each training iteration *i.e.* epoch, where the data is divided into batches for scalability and generalisation purposes. Therefore, the increase of the training data batch sizes lead to a decrease of the number of model executions for each training iteration. Despite the high scalability that can be achieved with large batch sizes, the best predictive accuracy is often achieved using small data batch sizes. Usually, the most efficient training data batch size is chosen during a hyper-parameter grid search along with other hyperparameters such as the embedding size and the number of negative samples.

Analysis of the predictive scalability experiments

Our experiments on the effects of training parameters of KGE models on the models' scalability suggests the followings:

- The results confirm that KGE models have linear time complexity as shown in Fig. 4.5 where the models' runtime grows linearly corresponding to increase of both data size and embedding vector sizes.
- The results also confirm that models such as the TriModel and Complex model which have more than one embedding vector for each entity and relation require more training time compared to models with only one embedding vector.
- The results also show that the training batch size has a significant effect on the training runtime therefore, larger batch sizes are suggested to significantly enhance scalability of the training of KGE models.

4.4.2 Training hyperparameters effects on KGE accuracy

In the following, we study the relation between changes in different training hyperparameters and the accuracy of KGE models. We perform an experimental evaluation where we examine the changes of the accuracy of KGE models in terms of MRR compared to the changes of the model's training hyperparameters. Fig. 4.6 shows the outcome of our experiments where it presents a series of plots for the changes of the MRR corresponding to changes of the training hyperparameters of different KGE models on the NELL239, WN18RR, FB15k-237 and YAGO10 datasets.

The datasets reported in the illustration of Fig. 4.6 are sorted in a descending order from up to down in terms of the number of facts contained in the dataset (dataset size). In the first row of

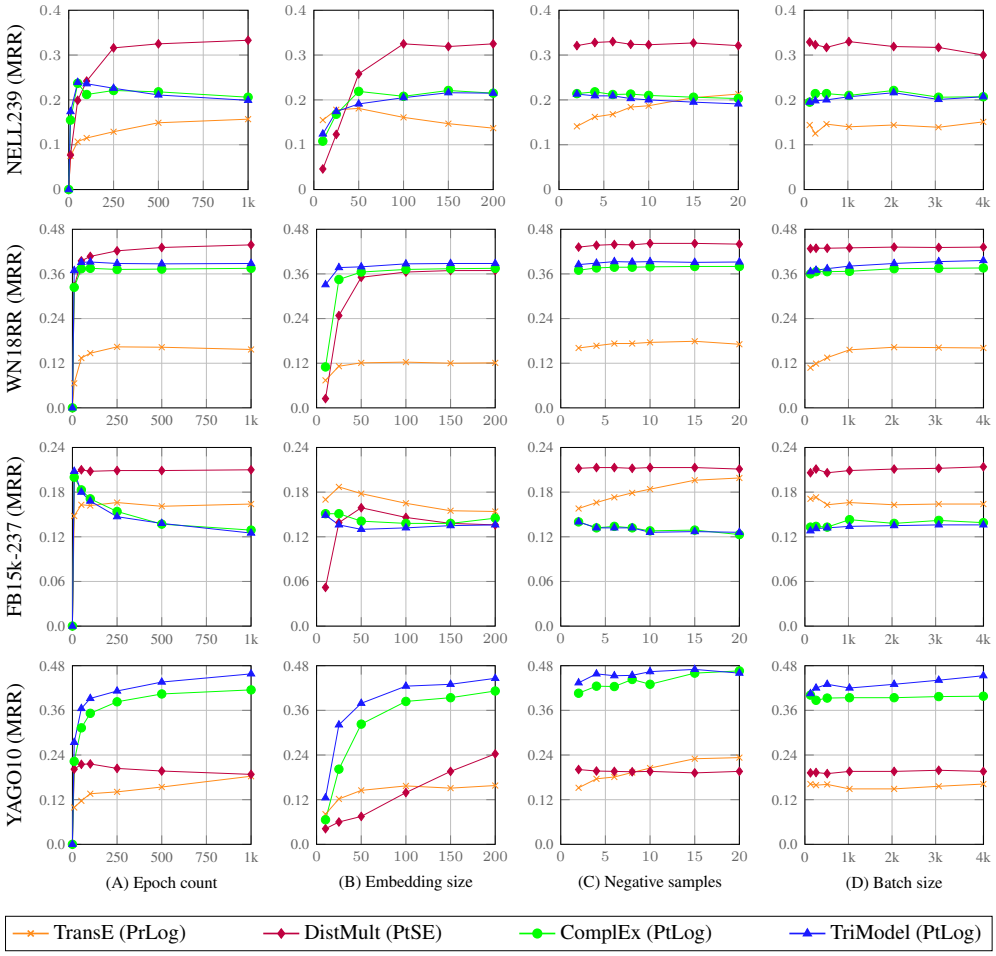

Figure 4.6 – A set of plots which describe the effects of training hyperparameters of KGE models and their effects on the models' accuracy in terms of MRR on different benchmarking datasets. The base hyperparameters for our experiments are: {embedding size (k = 150), negative samples per positive (n=2), batch size (b = 2048), number of epochs (e = 500), optimizer (AMSgrad), learning rate (lr = 0.01)}

plots corresponding to the experiments done on the NELL239 dataset (the smallest dataset), the results show that the changes of the number of training iteration (epochs) and the embedding size have. On the other hand, the negative samples and batch size hyperparameters have less relation to the MRR score values where the growth of the batch size insignificantly affects the MRR score for both hyperparameters except for the TransE model which have a positive MRR score relation with the number of negative samples.

The results corresponding to the WN18RR dataset also show that the epoch count and embedding size have a significant effect of the KGE accuracy. The results also show that both hyperparameters have positive relation with the MRR score of KGE models compared to the variable relation on the NELL dataset. The results also show that the MRR score of KGE models

stabilise after 250 training iterations and embedding size of 50 where the increases above these values does not have a significant effect on the MRR score of KGE models. Similar to the results on the NELL230, the negative samples and batch size hyperparameters show no significant relation with the MRR scores of different KGE models.

The results of the FB15k-237 dataset have a different relation pattern corresponding to the number of training iterations compared to other datasets where the MRR scores of different KGE models have negative or no relation with the changes of the number of training iterations. For example, the MRR scores TransE and DistMult approximately have the sample values corresponding to all the different values of the number of training iterations. On the other hand, the TriModel and ComplEx models have a negative relation with the number of training iteration where their MRR scores decrease with the growth of the training iterations count. The changes of the embedding size of the KGE models on the FB15k-237 dataset also shows valiant patterns where different KGE models have different relation to the change of the size of the embeddings.

On the other hand, the changes of the batch size and number of negative samples have a similar pattern as in the NELL239 dataset where models' MRR scores have no significant relation with the changes of both hyperparameters except for the TransE model which have a positive MRR score relation with the number of negative samples.

The results of the YAGO10 dataset (the largest dataset) shows a positive relation between the number of training iterations and the MRR score of the TransE, TriModel and ComplEx models. On the other hand, the DistMult model have negative relation with the the number of training iterations. On the other hand, the results show that all models have a positive relation between their MRR scores and the size of the embeddings. The results of the batch size and negative samples hyperparameters show less relation to the MRR scores as in the previous dataset, however, there is a noticeable low positive relation between the number of negative samples and the MRR scores of the TransE, TriModel and ComplEx models.

Analysis of the predictive accuracy experiments

From the above discussed observations, we can suggest the following:

- Changes on the embedding vectors' size have the biggest effect on the predictive accuracy of KGE models. Thus, we suggest careful selection of this parameter by search through a larger search space of possible embedding sizes.

- The increased number of training iterations can sometimes have a negative effect on the outcome predictive accuracy. Thus, we suggest using early stopping techniques to decide when to stop model training before accuracy decreases.

- Both the number of negative samples and batch sizes showed small effect on the predictive accuracy of KGE models. Thus, these parameter can be assigned fixed values or smaller search spaces to help decrease the time required for the tuning of hyperparameters.

4.5 Discussion

In this section, we discuss the compromise between scalability and accuracy in the training of KGE models. We also discuss the properties of some datasets and their relation with KGE interaction functions. We finally discuss the compatibility between specific KGE scoring and loss functions.

4.5.1 The compromise between scalability and accuracy

We have shown that KGE models achieve their best result in terms of accuracy using multi-class loss functions. However, these functions depend on the 1-vs-all negative sampling which is time-consuming as we have shown in Section 4.3.5. On the other hand, KGE models with ranking-based loss function are significantly more scalable but they have less accurate predictions compared to the multi-class losses. This variability between the capabilities of the two approaches results in a compromise between the scalability and accuracy of KGE models when choosing loss functions for KGE models. In our experiments, we found that the training runtime of multi-class loss functions is affected by the entity count in the dataset along with the dataset size where datasets with higher number of entities require more training time than others even if they have the same size.

We ran all our experiments on GPU where we found out that the multi-class based models consume a large amount of the GPU memory. This, therefore, forced us to use small training batch sizes to fit to the GPU's memory specially on large datasets. The use of these smaller batches resulted in longer training runtime due to the increased number of training iterations over the batches. On the other hand, ranking based loss functions have significantly lower GPU memory consumption compared to the multi-class loss functions. However, there memory consumption grow positively with relation to the number of used negative samples.

We thus suggest that KGE models with multi-class losses can be used comfortably used for training of small size knowledge graphs (Less than 5M facts). We also suggest the use of multiple GPUs when available for the grid-search process of KGE models with multi-class objectives.

4.5.2 The relationship between dataset properties and embedding interaction functions

In our experiments, it is noticeable that the tensor factorisation based methods such as the DistMult, TriModel and ComplEx models consistently have better accuracy than distance based models such as the TransE model on all benchmark in the ranking losses configuration as shown in Table 4.2. However, we can also see that the TransE model significantly outperforms all other ranking based models on the FB15k-237 dataset. A further study of Nguyen et. al. (Nguyen et al. 2018) also shows that translation based methods achieve significantly higher accuracy than tensor factorisation based methods in terms ob both MRR and Hits@10.

We suggest that this can be due to specific properties in the dataset which is compatible with translation based embedding interaction approaches compared to tensor factorisation methods. We also intend to study this specific relation in future works where we intend to investigate different properties of knowledge graph and their possible relations to specific KGE embedding components.

4.5.3 Compatibility between scoring and loss functions

In the ranking loss functions experiments, we can see that the TransE model achieves its best result using pairwise loss functions while its version with the pointwise loss function have significantly worse results. On the other hand other tensor factorisation based approaches such as the DistMult and ComplEx models achieve their best results with their version which uses pointwise loss functions such as the pointwise squared error and logistic losses. The pairwise loss function versions of these models also have significantly worse results in terms of both the MRR and Hits@10 metrics on all benchmarks as shown in Table 4.2.

5 Multi-Part Graph Embeddings

5.1 Introduction

In recent years, knowledge graph embedding (KGE) models have witnessed rapid developments that have allowed them to excel in the task of link prediction for knowledge graphs (Wang et al. 2017). They learn embeddings using different techniques like tensor factorisation, latent distance similarity and convolutional filters in order to rank facts in the form of (subject, predicate, object) triples according to their factuality. In this context, their tensor factorisation based versions like the DistMult (Yang et al. 2015*b*) and the ComplEx (Trouillon et al. 2016) models are known to provide state-of-the-art results within linear time and space complexity (Wang et al. 2017). The scalable and efficient predictions achieved by these models have encouraged researchers to investigate advancing the DistMult and the ComplEx models by utilising different training objectives and regularisation terms (Kadlec et al., Lacroix et al. 2017, 2018).

In this chapter, our objective is to propose a new factorisation based knowledge graph embedding model that extends the works of the DistMult and the ComplEx models while preserving their linear time and space complexity. We achieve that by modifying two of their main components: the embedding representation, and the embedding interaction function.

While both the DistMult and the ComplEx models use the bilinear product of the subject, the predicate and the object embeddings as an embedding interaction function to encode knowledge facts, they represent their embeddings using different systems. The DistMult model uses real values to represent its embedding vectors, which leads to learning a symmetric representation of all predicates due to the symmetric nature of the product operator on real numbers. On the other hand, the ComplEx model represents embeddings using complex numbers, where each the embeddings of an entity or a relation is represented using two vectors (real and imaginary parts). The ComplEx model also represents entities in the object mode as the complex conjugate of their subject form (Trouillon et al. 2016). This enables the ComplEx model to encode both symmetric and asymmetric predicates.

Since the embeddings of the ComplEx models are represented using two part embeddings (real and imaginary parts), their bilinear product (ComplEx's embedding interaction function) consists of different interaction components unlike the DisMult model with only one bilinear product component. Each of these components is a bilinear product of a combination of real and imaginary vectors of the subject, the predicate and the object embeddings, which gives the ComplEx model its ability to model asymmetric predicates.

In this work, we investigate both the embedding representation and the embedding interaction components of the ComplEx model, where we show that the ComplEx embedding interaction components are sufficient but not necessary to model asymmetric predicates. We also show that our proposed model, TriVec , can efficiently encode both symmetric and asymmetric predicates using simple embedding interaction components that rely on embeddings of three parts. To assess our model compared to the ComplEx model, we carry experiments on both models using different training objectives and regularisation terms, where our results show that our new model, TriVec , provide equivalent or better results than the ComplEx model on all configurations. We also propose a new NELL (Mitchell et al. 2015) based benchmarking dataset that contains a small number of training, validation and testing facts that can be used to facilitate fast development of new knowledge graph embedding models.

5.2 Background

Knowledge graph embedding models learn low rank vector representation *i.e.* embeddings for graph entities and relations. In the link prediction task, they learn embeddings in order to rank knowledge graph facts according to their factuality. The process of learning these embeddings consists of different phases. First, they initialise embeddings using random noise. These embeddings are then used to score a set of true and false facts, where a score of a fact is generated by computing the interaction between the fact's subject, predicate and object embeddings using a model dependent scoring function. Finally, embeddings are updated by a training loss that usually represents a min-max loss, where the objective is to maximise true facts scores and minimise false facts scores.

In this section we discuss scoring functions and training loss functions in state-of-the-art knowledge graph embedding models. We define our notation as follows: for any given knowledge graph, E is the set of all entities, R is the set of all relations *i.e.* predicates, N_e and N_r are the numbers of entities and relations respectively, T is the set of all known true facts, e and w are matrices of sizes $N_e \times K$ and $N_r \times K$ respectively that represent entities and relations embeddings of rank K, ϕ_{spo} is the score of the triple (s, p, o), and $\mathcal{L}$ is the model's training loss.

5.2.1 Scoring Functions

Knowledge graph embedding models generate scores for facts using model dependent scoring functions that compute interactions between facts' components embeddings. These functions use different approaches to compute embeddings interactions like distance between embeddings (Bordes et al. 2013), embedding factorisation (Trouillon et al. 2016) or embeddings convolutional filters (Dettmers et al. 2018).

In the following, we present these approaches and specify some examples of knowledge graph embedding models that use them.

• *Distance-based embeddings interactions*: The Translating Embedding model (TransE) (Bordes et al. 2013) is one of the early models that use distance between embeddings to generate triple scores. It interprets triple's embeddings interactions as a linear translation of the subject to the object such that $e_s + w_p = e_o$, and generates a score for a triple as follows:

$$\phi_{spo}^{\text{TransE}} = \| e_s + w_p - e_o \|_{l1/l2}, \tag{5.1}$$

where true facts have zero score and false facts have higher scores. This approach provides scalable and efficient embeddings learning as it has linear time and space complexity. However, it fails to provide efficient representation for interactions in one-to-many, many-to-many and many-to-one predicates as its design assumes one object per each subject-predicate combination.

• *Factorisation-based embedding interactions*: Interactions based on embedding factorisation provide better representation for predicates with high cardinality. They have been adopted in models like DistMult (Yang et al. 2015*b*) and ComplEx (Trouillon et al. 2016). The DistMult model uses the bilinear product of embeddings of the subject, the predicate, and the object as their interaction, and its scoring function is defined as follows:

$$\phi_{spo}^{\text{DistMult}} = \sum_{k=1}^{K} e_{s_k} w_{p_k} e_{o_k} \tag{5.2}$$

where e_{s_k} is the k-th component of subject entity s embedding vector e_s. DistMult achieved a significant improvement in accuracy in the task of link prediction over models like TransE. However, the symmetry of embedding scoring functions affects its predictive power on asymmetric predicates as it cannot capture the direction of the predicate. On the other hand, the ComplEx model uses embedding in a complex form to model data with asymmetry. It models embeddings interactions using the the product of complex embeddings, and its scores are defined as follows:

$$\phi_{spo}^{\text{ComplEx}} = \text{Re}(\sum_{k=1}^{K} e_{s_k} w_{p_k} \overline{e}_{o_k}) = \sum_{k=1}^{K} e_{s_k}^r w_{p_k}^r e_{o_k}^r + e_{s_k}^i w_{p_k}^r e_{o_k}^i + e_{s_k}^r w_{p_k}^i e_{o_k}^i - e_{s_k}^i w_{p_k}^i e_{o_k}^r \tag{5.3}$$

Table 5.1 – A comparison between the ComplEx model and different variants of its scoring functions on standard benchmarking datasets

Model	Definition	NELL239		WN18RR		FB237	
		MRR	H@10	MRR	H@10	MRR	H@10
ComplEx	i1+i2+i3-i4	0.35	0.51	0.44	0.51	0.22	0.41
ComplEx-V1	i1+i2+i3	0.34	0.51	0.45	0.52	0.22	0.40
ComplEx-V2	i2+i3+i4	0.34	0.50	0.44	0.51	0.21	0.38
ComplEx-V3	i1+i2-i4	0.34	0.51	0.45	0.52	0.22	0.40
ComplEx-V4	i1+i3-i4	0.33	0.50	0.45	0.50	0.21	0.39

where $Re(x)$ represents the real part of complex number x and all embeddings are in complex form such that $e, w \in C$, e^r and e^i are respectively the real and imaginary parts of e, and $\overline{e}_o$ is the complex conjugate of the object embeddings e_o such that $\overline{e}_o = e_o^r - i e_o^i$ and this introduces asymmetry to the scoring function. Using this notation, ComplEx can handle data with asymmetric predicates, and to keep scores in the real spaces it only uses the real part of embeddings product outcome. ComplEx preserves both linear time and linear space complexities as in TransE and DistMult, however, it surpasses their accuracies in the task of link prediction due to its ability to model a wider set of predicate types.

• *Convolution-based embeddings interactions*: Following the success of convolutional neural networks image processing tasks, models like R-GCN (Schlichtkrull et al. 2018) and ConvE (Dettmers et al. 2018) utilized convolutional networks to learn knowledge graph embeddings. The R-GCN model learns entity embeddings using a combination of convolutional filters of its neighbours, where each predicate represent a convolution filter and each neighbour entity represents an input for the corresponding predicate filter. This approach is combined with the DistMult model to perform link prediction. Meanwhile, the ConvE model concatenates subject and predicate embeddings vectors into an image (a matrix form), then it uses a 2D convolutional pipeline to transform this matrix into a vector and computes its interaction with the object entity embeddings to generate a corresponding score as follows:

$$\phi_{spo}^{\text{ConvE}} = f(vec(f([\overline{e_s}; \overline{w_p}] * \omega))W)e_o \tag{5.4}$$

where $\overline{e_s}$ and $\overline{w_p}$ denotes a 2D reshaping of e_s and w_p, ω is a convolution filter, f denotes a non-linear function, $vec(x)$ is a transformation function that reshape matrix x of size $m \times n$ into a vector of size $mn \times 1$.

5.3 The TriVec Model

In this section, we motivate for the design decision of TriVec model, and we present its way to model embeddings interaction and training loss.

5.3.1 Motivation

Currently, models using factorisation-based knowledge graph embedding approaches like DistMult and ComplEx achieve state-of-the-art results across all benchmarking datasets (Lacroix et al. 2018). In the DistMult model, embeddings interactions are modelled using a symmetric function that computes the product of embeddings of the subject, the predicate and the object. This approach was able to surpass other distance-based embedding techniques like TransE (Yang et al. 2015*b*). However, it failed to model facts with asymmetric predicate due to its design. The ComplEx model tackle this problem using a embeddings in the complex space where its embeddings interactions use the complex conjugate of object embeddings to break the symmetry of the interactions. This approach provided significant accuracy improvements over DistMult as it successfully models a wider range of predicates.

The ComplEx embeddings interaction function (defined in Sec. 5.2) can be redefined as a simple set of interactions of two part embeddings as follows:

$$\phi_{spo}^{\text{ComplEx}} = \sum_k i_1 + i_2 + i_3 - i_4 \tag{5.5}$$

where $\sum_k$ is the sum of all embeddings components of index $k = \{1, ..., K\}$, and interactions i_1, i_2, i_3 and i_4 are defined as follows:

$$i_1 = e_s^1 w_p^1 e_o^1, \quad i_2 = e_s^2 w_p^1 e_o^2, \quad i3 \quad = e_s^1 w_p^2 e_o^2, \quad i4 = e_s^2 w_p^2 e_o^1$$

where e^1 represents embeddings part 1, and e^2 is part 2 (1 → real and 2 → imaginary). Following this notation, we can see that the ComplEx model is a set of two symmetric interaction i_1 and i_2 and two asymmetric interactions i_3 and i_4. Furthermore, this encouraged us to investigate the effect of using other forms of combined symmetric and asymmetric interactions to model embeddings interactions in knowledge graph embeddings. We investigated different combination of interactions i_1, i_2, i_3 and i_4, and we have found that by removing and/or changing the definition of one of these interactions (maintaining that the interactions use all triple components) will preserve similar or insignificantly different prediction accuracy across different benchmarking datasets (See Table 5.1). This led us to investigate other different forms of interactions that uses a combination of symmetric and asymmetric interactions where we found that using embeddings of three parts can lead to better predictive accuracy than the ComplEx and the DistMult models.

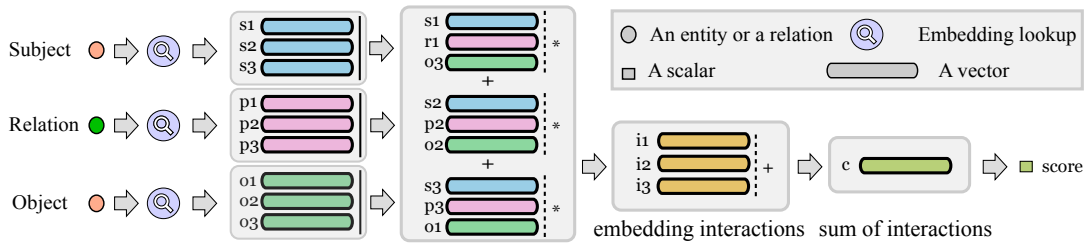

Figure 5.1 – Visual explanation for the flow of the TriVec model's scoring function, where embedding interactions are represented by $i_1 = e_s^3 w_p^3 e_o^1$, $i_2 = e_s^2 w_p^2 e_o^2$, and $i_3 = e_s^1 w_p^1 e_o^3$.

5.3.2 TriVec Embeddings Interactions

In the TriVec model, we represent each entity and relation using three embedding vectors such that the embedding of entity i is $\{e_i^1, e_i^2, e_i^3\}$ and the embedding of relation j is $\{w_j^1, w_j^2, w_j^3\}$ where e^m denotes the m part of the embeddings and where $m \in 1,2,3$ is used to represent the three embeddings parts.

The TriVec model is a tensor factorisation based model, where its embeddings interaction function (scoring function) is defined as follows:

$$\phi_{spo}^{\text{TriPart}} = \sum_{k=1}^{K} e_{sk}^1 w_{pk}^1 e_{ok}^3 + e_{sk}^2 w_{pk}^2 e_{ok}^2 + e_{sk}^3 w_{pk}^3 e_{ok}^1 \tag{5.6}$$

where k denotes the index of the embedding vector entries. The model uses a set of three interactions: one symmetric interaction: $(e_s^2 w_p^2 e_o^2)$ and two asymmetric interactions: $(e_s^1 w_p^1 e_o^3)$ and $(e_s^3 w_p^3 e_o^1)$ as shown in Fig. 5.1. This approach models both symmetry and asymmetry in a simple form similar to the DistMult model where the DisMult model can be seen as a special case of the TriVec model if the first and third embeddings part are equivalent ($e^1 = e^3$).

5.3.3 Training the TriVec embeddings

Trouillon et. al. (Trouillon & Nickel 2017) showed that despite the equivalence of HolE and ComplEx models' scoring functions, they produce different results as they use different loss functions. They concluded that the logistic loss version of ComplEx outperforms its hinge loss version. In addition, we have investigated different other ranking losses with the ComplEx model, and we have found that squared error loss can significantly enhance the performance of ComplEx on multiple benchmarking datasets.

The TriVec model performs its learning process using two different training loss configurations: the traditional ranking loss and the multi-class loss (*cf.* Section4.3). In the ranking loss configuration, the TriVec model uses the squared error (Eq. 4.1) and the logistic loss (Eq. 4.3) to model its training error, where a grid search is performed to choose the optimal loss representation

for each dataset. In the multi-class configuration, it uses the negative-log softmax loss (Eq. 4.6) with the nuclear 3-norm regularisation (Lacroix et al. 2018).

We also consider the use of predicate reciprocals in training as described in Lacroix et al. (2018), where inverses of training predicates are added to the training set and trained with their corresponding original facts as shown in the following:

$$\begin{aligned} \overset{\text{TriModel}}{\underset{spo}{\mathcal{L}}} &= -\phi_{spo} + \log(\sum_{o'} \exp(\phi_{spo'})) \\ &\quad -\phi_{spo} + \log(\sum_{s'} \exp(\phi_{o(p+N_r)s})) \\ &+ \frac{\lambda}{3} \sum_{k=1}^{K} \sum_{m=1}^{3} (|e_s^m|^3 + |w_p^m|^3 + |w_{p+N_r}^m|^3 + |e_o^m|^3) \end{aligned} \tag{5.7}$$

where predicate $p + N_r$ is the inverse of the predicate p where the model learns and evaluates inverse facts using inverses of their original predicates. For all the multi-class configurations, the TriVec model regularises the training facts embeddings using a dropout layer (Srivastava et al. 2014) with weighted probability that it learns during the grid search.

5.4 Experiments

In this section, we discuss the setup of our experiments where we present the evaluation protocol, the benchmarking datasets and our implementation details.

5.4.1 Benchmarking Datasets

In our experiments we use six knowledge graph benchmarking datasets:

- WN18 & WN18RR: subsets of the WordNet dataset (Miller 1995) that contains lexical information of the English language (Bordes et al., Dettmers et al. 2013, 2018).
- FB15k & FB15k-237: subsets of the Freebase dataset (Bollacker et al. 2008) that contains information about general human knowledge (Bordes et al., Toutanova et al. 2013, 2015).
- YAGO10: a subset of the YAGO3 dataset that contains information mostly about people and their citizenship, gender, and professions knowledge (Dettmers et al. 2018).
- NELL239: a subset of the NELL dataset (Mitchell et al., Gardner & Mitchell 2015, 2015) that we have created to test our model, which contains general knowledge about people, places, sports teams, universities, etc. We developed this dataset to be used for model prototyping as it is smaller in size compared to other standard benchmarks while having a diverse sets of entities and relations.

Table 5.2 – Statistics of the benchmarking datasets.

Dataset	Entity count	Relation count	Training	Validation	Testing
WN18	41k	18	141k	5k	5k
WN18RR	41k	11	87k	3k	3k
FB15k	15k	1k	500k	50k	60k
FB15k-237	15k	237	272k	18k	20k
YAGO10	123k	37	1M	5k	5k
NELL239	48k	239	74k	3k	3k

Table 5.2 contains statistics about our experiments' benchmarking datasets [1].

5.4.2 Implementation

We use TensorFlow framework (GPU) along with Python 3.5 to perform our experiments. All experiments were executed on a Linux machine with processor Intel(R) Core(TM) i70.4790K CPU @ 4.00GHz, 32 GB RAM, and an nVidia Titan Xp GPU.

5.4.3 Experiments Setup

We perform our experiments in two different configurations:
(1) Ranking loss based learning: the models are trained using a ranking based loss function, where our model chooses between squared error loss and logistic loss using grid search.

(2) Multi-class loss based learning: the models is trained using a multi-class based training functions, where our model uses the softmax negative log loss functions described in Eq. 4.6 and Eq. 5.7.

In all of our experiments we initialise our embeddings using the Glorot uniform random generator (Glorot & Bengio 2010) and we optimise the training loss using the Adagrad optimiser, where the learning rate $\text{lr} \in \{0.1, 0.3, 0.5\}$, embeddings size $K \in 50, 75, 100, 150, 200$ and batch size $b \in \{1000, 3000, 5000, 8000\}$ except for YAGO10 where we only use $b \in \{1000, 2000\}$. The rest of the grid search hyper parameters are defined as follows: in the ranking loss approach, we use the negative sampling ratio $n \in \{2, 5, 10, 25, 50\}$ and in the multi-class approach we use regularisation weight $\lambda \in \{0.1, 0.3, 0.35, 0.01, 0.03, 0.035\}$ and dropout $d \in \{0.0, 0.1, 0.2, 0.01, 0.02\}$. The number of training epochs is fixed to 1000, where in the ranking loss configuration we do an early check every 50 epochs to stop training when MRR stop improving on the validation set to prevent over-fitting.

[1] All the benchmarking datasets can be downloaded using the following url: https://figshare.com/s/88ea0f4b8b139a13224f

Table 5.3 – Link prediction results on standard benchmarking datasets. ⋆ Results taken from (Trouillon et al. 2016) for the WN18 and FB15k while the results on other datasets are extracted from our own experiments.

	Model	WN18		WN18RR		FB15k		FB15k-237		YAGO10		NELL239	
		MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10
Ranking loss	CP	0.08	0.13	-	-	0.33	0.53	-	-	-	-	-	-
	TransE ⋆	0.52	0.94	0.20	0.47	0.52	0.76	.29	0.48	0.27	0.44	0.27	0.43
	ConvKB	-	-	0.25	0.53	-	-	**0.40**	**0.52**	-	-	-	-
	DistMult ⋆	0.82	0.94	0.43	0.49	0.65	0.82	0.24	0.42	0.34	0.54	0.31	0.48
	ComplEx ⋆	0.94	0.95	0.44	0.51	0.70	0.84	0.22	0.41	0.36	0.55	0.35	0.52
	R-GCN	0.81	0.96	-	-	0.70	0.84	0.25	0.42	-	-	-	-
	TriVec	**0.95**	**0.96**	**0.50**	**0.57**	**0.73**	**0.86**	0.25	0.43	**0.46**	**0.62**	**0.37**	**0.53**

In the evaluation process, we only consider filtered MRR and Hits@10 metrics (Bordes et al. 2013). In addition, in the ranking loss configuration, TriVec model uses a softmax normalisation of the scores of objects and subjects corruptions, that a score of a corrupted object triple (s, p, o_i) is defined as:

$$\phi_{spo_i} = \frac{\exp(\phi_{spo_i})}{\sum_{o' \in E} \exp(\phi_{spo'})},$$

similarly, we apply a softmax normalisation to the scores of all possible subject entities.

5.5 Results and Discussion

In this section we discuss findings and results of our experiments shown in Table 5.3 and Table 5.4, where the experiments are divided into two configurations: models with ranking loss functions and models with multi-class based loss functions.

5.5.1 Results of The Ranking Loss Configuration

In the results of the ranking loss configuration shown in Table 5.3, the results show that the TriVec model achieves best results in terms of MRR and hits@10 in five out of six benchmarking datasets with a margin of up to 10% as in the YAGO10 dataset. However, on the FB15k-237 ConvKB (Nguyen et al. 2018) retains state-of-the-art results in terms of MRR and Hits@10. Results also show that the factorisation based models like the DistMult, ComplEx, R-GCN and TriVec models generally outperform distance based models like the TransE and ConvKB models. However, on the FB15k-237 dataset, both distance based models have higher predictive accuracy scores compared to other factorisation based models with a margin of up to 15% in the case of the ConvKB and the TriVec model. We intend to perform further analysis on this dataset compared to other datasets to investigate why tensor factorisation models fail to provide state-of-the-art results in future works.

Table 5.4 – Link prediction results on standard benchmarking datasets. † Results taken from (Lacroix et al. 2018) by re-running their provided code with embedding size (K) limited to 200.

	Model	WN18		WN18RR		FB15k		FB15k-237		YAGO10		NELL239	
		MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10
Multi-class loss	ConvE	0.94	0.95	0.46	0.48	0.75	0.87	0.32	0.49	0.52	0.66	0.37	0.45
	CP-N3 †	0.12	0.18	0.08	0.14	0.35	0.56	0.22	0.42	0.40	0.64	-	-
	ComplEx-N3 †	0.92	0.95	0.44	0.52	0.58	0.79	0.30	0.51	0.46	0.67	-	-
	CP-N3-R †	0.93	0.94	0.41	0.45	0.62	0.78	0.30	0.47	0.55	0.69	-	-
	ComplEx-N3-R †	**0.95**	**0.96**	**0.47**	**0.54**	0.79	0.88	**0.35**	**0.54**	**0.57**	0.70	-	-
	TriVec - N3	**0.95**	**0.96**	**0.47**	**0.54**	**0.84**	**0.91**	**0.35**	**0.54**	**0.57**	**0.71**	**0.41**	0.57
	TriVec -N3-R	**0.95**	**0.96**	**0.47**	**0.54**	0.81	**0.91**	**0.35**	**0.54**	**0.57**	0.70	**0.41**	**0.58**

5.5.2 Results of The Multi-class Loss Configuration

Results of the multi-class based approach show that TriVec model provide state-of-the-art result on all benchmarking datasets, where the ComplEx models provide equivalent results on 3 out 6 datasets. Our reported results of the ComplEx model with multi-class log-loss introduced by Lacroix et. al. (Lacroix et al. 2018) are slightly different from their reported results as we re-evaluated their models with restricted embeddings size to a maximum of 200. In their work they used an embedding size of 2000, which is impractical for embedding knowledge graphs in real applications. And other previous works using the TransE, DistMult, ComplEx, ConvE, and ConvKB models have limited their experiments to a maximum embedding size of 200. In our experiments, we limited our embedding size to 200 and we have re-evaluated the models of (Lacroix et al. 2018) using the same restriction for a fair comparison [2].

5.5.3 Ranking and Multi-class Approaches

In the link prediction task, the objective of knowledge graph embedding models is to learn embeddings that rank triples according to their faculty. This is achieved by learning to rank original true triples against other negative triple instances, where the negative instances are modelled in different ways in ranking approaches and multi-class loss approaches.

In learning to rank approach, models use a ranking loss *e.g.* pointwise or pairwise loss to rank a set of true and negative instances (Chen et al. 2009), where negative instances are generated by corrupting true training facts with a ratio of negative to positive instances (Bordes et al. 2013). This corruption routine happens by changing either the subject or object of the true triple instance. In this configuration, the ratio of negative to positive instances is traditionally learnt using a grid search, where models compromise between the accuracy achieved by increasing the ratio and the runtime required for training.

On the other hand, multi-class based models train to rank positive triples against all their

[2] We have used the code provided at: https://github.com/facebookresearch/kbc for the evaluation of the models: CP-N3, CP-N3-R, ComplEx-N3 and ComplEx-N3-R

possible corruptions as a multi-class problem where the range of classes is the set of all entities. For example, training on a triple (s, p, o) is achieved by learning the right classes "s" and "o" for the pairs (?, p, o) and (s, p, ?) respectively, where the set of possible class is E of size N_e. Despite the enhancements of the predictions accuracy achieved by such approaches (Dettmers et al., Lacroix et al. 2018, 2018), they can have scalability issues in real-world large sized knowledge graphs with large numbers of entities due to the fact that they use the full entities' vocabulary as negative instances (Mnih & Kavukcuoglu 2013).

In summary, our model provides significantly better results than other SOTA models in the ranking setting, which is scalable and thus better-suited to real-world applications. In addition to that, our model has equivalent or slightly better performance than SOTA models on the multi-class approach.

5.6 Summary

In this work, we have presented the TriVec embedding approach, a new tensor factorisation based knowledge graph embedding model which represents knowledge entities and relations using three parts embeddings, where its embedding interaction function encodes both symmetric and asymmetric predicates. We have shown using experimental evaluation that the TriVec approach outperforms other tensor factorisation based models like the ComplEx and the DistMult on different training objectives and across all standard benchmarking datasets. We have also introduced a new benchmarking dataset, NELL239, which can be used to facilitate fast development of new knowledge graph embedding models.

In our future works, we intend to investigate new possible approaches to model embedding interactions of tensor factorisation models, and we intend to analyse the effects of properties of knowledge graph datasets like FB15k-237 on the efficiency of tensor factorisation based models.

Case Studies Part III

6 Knowledge Graph Embeddings in Bioinformatics

6.1 Overview

Biological systems consist of complex interconnected biological entities that work together to sustain life in living systems. This occurs through complex and systematic biological interactions of the different biological entities. Understanding these interactions is key to elucidating the mechanism-of-action of the different biological functions (*e.g.* angiogenesis, metabolism, apoptosis, etc), and thus, understanding causes and activities of diseases and their possible therapies. This encouraged the development of multiple physical and computational methods to assess, verify and infer different types of these interactions. In this chapter, we focus on the use of computational methods for assessing and inferring interactions (associations) between different biological entities at the molecular level. We hereof study the use of knowledge graphs and their embedding models for modelling molecular biological systems and the interactions of their entities.

Initially, basic networks *i.e.* uni-relational graphs, were adopted by early efforts for modelling complex interactions in biological systems (Cohen & Servan-Schreiber, Gibrat et al., Barabási & Oltvai, Albert 1992, 1996, 2004, 2005). Despite their initial success (Janjic & Przulj 2012), these networks could not preserve the semantics of different types of associations between entities. For example, protein-protein interaction networks modelled with basic networks cannot differentiate between different types of interactions such as inhibition, activation, phosphorylation, etc. Therefore, more recent works modelled biological systems using heterogeneous multi-relational networks *i.e.* knowledge graphs, where they utilised different visual (Muñoz et al., Olayan et al. 2017, 2017) and latent representations (Zitnik et al., Mohamed, Nováček & Nounu 2018, 2019) of graph entities to infer associations between them.

In the context of biological applications, knowledge graphs were used to model biological data in different projects such as the UNIPROT (Consortium 2017), Gene Ontology (Consortium 2019) and Bio2RDF (Dumontier et al. 2014) knowledge bases. Moreover, they were the basis of multiple predictive models for drug adverse reactions (Muñoz et al., Zitnik et al. 2017, 2018), drug repurposing (Alshahrani et al., Mohamed, Nováček & Nounu 2017, 2019) and

other predictions for different types of biological concepts associations (Alshahrani et al., Su et al. 2017, 2018). The task of learning biological associations in this context is modelled as link prediction in knowledge graphs (Nickel, Murphy, Tresp & Gabrilovich 2016*a*). Predictive models then try to infer a typed link between two nodes in the graph using two different types of features: graph features and latent-space vector representations.

Graph features models (*i.e.* visual feature models) are part of the network analysis methods which learn their predictions using different feature types such as random walks (Lao et al., Xu et al. 2011, 2017), network similarity (Raman 2010), nodes connecting paths (Gardner & Mitchell 2015) and subgraph paths (Gardner & Mitchell, Mohamed et al. 2015, 2018). They are used in multiple biological predictive applications such as predicting drug targets (Olayan et al. 2017) and protein–protein interaction analysis (Raman 2010). Despite the expressiveness of graph feature models predictions, they suffer from two major drawbacks: limited scalability and low accuracy (Toutanova & Chen, Nickel, Murphy, Tresp & Gabrilovich 2015, 2016*b*). They are also focused on graph local features compared to embedding models which learn global latent features of the processed graph.

Latent feature models *i.e.* embedding models, on the other hand, express knowledge graphs' entities and relations using low-rank vector representations that preserve the graph's global structure. Knowledge graph embedding (KGE) models on the contrary are known to outperform other approaches in terms of both the accuracy and scalability of their predictions despite their lack of expressiveness (Nickel, Murphy, Tresp & Gabrilovich, Wang et al., Lacroix et al. 2016*b*, 2017, 2018).

In recent years, knowledge graph embedding models witnessed rapid developments that allowed them to excel in the task of link prediction (Wang et al., Bordes et al., Nickel et al., Yang et al., Trouillon et al., Dettmers et al., Lacroix et al. 2017, 2013, 2011, 2015*a*, 2016, 2018, 2018). They have then been widely used in various applications including computational biology in tasks like predicting drug target interactions (Mohamed, Nováček & Nounu 2019) and predicting drug polypharmacy side-effects (Zitnik et al. 2018). Despite their high accuracy predictions in different biological inference tasks, knowledge graph embeddings are in their early adoption stages in computational biology. Moreover, many computational biology studies that have used knowledge graph embedding models adopted old versions of these models (Zitnik & Zupan, Abdelaziz et al. 2016, 2017). These versions have then received significant modifications through recent computer science research advances (Lacroix et al. 2018).

In a previous study, Su et. al. (Su et al. 2018) have introduced the use of network embedding methods in biomedical data science. The study compiles a taxonomy of embedding methods for both basic and heterogeneous networks where it discusses a broad range of potential applications and limitation. The study's objective was to introduce the broad range of network embedding methods, however, it lacked deeper investigation into the technical capabilities of the models and how can they be integrated with biological problem. It also did not compare

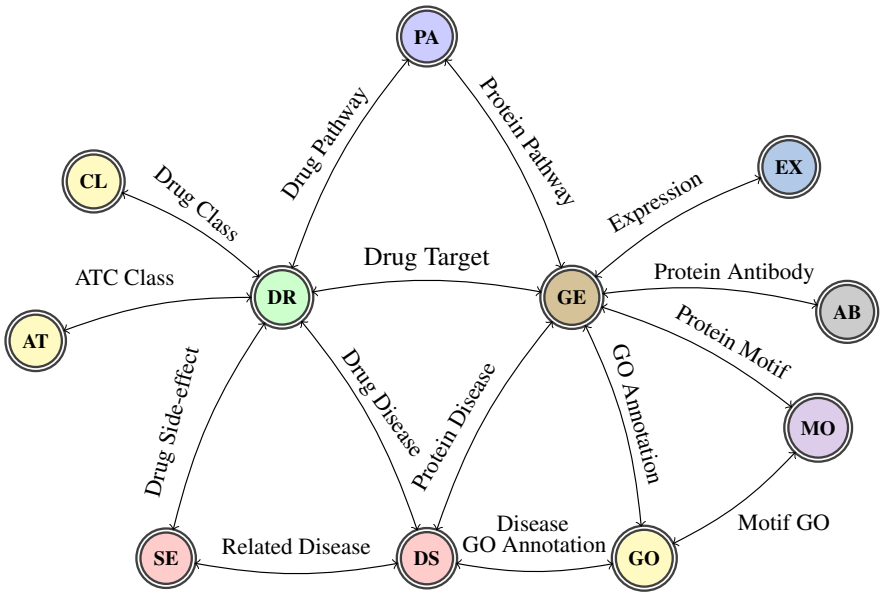

Figure 6.1 – A schema of a knowledge graph that models a complex biological system of different types of entities and concepts. The abbreviation DR represents drugs, GE represents proteins (their genes), EX represents protein expressions (tissues and cell-lines), AB represents protein antibodies, MO represents protein motifs and other sequence annotations, GO represents gene ontology, DS represents diseases, SE represents drug side-effects, AT represents ATC classes, CL represents drug classes and PA represents pathways.

the investigated models in terms of their accuracy and scalability which is essential to assist reader from the biological domain to understand the technical performance differences between these methods.

In this chapter, we exclusively focus on exploring KGE models (Best performing models in terms of both scalability and accuracy) in different biological tasks where we demonstrate the analytical capabilities of KGE models, *e.g.* learning clusters and similarity measures in different biological problems.We also explore the process of building biological knowledge graphs for generic and specific biological inference tasks. We then present computer-based experimental evaluation of knowledge graph embedding models on different tasks such as predicting drug target interactions, drug polypharmacy side-effects and tissue-specific protein functions prediction.

6.2 Background

In this section, we discuss both knowledge graphs and knowledge graph embedding models in the context of biological applications.

6.2.1 Knowledge graphs

A knowledge graph is a data modelling technique that models linked data as a graph, where the graph's nodes represent data entities and its edges represent the relations between these entities. In recent years, knowledge graphs became a popular means for modelling relational data where they were adopted in various industrial and academic applications such as semantic search engines (Qian 2013), question answering systems (Ferrucci et al. 2010) and general knowledge repositories (Mitchell et al. 2015). They were also used to model data from different types of domains such as general human knowledge (Mitchell et al. 2015), lexical information (Miller et al. 1990) and biological systems (Dumontier et al. 2014).

Knowledge graphs model facts as subject, predicate and object (SPO) triples, where subjects and objects are the knowledge entities and predicates are the knowledge relations. In this context, the subject entity is associated to the object entity with the predicate relation *e.g.* (*Aspirin*, *drug_target*, *COX1*). Fig. 6.1 shows an illustration of a schema of a knowledge graph that models complex associations between different types of biological entities such as drugs, proteins, antibodies, etc. It also models different types of relations between these entities, where these relation carry different association semantics.

In our study, we use $\mathcal{G}$ to denote a knowledge graph, $\mathcal{E}$ to denote entities and $\mathcal{R}$ to denote relations *i.e.* predicates. We also use $\mathcal{N}_e$ and $\mathcal{N}_r$ to denote the total count of both entities and relations in a knowledge graph respectively.

Popular Biological Sources. Online knowledge bases are a popular means for publishing large volumes of biological data (Zhu et al. 2018). In recent years, the number of these knowledge bases have grown, where they cover different types of data such as paper abstracts (Aronson et al. 2004), raw experimental data (Landrum et al. 2014), curated annotations (Consortium, Kanehisa et al., et. al. 2017, 2017, 2014), etc. Biological knowledge bases store data in different structured and unstructured (free text *e.g.* comments) forms. Although both data forms can be easily comprehended by humans, structured data is significantly easier for automated systems. In the following, we explore popular examples of these knowledge bases which offer structured data that can be easily and automatically consumed to generate knowledge graphs.

Table 6.1 summarises the specialisations and the different types of covered biological entities of a set of popular biological knowledge bases. The table also shows that most of the current knowledge bases are compiled around proteins (genes). However, it also shows their wide coverage of the different types of biological entities such as drugs, their indications, gene ontology annotations, etc.

Building Biological Knowledge Graphs. Knowledge graphs store information in a triplet form, where each triplet (*i.e.* triple) model a labelled association between two unique unambiguous entities. Data in biological knowledge bases, however, lacks these association labels.

Table 6.1 – A comparison between popular biological knowledge graph in terms of the coverage of different types of biological entities. The abbreviation S represent structured data, U represents unstructured data, DR represents drugs, GE represents proteins, GO represents gene ontology,PA represents pathways and *CH* denotes chemicals.

	Properties		Entity coverage								
Knowledge base	**Format**	**Speciality**	**Proteins**	**Drugs**	**Indications**	**Diseases**	**Gene Ontology**	**Expressions**	**Antibodies**	**Phenotypes**	**Pathways**
UNIPROT (Consortium 2017)	S/U	GE	✓	✓		✓	✓	✓	✓		✓
REACTOME (et. al. 2016)	S	PA	✓				✓				✓
KEGG (Kanehisa et al., Kanehisa et al. 2017, 2016)	S	PA	✓	✓		✓					✓
DrugBank (Wishart et al. 2008)	S/U	DR	✓	✓							✓
Gene Ontology (Consortium 2019)	S	GO	✓				✓				✓
CTD (Mattingly et al. 2003)	S/U	CH	✓	✓			✓			✓	✓
ChEMBL (Gaulton et al. 2017)	S/U	CH	✓	✓	✓	✓		✓			
SIDER (Kuhn et al. 2016)	S	DR		✓	✓						
HPA (Uhlén et al. 2015)	S/U	GE	✓				✓	✓	✓		
STRING (Szklarczyk et al. 2017)	S	GE	✓								
BIOGRID (Stark et al. 2007)	S	GE	✓								
InAct (et. al. 2014)	S	GE	✓								
InterPro (Mitchell & Attwood 2019)	S	GE	✓								
PharmaGKB (Whirl-Carrillo et al. 2012)	S	DR	✓	✓							
TTD (Chen et al. 2002)	S	DR	✓	✓							
Supertarget (Hecker & et. al. 2012)	S	DR	✓	✓							

Different knowledge bases also use different identifier systems for the same entity types which results in the ambiguity of entities of merged databases. Building biological knowledge graph process therefore mainly deals with these two issues.

In the association labelling routine, one can use different techniques to provide meaningful labels for links between different biological entities. This, however, is commonly achieved by using entity types of both subject and object entities to denote the relation labels as shown in Fig 6.1 (*e.g.* "Drug Side-effect" as a label for link between two entities that are known to be types of Drug and Side-effect, respectively).

The ambiguity issue, *i.e.* merging entities of different identifier systems, is commonly achieved using identifier mapping resource files. Different systems study entities on different speciality levels. As a result, the links between their different identifier systems is not always in a form of one-to-one relationships. In such cases, a decision is made to apply a specific filtering strategy based on either expert's opinion or problem-specific properties (for instance, deciding on an authoritative resource such as UniProt for protein entities and resolving all conflicts by sticking to that resource's naming scheme and conventions).

To complement the basic principles introduced in the previous paragraphs, we refer the reader

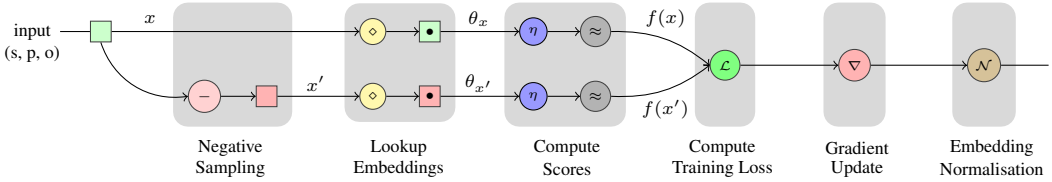

Figure 6.2 – An illustration of the training network of one training instance of a knowledge graph embedding model.

to the Bio2RDF initiative (Dumontier et al. 2014) that has extensively studied the general topic of building interlinked biological knowledge graphs (see also Bio2RDF scripts[1] for corresponding scripts and conversion convention details). General principles as well as an example of actual implementation of conversion from (relational) databases into RDF (*i.e.* knowledge graphs) are discussed in the study of Bizer et. al. (Bizer & Cyganiak 2006). Possible solutions to the problem of aligning and/or merging several such knowledge graphs are reviewed in the study of Amrouch et.al. (Amrouch & Mostefai 2012) that focuses on ontology matching. A more data-oriented approach is described for instance in LIMES (Ngomo & Auer 2011). All these approaches may provide a wealth of inspiration for building bespoke approaches to building knowledge graphs in specific biomedical use cases, should the information we provide in this section be insufficient.

6.2.2 Knowledge graph embeddings (KGE)

In this section, we discuss knowledge graph embedding models where we briefly explore their learning procedure. We then explore different embedding representation types and their potential uses and application.

The learning procedure. Multiple studies have explored knowledge graph embedding (KGE) models, their technical design, training objectives and predictive capabilities on general benchmarking settings (Nickel, Murphy, Tresp & Gabrilovich, Wang et al., Mohamed, Novácek, Vandenbussche & Muñoz 2016*a*, 2017, 2019). Therefore, in the following we only focus on providing a brief and concise description of how KGE models work.

KGE models operate by learning low-rank representations of knowledge graph entities and relations. The KGE learning step is a multi-phase procedure as shown in Fig. 6.2 which is executed iteratively on knowledge graph data. Initially, all entities and relations are assigned random embeddings (noise). They are then updated using a multiphase learning procedure.

KGE models consume knowledge graphs in the form of subject, predicate and object (spo) triplets. They first generate negative samples from the input true triplets using uniform random corruptions of the subjects and objects (Bordes et al. 2014). KGE models then lookup

[1] https://github.com/bio2rdf/bio2rdf-scripts/wiki

corresponding embedding of both the true and corrupted triplets. The embeddings are then processed using model-dependent scoring functions (cf. mechanism-of-action in Table 6.2) to generate scores for all the triplets. The training loss is then computed using model-dependent loss functions where the objective is to maximise the scores of true triplets and minimise the scores of corrupted triplets. This objective can be formulated as follows:

$$\forall_{t \in \mathbb{T}, t' \in \mathbb{T}'} f(\theta_t) > f(\theta_{t'}), \tag{6.1}$$

where $\mathbb{T}$ denotes the set of true triplets, $\mathbb{T}'$ denotes the set of corrupted triplets, f denotes the model-dependent scoring function and θ_t denotes the embeddings of the triplet t.

Traditionally, KGE models use a ranking loss, *e.g.* hinge loss or logistic loss, to model the objective training cost (Bordes et al., Yang et al., Trouillon et al. 2013, 2015*a*, 2016). This strategy allows KGE models to efficiently train their embeddings in linear time, $\mathcal{O}(d)$, where K denotes the size of the embedding vectors. On the other hand, some KGE models such as the ConvE (Dettmers et al. 2018) and the ComplEx-N3 (Lacroix et al. 2018) models adopt multi-class based strategies to model their training loss. These approaches have shown superior predictive accuracy compared to traditional ranking based loss strategies (Dettmers et al., Lacroix et al. 2018, 2018). However, they suffer from limited scalability as they operate on the full entity vocabulary.

The KGE models minimise their training loss using different variations of the gradient descent algorithm *e.g.* Adagrad, AMSGrad, etc. Finally, some KGE models normalise their embeddings as a regularisation strategy to enhance their generalisation. This strategy is often associated to models which adopt ranking based training loss strategies such as the TransE and DistMult models (Bordes et al., Yang et al. 2013, 2015*a*).

The learning multi-phase procedure is executed iteratively to update the model's embeddings until they reach an optimal state that satisfies the condition in Eq. 6.1. Table 6.2 also provides a summary of properties of popular KGE models, their mechanism of action *i.e.* scoring mechanism, output embeddings format, runtime complexity, release year and available code bases.

Knowledge graph embedding models ingest graph data in triplets form where they learn global graph low-rank latent features which preserve the graph's coherent structure. These features encode semantics such as node types and their neighbours by isolating nodes' embeddings on different embedding dimensions (Nickel, Murphy, Tresp & Gabrilovich 2016*b*). However, they have limited ability to encode indirect semantics such as logical rules and in-direct relations (Guo et al. 2016).

Embedding representation. Knowledge graph embeddings have different formats *e.g.* vectors,

Table 6.2 – A comparison between popular KGE models, their learning mechanism, published year and available code bases. Em. format column denotes the format of the model embeddings in the form $(g(d), h(d))$, where d denotes the embeddings size, $g(d)$ denotes the shape of the entities embeddings and $h(d)$ denotes the shape of the relations embeddings. n and m denote the number of entities and relations respectively in the space complexity column.

Model	Scoring mechanism	Format	Time	Space	Year	Repository (Python)
RESCAL	Tensor factorisation	(d, d^2)	$\mathcal{O}(d^2)$	$\mathcal{O}(nd + md^2)$	2011	mnick/rescal.py
TransE	Linear translation	(d, d)	$\mathcal{O}(d)$	$\mathcal{O}(nd + md)$	2014	ttrouill/complex
DistMult	Bilinear dot product	(d, d)	$\mathcal{O}(d)$	$\mathcal{O}(nd + md)$	2015	ttrouill/complex
HolE	FFT	(d, d)	$\mathcal{O}(d \log d)$	$\mathcal{O}(nd + md)$	2016	mnick/holographic
ComplEx	Complex product	$(2d, 2d)$	$\mathcal{O}(d)$	$\mathcal{O}(nd + md)$	2016	ttrouill/complex
ANALOGY	Analogical structure	(d, d)	$\mathcal{O}(d)$	$\mathcal{O}(nd + md)$	2017	quark0/ANALOGY
ConvE	Conv. filters	(d, d)	$\mathcal{O}(d)$	$\mathcal{O}(nd + md)$	2018	TimDettmers/ConvE
TriVec	Multi vectors	$(3d, 3d)$	$\mathcal{O}(d)$	$\mathcal{O}(nd + md)$	2019	samehkamaleldin/libkge

matrices, etc, which serve as numerical feature representations of their respective objects. These representations can be used in both general tasks such as clustering and similarity analysis, as well as in specific inference tasks such as predicting different association types. Similarly, in computational biology, they can be used to cluster biological entities such as protein, drugs, etc, as well as to learn specific biological associations such as drug targets, gene related diseases, etc. Embeddings of biological entities can also be used as representative features in traditional regression and classification models *e.g.* logistic regression or SVM classifiers.

Popular KGE models. Table 6.2 presents a comparison between a set of popular KGE models, their scoring mechanism, embeddings format, time complexity, space complexity, year of publication, and corresponding source code repository. These models use different approaches to learn their embeddings where they can be categorised into three categories: distance based models, factorisation based models and convolutional models. Distance based models such as the TransE model use linear translations to model their embeddings interactions using a linear time and space complexity procedure. Convolution based methods such as the ConvE use convolutional neural networks to model embedding interactions which also have a linear time and space complexity. Factorisation based models, on the other hand, use dot product based procedures to model embedding interactions, where they also have linear time and space complexity. However, tensor factorisation based models commonly use higher rank embeddings than convolution and distance based models (Trouillon et al., Mohamed & Novácek 2016, 2019).

In this chapter, we are focused on embedding methods which operate on multi-relational graphs as we mentioned in the introduction of the paper. The DeepWalk (Perozzi et al. 2014), Node2Vec (Grover & Leskovec 2016), etc are uni-relational graphs embedding methods, thus, they we do not include them in this chapter.

6.3 Examples of biological case studies

In the following, we present two example biological case studies that we use through this chapter to demonstrate the capabilities of KGE models. Firstly, we discuss the task of predicting drug target interactions where we model biological information as a knowledge graph. We then evaluate the predictive accuracy of KGE models and we compare them to other state-of-the-art approaches. Secondly, we discuss the task of predicting drug polypharmacy side-effects, where we model the investigated drug polypharmacy data as a 3D tensor.

6.3.1 Predicting drug target interactions

The study of drug targets has become very popular with the objective of explaining mechanisms of actions of current drugs and their possible unknown off-target activities. Knowing targets of potential clinical significance also plays a crucial role in the process of rational drug development. With such knowledge, one can design candidate compounds targeting specific proteins to achieve intended therapeutic effects. Large-scale and reliable prediction of drug-target interactions (DTIs) can substantially facilitate development of such new treatments. Various DTI prediction methods have been proposed to date. Examples include chemical genetic (Terstappen et al. 2007) and proteomic methods (Sleno & Emili 2008) such as affinity chromatography and expression cloning approaches. These, however, can only process a limited number of possible drugs and targets due to the dependency on laboratory experiments and available physical resources. Computational prediction approaches have therefore received a lot of attention lately as they can lead to much faster assessments of possible drug-target interactions (Yamanishi et al., Mei et al. 2008, 2012).

Data. We consider the DrugBank_FDA (Wishart et al. 2006) benchmarking dataset as an example to evaluate the predictive accuracy of KGE models and to compare them to other approaches. We also utilise the UNIPROT (Consortium 2017) database to provide richer information about both drugs and their protein targets in the input knowledge graph. The dataset contains 9881 known drug target interactions which involve 1482 drugs and 1408 protein targets.

Related work. The work of Yamanishi et. al.(Yamanishi et al. 2008) was one of the first approaches to predict drug targets computationally. Their approach utilised a statistical model that infers drug targets based on a bipartite graph of both chemical and genomic information. The BLM-NII (Mei et al. 2012) model was developed to improve the previous approach by using neighbour-based interaction-profile inference for both drugs and targets. More recently, Cheng et. al. (Cheng, Zhou, Li, Liu & Tang, Cheng, Liu, Jiang, Lu, Li, Liu, Zhou, Huang & Tang 2012, 2012) proposed a new way for predicting DTIs, where they have used a combination of drug similarity, target similarity and network-based inference. The COSINE (Rosdah et al. 2016) and NRLMF (Liu et al. 2015) models introduced the exclusive use of drug-drug and target-target similarity measures to infer possible drug targets. This has an advantage of being

able to compute predictions even for drugs and targets with limited information about their interaction data. However, these methods only utilised a single measure to model components similarity. Other approaches such as the KronRLS-MKL (Nascimento et al. 2016) model used a linear combination of multiple similarity measures to model the overall similarity between drugs and targets. Non-linear combinations were also explored in an early study (Mei et al. 2012) and shown to provide better predictions. Recently, further predictive models were developed to utilise matrix factorisation (Hao et al. 2017) and biological graph path features (Olayan et al. 2017) to enable more accurate drug target prediction.

6.3.2 Predicting polypharmacy side-effects

Polypharmacy side-effects are a specific case of adverse drug reactions that can cause significant clinical problems and represent a major challenge for public health and pharmaceutical industry (Bowes et al. 2012). Pharmacology profiling leads to identification of both intended (target) and unintended (off-target) drug-induced effects, i.e. biological system perturbations. While most of these effects are discovered during pre-clinical and clinical trials before a drug release on the market, some potentially serious adverse effects only become known when the drug is in use already.

When more drugs are used jointly (i.e. polypharmacy), the risk of adverse effects rises rather rapidly (Kantor et al., Tatonetti et al. 2015, 2012). Therefore, reliable automated predictions of such risks are highly desirable to mitigate their impact on patients.

Data. In this case study, we consider the dataset compiled by Zitnik et al. (Zitnik et al. 2018) as an example benchmark. The dataset includes information about multiple polypharmacy drug side-effects [2]. The dataset also contains facts about single drug side-effects, protein-protein interactions and protein-drug targets. The drug side-effects represented in the dataset are collected from the SIDER (Side Effect Resource) database (Kuhn et al. 2016) and the OFFSIDES and TWOSIDES databases (Tatonetti et al. 2012). These side-effects are categorised into two groups: mono-drug and polypharmacy drug-drug interaction side-effects.

In our study, we only consider the polypharmacy side-effects and we filter out both the mono-side effects and drug targets data.

Related work. The research into predictive approaches for learning drug polypharmacy side effects is in its early stages (Zitnik et al. 2018). The decagon model (Zitnik et al. 2018) is one of the first introduced methods for predicting polypharmacy side-effects which models the polypharmacy side-effects data as a knowledge graph. It then solves the problem as a link prediction problem using a generative convolution based strategy. Despite its effectiveness, this approach still suffers from a high rate of false positives. Furthermore, other approaches considered using a multi-source embedding model (García-Durán & Niepert 2018) to learn

[2] http://snap.stanford.edu/decagon/

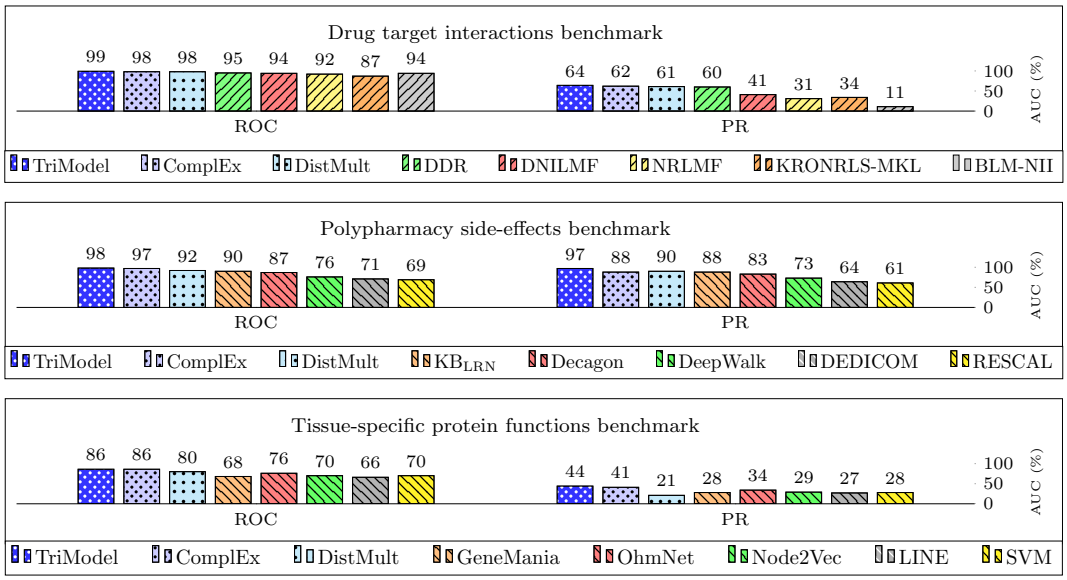

Figure 6.3 – A summary of results of an evaluation of the predictive accuracy of knowledge graph embedding models compared to other models on two biological inference tasks: predicting drug targets and predicting polypharmacy side-effects. The reported results represent the score percentage of the area under the ROC and precision recall curves for the left and right side bars respectively.

representations of drugs and polypharmacy side-effects. These approaches achieved similar performance to the Decagon model with a more scalable training procedure (García-Durán & Niepert 2018).

6.3.3 Predicting tissue-specific protein functions

Proteins are usually expressed in specific tissues within the body where their precise interactions and biological functions are frequently dependant on their tissue context (Fagerberg et al., Greene et al. 2014, 2015). The disorder of these interactions and functions results in diseases (D D'Agati, Cai & Petrov 2008, 2010). Thus, the deep understanding of tissue-specific protein activities is essential to elucidate the causes of diseases and their possible therapeutic treatments.

Data. We consider the tissue-specific dataset compiled by Zitnik et. al (Zitnik & Leskovec 2017) to study tissue-specific protein functions. The dataset contain protein-protein interactions and protein functions of 144 tissue types[3].

Related work. Recently, Zitnik et. al. have developed the state-of-the-art model, the OhmNet model (Zitnik & Leskovec 2017), a hierarchy-aware unsupervised learning method for multi-layer networks. It models each tissue information as a separate network, and learns

[3] http://snap.stanford.edu/ohmnet/

efficient representations for proteins and functions by generating their embeddings using the tissue-specific protein-protein interactome and protein functions. They have also examined other different approaches such as the LINE model (Tang et al. 2015) which uses a composite learning technique where it learns half of the embeddings' dimensions from the direct neighbour nodes, and the other half from the second hop connected neighbours. The GeneMania model (Warde-Farley et al. 2010) is another model which has suggested a propagation based approach for predicting tissue-specific protein functions. In this method, the tissue-specific networks are firstly combined into one weighted network, and they are then propagated to allow predicting other unknown protein functions.

6.4 Capabilities of KGE models

KGE models can be used in different supervised and unsupervised applications where they provide efficient representations of biological concepts. They can be used in applications such as learning biological associations, concepts similarity and clustering biological entities. In this section, we discuss these applications in different computational biology tasks. We provide a set of example uses cases where we present the data integrated in each example, how the KGE models were utilised and we report the predictive accuracy of the KGE models and we compare it to other approaches when possible.

6.4.1 Learning biological associations

KGE models can process data in the form of a knowledge graph. They then try to learn low-rank representations of entities and relations in the graph which preserve its coherent structure. They can also process data in a three dimensional (3D) tensor form where they learn low-rank representations for the tensor entities that preserve true entity combination instances in the tensor.

In the following, we provide two examples for learning biological associations on a knowledge graph and a 3D tensor in a biological application. First, we discuss the task of predicting drug target interactions where we model biological information as a knowledge graph. We then evaluate the predictive accuracies of KGE models and we compare them to other state-of-the-art approaches. Secondly, we discuss the task of predicting drug polypharmacy side-effects, where we model the related data as a 3D tensor. We then apply KGE models to perform tensor factorisation and we evaluate their predictive accuracy in learning new polypharmacy side-effects compared to other state-of-the-art approaches.

- **Drug target prediction benchmark** We present a comparison between state-of-the-art drug target predictors and knowledge graph embedding models in predicting drug target interactions. The KGE models in this context utilise the fact that the current drug target knowledge bases like DrugBank (Wishart et al. 2006) and KEGG (Kanehisa et al. 2017) are largely structured as networks representing information about drugs and their

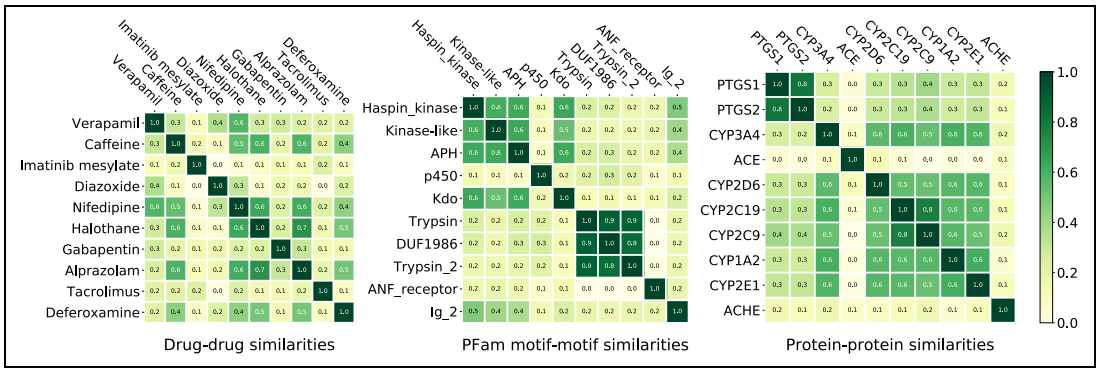

Figure 6.4 – Three similarity matrices that denote the Drug-drug similarities, motif-motif similarities and protein-protein similarities. The similarity values are generated by computing the cosine similarity between the embeddings of the pairs of compared entities. All the embeddings used to generated this figure are computed on the DrugBank_FDA datasets with the proteins associated to their PFam (Bateman et al. 2000) motifs and protein families.

relationship with target proteins (or their genes), action pathways, and targeted diseases. Such data can naturally be interpreted as a knowledge graph. The task of finding new associations between drugs and their targets can then be formulated as a link prediction problem on a biological knowledge graph.

We use the standard evaluation protocol for the drug target interaction task (Olayan et al. 2017) on the DrugBank_FDA dataset that we introduced in Sec. 6.3.1. We use a 5-fold cross validation evaluation on the drug target interactions where they are divided into splits with uniform random sampled negative instances with a 1:10 positive to negative ratio.

Fig. 6.3 presents the outcome results of the KGE models (DistMult, ComplEx and TriVec) compared to other approaches (DDR (Olayan et al. 2017), DNILMF (Hao et al. 2017), NRLMF Hao et al. (2017), NRLMF (Liu et al. 2015), KRONRLS-MKL Nascimento et al. (2016), COSINE (Lim et al. 2016), and BLM-NII (Mei et al. 2012)) on the DrugBank_FDA dataset. The figure shows that the KGE models outperform all other approaches in terms of both the area under the ROC and precision recall curves.

- **Polypharmacy side-effects prediction benchmark** In Sec. 6.3.2 we discussed the problem of predicting polypharmacy side-effects, the currently available data and related works. In the following, we present an evaluation benchmark for present polypharmacy side-effects where we compare the KGE models with current state-of-the-art approaches. We first split the data into two sets, train and test splits, where the two splits represent 90% and 10% of the data respectively. We then generate random negative polypharmacy side-effects by randomly generating combinations of drugs for each polypharmacy side effect where the ratio between negative and positive instances is 1:1. We only consider drug combinations that did not appear in the both training and test splits to enhance the quality of sampled negatives and decrease the ratio of false negatives.

We use the holdout test defined by Zitnik et. al. (Zitnik et al. 2018) where we train the predictive models on the training data and test their accuracy on the testing data split. We also run a 5-runs averaged 5-fold cross validation evaluation to ensure the consistency of the model reported results over the different folds, however, we only report the holdout test results which are comparable with state-of-the-art methods. Our k-fold cross validation experiments confirm that the model results are similar or insignificantly different across different random testing splits.

We use the area under the ROC and precision recall metrics to assess the quality of the predicted scores. Fig. 6.3 presents the results of our evaluation where we compare KGE models such as the DistMult, ComplEx and TriVec models to the current popular approaches (Decagon (Zitnik et al. 2018), KB_LRN (Malone et al. 2018), RESCAL (Nickel et al. 2011), DEDICOM (Papalexakis et al. 2016), DeepWalk (Perozzi et al. 2014)). The results show that KGE models outperform other state-of-the-art approaches in terms of both the area under the ROC and precision recall curves.

- **Tissue-specific protein function prediction benchmark** In Sec. 6.3.3 we have presented the problem of tissue-specific protein function prediction benchmark where we have discussed current predictive models and established benchmarking datasets. In the following, we present an evaluation benchmark between a set of traditional approaches such as the OhmNet (Zitnik & Leskovec 2017), LINE (Tang et al. 2015), GeneMania (Warde-Farley et al. 2010) and SVM (Zitnik & Leskovec 2017) models and other KGE models. We use the dataset generated by Zitnik et. al. (Zitnik & Leskovec 2017) which provides training and testing data with both positive and negative instances where the negative to positive ratio is 1 to 10.

  We conduct a holdout test using the provided training and testing dataset where we train our models on the training split and evaluate them on the testing using the area under the ROC and precision recall curves. Fig. 6.3 presents the outcome of our experiments where it shows that KGE models such as the TriVec and ComplEx models achieve the best results in terms of both the area under the ROC and precision recall curves. Similar to the previous experiments, we also ran a 5-runs 5-fold cross validation test to ensure the consistency of our results and the results of our experiments confirm the results reported in the holdout test. However, we only report the holdout test results to be able to compare to other approaches.

In all of our experiments, we learn the best hyperparameters using a grid-search on the validation data split. We have found the the embedding size is the most sensitive hyperparameters where it correlates with the graph size. The regulation weight and and embedding dropout also are important hyperparameters which affect the generality of the models from the validation to the testing split.

Example source code scripts and datasets of the experiments which we executed in this chapter

are available online[4].

6.4.2 Learning similarities between biological entities

The KGE models enable a new type of similarity which can be measured between any two biological entities using the similarity between their vector representation. The similarity between vectors can be computed using different techniques such as the *cosine* and *p-norm* similarities. Since the KGE representation is trained to preserve the knowledge graph structure, the similarity between two KGE representations reflects their similarity in the original knowledge. Therefore, the similarities between vector representations of KGE models, which are trained on a biological knowledge graphs, represent the similarities between corresponding entities in the original knowledge graph.

In the following, we explore a set of examples for using KGE similarities on biological knowledge graphs. We have used the drug-target knowledge graph created for the drug target prediction task to learn embeddings of drugs, their target proteins and the entities of the motifs of these proteins according to the PFam database (Bateman et al. 2000). We have then computed the similarities between embeddings of entities of the same type such as drugs, proteins and motifs as shown in Fig. 6.4. All the similarity scores in the illustration are computed using cosine similarity between the embeddings of the corresponding entity pair. The results show that the similarity scores are distributed from 0.0 to 1.0, where the 0.0 represents the least similar pairs and the 1.0 scores represent the similarity between the entity and itself. We then assess the validity of resulting scores by investigating the similarity of attributes of a set of the examined concepts with highest and lowest scores.

- **Drug-drug embedding similarity** The left similarity matrix in Fig. 6.4 illustrates the drug-drug similarity scores between the set of the most frequent drugs in the DrugBank_FDA dataset. The scores are computed on the embeddings of drugs learnt in the drug target interaction training pipeline. The figure shows that the majority of drug pairs have a low similarity (0.0 ~ 0.2). For example, the similarity score between the drug pairs (*Diazoxide*, *Caffeine*) and (*Tacromlimus*, *Diazoxide*) are zero. We asses these results by assessing the commonalities between the investigated drugs in terms of indications, pharmacodynamics, mechanism of action, targets, enzymes, carriers and transporters. The *Caffeine* and *Diazoxide* in this context have no commonalities except for that they are both diuretics (Lipschitz et al., Pohl et al. 1943, 1972). On the other hand, Halothane and Alprazolam does not share any of the investigated commonalities.

  The results also shows a few drug-drug similarities with relatively higher scores (0.6 ~ 0.7). For example, the similarity scores of the drug pairs (*Alprazolam*, *Halothane*), (*Alprazolam*, *Caffeine*) and (*Halothane*, *Caffeine*) are 0.7, 0.6 and 0.6 respectively. These finding can be supported by the fact that the two drug pairs share common attributes

[4] https://github.com/samehkamaleldin/bio-kge-apps

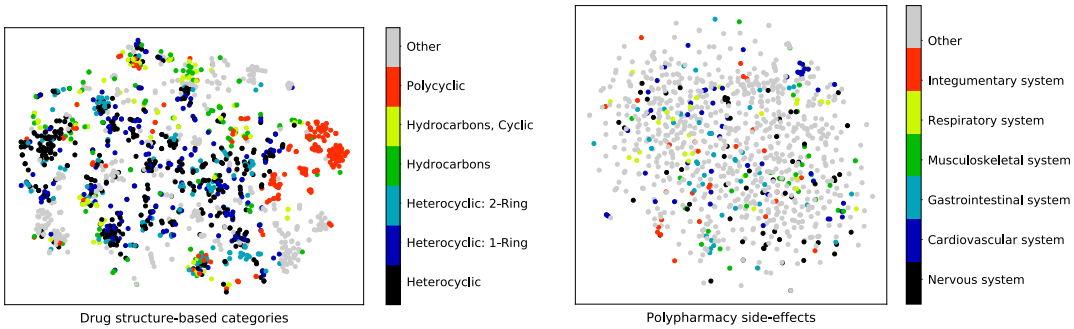

Figure 6.5 – Three similarity matrices that denotes the Drug-drug similarities, motif-motif similarities and protein-protein similarities. The similarity values are generated by computing the cosine similarity between the embeddings of the pairs of compared entities. All the embeddings used to generated this figure are computed on the DrugBank_FDA datasets with the proteins associated to their PFam (Bateman et al. 2000) motifs and protein families.

in terms of their targets, enzymes and carriers. For example, both *Alprazolam* and *Halothane* act on sedating individuals and they target the GABRA1 protein (Verster & Volkerts, Overington et al. 2004, 2006). They are also broken by CYP3A4 and CYP2C9 enzymes and carried by albumin (Minoda & Kharasch 2001). Similarly, the (*Alprazolam*, *Caffeine*) and (*Halothane*, *Caffeine*) pairs have common associated enzymes.

- **Motif-motif embedding similarity** The middle similarity matrix in Fig. 6.4 illustrates the motif-motif similarity scores between the set of the most frequent PFam motifs associated with protein targets from the drug target interaction benchmark. The lowest motif-motif KGE based similarity scores correspond to the pairs (*ANF_receptor*, Trypsin) , (*ANF_receptor*, *DUF1986*) and (*ANF_receptor*, *Trypsin_2*).

  On the other hand, The highest similarity scores (0.8, 0.9 and 0.9) exist between the pairs (*Trypsin*, *DUF1986*), (*Trypsin_2*, *DUF1986*) and (*Trypsin*, *Trypsin_2*) respectively.

  We assess the aforementioned findings by investigating the nature and activities of each of the discussed motifs. For example, *Trypsin* is a serine protease that breaks down proteins and cleaves peptide chains while *Trypsin_2* is an isozyme of *Trypsin* which has a different amino acid sequence but catalyses the same chemical reaction as *Trypsin* (Rungruangsak-Torrissen et al. 1999).

  Moreover, the DUF1986 is a domain that is found in both of these motifs which supports the high similarity scores. On the other hand, the *ANF receptor* is an atrial natriuretic factor receptor that binds to the receptor and causes the receptor to convert *GTP* to *cGMP*, and it plays a completely different role to trypsin, which supports its reported low similarity scores with trypsin.

- **Protein-protein embedding similarity** The right similarity matrix in Fig. 6.4 illustrates the protein-protein similarity scores between the set of the most frequent protein targets from the drug target interaction benchmark. The highest scored protein-protein pairs are (*PTGS1*, *PTGS2*) and (*CYP2C19*, *CYP2C9*) with the scores 0.8 and 0.8 respectively.

This can be supported by the fact that the proteins CYP2C9, CYP1A2 and CYP2E1 belong to the same family of enzymes and thus they have similar roles.

On the other hand, The *ACE* protein have the lowest similarity scores with the *CYP2C9*, *CYP1A2* and *CYP2E1* proteins with 0.0 similarity score. This can be supported by the fact that ACE is a a hydrolase enzyme which is completely different from *CYP2C9*, *CYP1A2* and *CYP2E1* which are Oxidoreductases enzymes.

6.4.3 Clustering biological entities

In the following, we demonstrate the possible uses of embeddings based clustering in different biological tasks. We explore two cases where we use the embeddings of KGE models to generate clusters of biological entities such as drugs and polypharmacy side-effects. We use visual clustering as an example to demonstrate cluster separation on a 2D space. However, in real scenarios, clustering algorithms utilise the full dimensionality of embedding vectors to build richer semantics of outcome clusters. Fig 6.5 shows two scatter plots of the embeddings of drugs from the DrugBank_FDA dataset and the polypharmacy side-effects reduced to a 2D space. We reduced the original embeddings using the T-SNE dimensionality reduction module (van der Maaten 2014) with the cosine distance configuration to reduce the embedding vectors to a 2D space.

The following examples examines two cases that differs in terms of the quality of generated clusters where we examine both drugs and polypharmacy side-effects according to different properties. In the first example (drug clustering), the generated embeddings is able to provide efficient clustering. On the other hand, in the second example, the polypharmacy side-effects, the learnt embeddings could not be separated into visible clusters according to the investigated property.

- **Clustering drugs** The left plot in Fig. 6.5 shows a scatter plot of the reduced embedding vectors of drugs coloured according to their chemical structure properties. The drugs are annotated with seven different chemical structure annotations: *Polycyclic*, *Hydrocarbons Cyclic*, *Hydrocarbons*, *Heterocyclic*, *Heterocyclic 1-Ring*, *Heterocyclic 2-Ring* and other chemicals. These annotations represent the six most frequent drug chemical structure category annotation extracted from the DrugBank database.

  We can see in the plot that the *Polycyclic* chemicals are located within a distinguishable cluster in the right side of the plot. The plot also shows that other types of *Hydrocarbons* and *Heterocyclic* chemicals form different micro-clusters in different locations in the plot.

  These different clusters can be used to represent a form of similarity between the different drugs. It can also be used to examine the relation between the embeddings as a representation with the original attributes of the examined drugs.

- **Clustering polypharmacy side-effects** The right plot in Fig. 6.5 shows a scatter plot of

the reduced embedding vectors of polypharmacy side-effects. The plot polypharmacy side-effect points are coloured according to the human body systems they affect. The plot includes a set of six categories of polypharmacy side-effects that represent six different human body systems *e.g.* nervous system.

Unlike the drug clusters illustrated in the left plot, the polypharmacy side-effects system-based categorisation does not yield obvious clusters. They, however, form tiny and scattered groups across the plot. This shows that the KGE models are unable to learn representations that can easily separate polypharmacy side-effects according to their associated body system.

6.5 Example application: predicting polypharmacy side-effects

We use the TriVec model as a tensor factorisation based embedding model that solves the problem of link prediction as a 3D tensor completion, where the tensor dimensions represent entities and relations. In the task of predicting polypharmacy side-effects, the drug combinations are modelled as the subjects and objects of triples while the corresponding polypharmacy side-effects are modelled as relations. In the training process, the model processes the different types of assertions such as protein-protein interactions, drug-protein interactions, drug-drug interactions, single drug side effects and polypharmacy side-effects. This allows the model to learn efficient embeddings for the components corresponding to the different entities and relations in the knowledge graph. In the prediction phase, the TriVec model then learns the probability of polypharmacy side-effects associations to drug combinations by completing a 3D tensor of drugs and polypharmacy side-effects.

6.5.1 Experiments

In this section, we discuss the setup of our experiments, the evaluation protocol and the detail of the frameworks and technologies used to implement our experiments.

Benchmarking dataset

In this chapter, we build a benchmarking dataset to evaluate our model following the evaluation protocol proposed by Zitnik et. al. (Zitnik et al. 2018). We first divide polypharmacy side-effects assertions into groups according to the side-effect type. We then divide the assertions of each group into three groups: training, validation and testing with 80%, 10% and 10% percentages of the data respectively. This process is applied to the polypharmacy side-effect groups with 500 or more assertions to assure a minimum of 50 validation and testing instances for each side-effect. We then add other types of assertions such as drug-protein interactions, protein-protein interactions and single drug side-effects into the training data[5].

[5]The preprocessed knowledge graph with the train, validation and testing splits is available to download at: https://figshare.com/articles/polypharmacy-dataset/7958747

Table 6.3 – Summary of statistics of entities, relations and triples in the different splits of the benchmarking dataset.

Dataset	Entities	Relations	Triples	P. Side-effects
Training data	32K	967	4.7M	3.7M
Validation data	643	963	459K	459K
Testing data	643	963	459K	459K
All	32K	967	5.6M	4.6M

For each validation and testing splits we generate negative samples by using random unobserved drug combinations as negatives. This process is executed independently for each polypharmacy side-effect, where the positive to negative ratio is 1:1. (Table 6.3) shows a summary of the statistics of different components in the data generated for the set of all polypharmacy side-effects.

Experimental setup

We use the supporting knowledge graph to perform a grid search to learn the model's best hyperparameters. In all of our experiments we initialise our model embeddings using the Glorot uniform random generator (Glorot & Bengio 2010) and we optimise the training loss using the Adagrad optimiser, where the learning rate (lr) $\in \{0.1, 0.01, 0.001\}$, embeddings size $(K) \in \{50, 100, 150, 200\}$ and batch size $(b) \in \{512, 1024, 4000, 6000\}$. The rest of the grid search hyper parameters are defined as follows: the regularisation weight $(\lambda) \in \{0.1, 0.3, 0.35, 0.01, 0.03, 0.035\}$ and dropout $(d) \in \{0.0, 0.1, 0.2, 0.01, 0.02\}$. The number of training epochs is fixed to 1000. We found that the best hyper parameter for our models are $\{\text{lr} = 0.1, k = 100, b = 6000, \lambda = 0.03, d = 0.2\}$.

Evaluation protocol

In our experiments, we follow the evaluation protocol introduced by Zitnik et. al (Zitnik et al. 2018), and we evaluate the TriVec model on the testing data split using three evaluation metrics: area under the roc curve, area under the precision recall curve and average precision at 50 positives. The testing data contains both positive and negative data samples with a ratio of 1:1. All reported evaluation scores represent the average of its corresponding scores for all the investigated polypharmacy side-effects.

Implementation

We use Tensorflow framework (GPU) along with Python 3.5 to perform our experiments. All experiments were executed on a Linux machine with processor Intel(R) Core(TM) i70.4790K CPU @ 4.00GHz, 32 GB RAM, and an nVidia Titan Xp GPU.

Table 6.4 – Summary of the results of our experiments. † represents the results of the state-of-the-art models that are obtained from the study of Zitnik et. al. (Zitnik et al. 2018). * represents the results of the state-of-the-art models that are obtained from the study of Malone et. al. (Malone et al. 2018).

Model	AUC-ROC	AUC-PR	AP@50
RESCAL † (Nickel et al. 2011)	0.693	0.613	0.476
DEDICOM † (Perros et al. 2017)	0.705	0.637	0.567
DeepWalk † (Perozzi et al. 2014)	0.761	0.737	0.658
Concatnated Features † (Zitnik et al. 2018)	0.793	0.764	0.712
Decagon † (Zitnik et al. 2018)	0.872	0.832	0.803
TransE (Bordes et al. 2013)	0.949	0.934	0.962
DistMult * (Yang et al. 2015*a*)	0.923	0.898	0.899
KB_{LRN} * (Malone et al. 2018)	0.899	0.878	0.857
ComplEx (Trouillon et al. 2016)	0.965	0.944	0.952
TriVec (This study)	**0.975**	**0.966**	**0.983**

6.5.2 Results

In this section we discuss the outcomes of our experiments and we compare the predictive accuracy of our proposed approach with other state-of-the-art approaches.

(Table 6.4) shows the results of our experiments, where the models are compared in terms of the area under the ROC and precision recall curves and the average precision at 50. The results show that our model, the TriVec model, significantly outperforms other models with 12%, 16% and 22% margins in terms of the area under the roc, precision recall curves and with an average precision at 50 compared to the decagon model. The results also show that our methods outperforms other knowledge graph embedding models such as the RESCAL, TransE, DistMult, ComplEx and KB_{LRN} models on all metrics.

Additionally, the results show that models such as the TransE, TriVec and ComplEx models achieve significantly high scores (above 90%) in terms of the area under both the ROC and precision recall curves. We suggest that this is due to the easy nature of the evaluation protocol that uses a 1:1 negative to positive ratio in the testing set. Therefore, we suggest that future works should adapt high negative to positive ratios such as 1:10 or 1:50.

6.5.3 Discussion

In this section, we discuss the outcome of our experiments, the limitations of our approach and our future research direction.

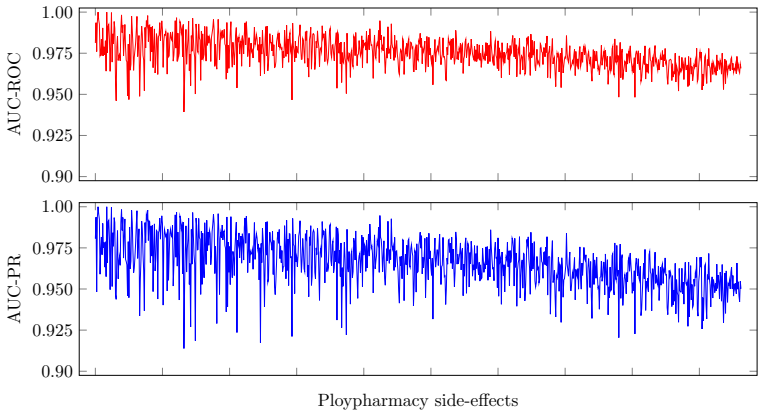

Figure 6.6 – Plot of the area under the ROC and precision recall scores of all the polypharmacy side-effect groups in the benchmarking dataset. The X-axis represents polypharmacy side-effects, where they are sorted in an ascending order from left to right according to their count in the whole benchmarking dataset.

Side-effect Specific Analysis

The outcomes of our experiments show that the model yields consistently high scores on all the investigated side-effects with one or few outliers. The figure also shows an observed low negative correlation between the model's metric score of a side-effect and the count of its associated drug combination. We suggest that this is related to difficulty of the evaluation, where side-effects with a high number of positive drug combinations also have a higher number of negative samples. This makes the prediction of true positive instances of these side-effects relatively harder compared to other side-effects with a small number of negative examples.

In Table 6.5 we provide a summary of the results of our model on a set of 20 polypharmacy side-effects where our model achieved its highest and lowest predictive accuracy in terms of the area under the precision recall curve. The results gives a description of how the model accuracy scores are distributed on its highest and lowest scored side-effects with relation to their associated drug combinations. The results also confirm the findings in Fig. 6.6 where the top-10 lowest scored side-effects has an average of 4050 associated drug combinations compared to the top-10 highest scored side-effects that have an average of 707 associated drug combinations.

6.6 Practical considerations for KGE models

In this section, we discuss different practical considerations related to the use of KGE models. We discuss their scalability on different experimental configurations, and we explore their different training and implementation strategies.

Table 6.5 – Summary of results for the set of polypharmacy side-effects where the TriVec model achieved its highest and lowest predictive accuracy in terms of the area under the precision recall curve.

#	SE. Name	SE. Code	Count	AUC-ROC	AUC-PR
		Lowest accuracies			
1	Ventricular septal defect	C0018818	842	0.939	0.914
2	Malabsorption	C0024523	1258	0.961	0.917
3	Congenital heart disease	C0152021	912	0.950	0.919
4	Icterus	C0022346	7944	0.948	0.920
5	Stridor	C0038450	1552	0.947	0.921
6	Allergic vasculitis	C0151436	2052	0.951	0.922
7	Cardiovascular collapse	C0036974	8681	0.948	0.923
8	Esophageal cancer	C0152018	1123	0.954	0.924
9	Bleeding	C0019080	14143	0.953	0.926
10	Strabismus	C0038379	1995	0.957	0.926
		Highest accuracies			
1	Ingrowing nail	C0027343	1012	0.996	0.996
2	Acute psychosis	C0281774	890	0.996	0.997
3	Hair disease	C0018500	810	0.997	0.997
4	Temporal arteritis	C0039483	654	0.997	0.998
5	Substance abuse	C0740858	644	0.998	0.998
6	Ganglion	C1258666	694	0.998	0.998
7	Dyspareunia	C1384606	598	0.998	0.998
8	Dyshidrosis	C0032633	553	1.000	1.000
9	Coccydynia	C0009193	508	1.000	1.000
10	Splenectomy	C0037995	530	1.000	1.000

6.6.1 Scalability

Not only KGE models outperform other approaches in biological knowledge graphs completion tasks, but they also have better scalability compared to usual graph exploratory approaches. Often, complex biological systems are modelled as graphs where exploratory graph analytics methods are applied to perform different predictive tasks (Janjic & Przulj, Muñoz et al., Olayan et al. 2012, 2017, 2017). These models however suffer from limited scalability as they depend on graph traversal techniques that require complex training and predictions times (Cheung, Fraigniaud et al. 1983, 2006). On the other hand, KGE models operate using linear time and space complexity (Trouillon et al., Mohamed & Novácek 2016, 2019).

On the other hand, explanatory graph models use graph path searches which require higher time and space complexity (Toutanova & Chen 2015). For example, the DDR model (Olayan et al. 2017) is an exploratory graph drug-target predictor which uses graph random walks as features. A recent study (Mohamed, Nováček & Nounu 2019) has shown that knowledge graph

embedding models can outperform such models with higher scalability and better predictive accuracy. This is due to their linear time and space complexity procedures (Trouillon et al. 2016) compared to other exploratory models which use polynomial and exponential time and space procedures (Nickel, Murphy, Tresp & Gabrilovich, Mohamed et al. 2016*b*, 2017).

Chapter 4 provides an extensive discussion of the different components of the KGE training pipeline and how they affect the scalability and accuracy of the models.

6.6.2 Implementation and training strategies

Different implementations of KGE models are available online in different repositories as shown in Table. 6.2. The high scalability of KGE models allows them to be ported to both CPUs and GPUs where they can benefit from the high performance capabilities of GPU cores. They can also be implemented to operate in a multi-machine design, where they perform embedding training in a distributed fashion (Lerer et al. 2019). This configuration is better suited for processing knowledge graph of massive volumes that is hard to fit into one machine.

In this chapter, all our experiments are implemented in Python 3.5 using the Tensorflow library where we train our models on a single GPU card on one machine. We run our experiments on a Linux machine with an Intel(R) Core(TM) i7 processor, 32 GB RAM, and an nVidia Titan Xp GPU.

6.7 Opportunities and challenges

In this section, we discuss the challenges and opportunities related to the general and biological applications of KGE models. We begin by discussing the scope of input data for these models. We then discuss possible applications of KGE models in the biological domain. We conclude by discussing the limited interpretability of KGE models and other general limitations related to their biological applications.

6.7.1 Potential applications

KGE models can build efficient representations of biological data which is modelled as 3D tensors or knowledge graphs. This includes multiple types of biological data such as protein interactome and drug target interactions. In the following, we discuss examples of biological tasks and applications that can be performed using KGE models.

- **Modelling proteomics data.** KGE models can be used to model the different types of protein–protein interactions such as binding, phosphorylation, etc. This can be achieved by modelling these interactions as a knowledge graphs and applying the KGE models to learn the embeddings of the different proteins and interaction types. They can also be used to model the tissue context of interactions where different body tissues

have different expression profiles of proteins. Therefore, these differences in expression affect the the proteins' interaction network.

The biological activities of proteins also differ depending on their tissue context (Zitnik & Leskovec 2017). This type of information can easily be modelled using tensors where KGE models can be used to analyse the different functions of proteins depending on their tissue context.

- **Modelling genomics data.** Genomics data has been widely used to predict multiple gene associated biological entities such as gene–disease and gene–function associations (Bamshad et al., Zeng et al. 2011, 2017). These approaches model the gene association in different ways including tensors and graph based representations (Bauer-Mehren et al. 2011). KGE models can be easily utilised to process such data and provide efficient representations of genes and their associated biological objects. They can be further used to analyse and predict new disease–gene and gene–function associations.

- **Modelling pharmacological systems.** Information on pharmaceutical chemical substances is becoming widely available on different knowledge bases (Wishart et al., Gaulton et al. 2006, 2017). This information includes the drug–drug and drug–protein interactome. In this context, KGE models can be a natural fit, where they can be used to model and extend the current pharmacological knowledge. They can also be used to model and predict both traditional and polypharmacy side-effects of drugs as shown in recent works (Muñoz et al., Zitnik et al. 2016, 2018).

More details and discussion of the possible uses of KGE models and other general network embedding methods can be found in the study of Su et. al. (Su et al. 2018) which discusses further potential uses of these methods in the biological domain.

6.7.2 Limitations of the KGE models

In the following, we discuss the limitations of the KGE models in both general and biological applications.

- **Lack of interpretability** In knowledge graph embedding models, the learning objective is to model nodes and edges of the graph using low-rank vector embeddings that preserve the graph's coherent structure. The embedding learning procedure operates mainly by transforming noise vectors to useful embeddings using gradient decent optimisation on a specific objective loss. Despite the high accuracy and scalability of this procedure, these models work as a black box and they are hard to interpret. Some approaches have suggested enhancing the interpretability of KGE models by using constraining training with a set of predefined rules such as type constraints (Krompass et al. 2015), basic relation axioms (Minervini et al. 2017*b*), etc. These approaches thus

enforce the KGE models to learn embeddings that can be partially interpretable by their employed constraints.

- **Data quality** KGE models generate vector representations of biological entities according to their prior knowledge. Therefore, the quality of this knowledge affects the quality of the generated embeddings. For example, there is a high variance in the available prior knowledge on proteins where well studied proteins have significantly higher coverage in most databases (The Uniprot Consortium 2015). This has a significant impact on quality of the less represented proteins as KGE models will be biased towards more studied proteins (*i.e.* highly covered proteins).

  In recent years, multiple works have explored the quality of currently available knowledge graphs (Färber et al. 2017) and the effect of low quality graphs on embedding models (Pujara et al. 2017). These works have shown that the accuracy KGE predictions degrade as sparsity and unreliability increase (Pujara et al. 2017).

  This issue can be addressed by extending the available knowledge graph facts through merging knowledge bases of similar content. For example, drug target prediction using KGE models can be enhanced by extending the knowledge of protein–drug interactions by extra information such as protein-protein interactions and drug properties (Mohamed, Nováček & Nounu 2019).

- **Hyper-parameter sensitivity** The outcome predictive accuracy of KGE embeddings is sensitive to their hyper-parameters (Kadlec et al. 2017). Therefore, minor changes in these parameters can have significant effects on the outcome predictive accuracy of KGE models. The process of finding the optimal parameters of KGE models is traditionally achieved through an exhausting brute-force parameter search. As a result, their training may require rather time-consuming grid search procedure to find the right parameters for each new dataset.

  In this regard, new strategies for hyper parameter tuning such as differential evolution (Fu et al. 2016), random searches (Solis & Wets 1981) and Bayesian hyper parameter optimisation (Snoek et al. 2012). These strategies can yield a more informed parameter search results with less running time.

- **Reflecting complex semantics of biological data in models based on knowledge graphs** Knowledge graph embedding methods are powerful in encoding direct links between entities, however, they have limited ability in encoding simple indirect semantics such as types at different abstraction levels (*i.e.* taxonomies). For example, a KGE model can be very useful in encoding networks of interconnecting proteins which are modelled using direct relations. However, it has limited ability in encoding compound, multi-level relationships such as protein involvement in diseases due to their involvement in pathways that cause this disease. Such compound relationships that could be used for modelling complex biological knowledge are notoriously hard to reflect in KGE models (Weber et al. 2019). However, the KGE models do have some limited ability to encode for instance type constrains (Minervini et al. 2017*a*), basic triangular rules (Weber et al.

2019) or cardinality constraints (Muñoz et al. 2019). This could be used for modelling complex semantic features reflecting biological knowledge in future works. One has to bear in mind, though, that the designs of these semantics-enhanced KGE models typically depends on an extra computational routines to regularise the learning process which affects their scalability.

In their study, Su et. al. (Su et al. 2018) have also discussed further general limitations of network embedding methods, and the effects and consequences of such limitations on the use of network embedding methods in the biological domain.

7 Case Study: Predicting Protein Drug Targets

7.1 Overview

The development of drugs has a long history (Drews 2000). Until quite recently, pharmacological effects were often discovered using primitive trial and error procedures, such as applying plant extracts on living systems and observing the outcomes. Later, the drug development process evolved to elucidating mechanisms of action of drug substances and their effects on phenotype. The ability to isolate pharmacologically active substances was a key step towards modern drug discovery (Terstappen et al., Sneader 2007, 2005). More recently, advances in molecular biology and biochemistry allowed for more complex analyses of drugs, their targets and their mechanisms of action. The study of drug targets has become very popular with the objective of explaining mechanisms of actions of current drugs and their possible unknown off-target activities. Knowing targets of potential clinical significance also plays a crucial role in the process of rational drug development. With such knowledge, one can design candidate compounds targeting specific proteins to achieve intended therapeutic effects.

However, a drug rarely binds only to the intended targets, and off-target effects are common (Xie et al. 2012). This may lead to unwanted adverse effects (Bowes et al. 2012), but also to successful drug re-purposing, *i.e.* use of approved drugs for new diseases (Corbett et al. 2012). To illustrate the impact off-target effects can have in new therapy development, let us consider *aspirin* that is currently being considered for use as a chemopreventive agent (Rothwell et al. 2010). However, such a therapy would be hampered by known adverse side-effects caused by long-term use of the drug, such as bleeding of upper gastrointestinal tract (Li et al. 2017). After identifying the exact protein targets of *aspirin* that cause these adverse effects, the proteins can be targeted by newly developed and/or re-purposed drugs to avoid the unwanted side-effects of the proposed treatment.

Large-scale and reliable prediction of drug-target interactions (DTIs) can substantially facilitate development of such new treatments. Various DTI prediction methods have been proposed to date. Examples include chemical genetic (Terstappen et al. 2007) and proteomic methods (Sleno & Emili 2008) such as affinity chromatography and expression cloning ap-

proaches. These, however, can only process a limited number of possible drugs and targets due to the dependency on laboratory experiments and available physical resources. Computational prediction approaches have therefore received a lot of attention lately as they can lead to much faster assessments of possible drug-target interactions (Yamanishi et al., Mei et al. 2008, 2012).

The work of (Yamanishi et al. 2008) was one of the first approaches to predict drug targets computationally. Their approach utilised a statistical model that infers drug targets based on a bipartite graph of both chemical and genomic information. The BLM-NII (Mei et al. 2012) model was developed to improve the previous approach by using neighbour-based interaction-profile inference for both drugs and targets. More recently, (Cheng, Zhou, Li, Liu & Tang, Cheng, Liu, Jiang, Lu, Li, Liu, Zhou, Huang & Tang 2012, 2012) proposed a new way for predicting DTIs, where they have used a combination of drug similarity, target similarity and network-based inference. The COSINE (Rosdah et al. 2016) and NRLMF (Liu et al. 2015) models introduced the exclusive use of drug-drug and target-target similarity measures to infer possible drug targets. This has an advantage of being able to compute predictions even for drugs and targets with limited information about their interaction data. However, these methods only utilised a single measure to model components similarity. Other approaches such as the KronRLS-MKL (Nascimento et al. 2016) model used a linear combinations of multiple similarity measures to model the overall similarity between drugs and targets. Non-linear combinations were also explored in (Mei et al. 2012) and shown to provide better predictions.

Recently, (Hao et al. 2017) proposed a model called DNILMF that uses matrix factorisation to predict drug targets over drug information networks. This approach showed significant improvements over other methods on standard benchmarking datasets (Hao et al., Yamanishi et al. 2017, 2008). All the previously discussed works were designed to operate on generic similarities of drug structure and protein sequence, therefore they can provide efficient predictions on new chemicals. More recently, approaches that incorporate prior knowledge about drugs and targets were proposed to enhance predictive accuracy on well-studied chemicals and targets. Such models may not be best suited to de novo drug discovery. However, they may provide valuable new insights in the context of drug repurposing and understanding the general mechanisms of drug action. The current state-of-the-art work in this context is arguably the DDR model (Olayan et al. 2017), which uses a a multi-phase procedure to predict drug targets from relevant heterogeneous graphs. The gist of the approach is to combine various similarity indices and random walk features gained from the input graphs by means of non-linear fusion. Similarly, the NeoDTI model (Wan et al. 2019) predicts DTIs using supporting information about drugs and targets and a non-linear learning model over heterogeneous network data.

Despite continuous advances of similarity based approaches like DDR, these models depended on time-consuming training and prediction procedures as they need to compute the similarity features for each drug and target pair during both training and prediction. Also, the models still have a high false positive rate, especially when using large drug target interaction datasets

like DrugBank_FDA (Olayan et al. 2017).

Here, we propose a method utilising prior knowledge about drugs and targets, similarly to the DDR and NeoDTI model. Our method overcomes the afore-mentioned limitations by approaching the problem as link prediction in knowledge graphs. Our work utilises the fact that the current drug target knowledge bases like DrugBank (Wishart et al. 2006) and KEGG (Kanehisa et al. 2017) are largely structured as networks representing information about drugs in relationship with target proteins (or their genes), action pathways, and targeted diseases. Such data can naturally be interpreted as a knowledge graph. The task of finding new associations between drugs and their targets can then be formulated as a link prediction problem based on knowledge graph embeddings (Nickel, Murphy, Tresp & Gabrilovich 2016*a*).

We then use the TriVec model discussed in Chapter 5 for predicting drug target interactions in a multi-phase procedure. We first use the currently available knowledge bases to generate a knowledge graph of biological entities related to both drugs and targets. We then train our model to learn efficient vector representations (*i.e.* embeddings) of drugs and target in the knowledge graph. These representations were then used to score possible drug target pairs using a scalable procedure that has a linear time and space complexity.

We also compare our proposed approach to other state-of-the-art models using experimental evaluation on standard benchmarks. Our results show that the TriVec model outperforms all other approaches in areas under ROC and precision recall curve, metrics that are well suited to assessing general predictive power of ranking models (Davis & Goadrich 2006).

7.2 Materials

In this section we discuss the datasets that we used to train and evaluate our model. We present the standard benchmarking datasets: Yamanishi_08 (Yamanishi et al. 2008) and DrugBank_FDA (Wishart et al. 2008), and we present statistics for elements in both datasets. We also discuss some flaws in the Yamanishi_08 dataset, and we present a new KEGG based drug targets dataset that addresses these flaws.

7.2.1 Standard benchmarks

The Yamanishi_08 (Yamanishi et al. 2008) and DrugBank_FDA (Wishart et al. 2008) datasets represent the most frequently used gold standard datasets in the previous state-of-the-art models for predicting drug targets (Olayan et al. 2017). The DrugBank_FDA (Wishart et al. 2008) dataset consists of a collection of DTIs of FDA approved drugs that are gathered from DrugBank Database [1]. The Yamanishi_08 dataset is a collection of known drug target interactions gathered from different sources like KEGG BRITE (Kanehisa & Goto 2006), BRENDA (Schomburg et al. 2004), SuperTarget (Günther et al. 2007), and DrugBank (Wishart et al. 2008).

[1] https://www.drugbank.ca

Table 7.1 – Statistics of elements in the benchmarking datasets used in this work. The DTIs column represent the number of known drug target interactions, the Corruptions column represent the number of all possible combinations of drugs and targets that are not in the known drug target interactions which is used as negative in model training and evaluation, and the P2N column represents the ratio of positive to negative instances.

Dataset	Group	Drugs	Proteins	DTIs	Corruptions	P2N
Yamanishi_08	E	445	664	2926	≈ 300K	1.00%
	IC	210	204	1476	≈ 41K	3.57%
	GPCR	223	95	635	≈ 21K	3.03%
	NR	54	26	90	1314	6.67%
	All	791	989	5127	≈ 777K	0.66%
DrugBank_FDA	–	1482	1408	9881	≈ 2.1M	0.48%
KEGG_MED	–	4284	945	12112	≈ 4M	0.30%

It consists of four groups of drug target interactions corresponding to four different target protein classes: (1) enzymes (E), (2) ion-channels (IC) (3) G-protein-coupled receptors (GPCR) and (4) nuclear receptors (NR). The data in these groups vary in terms of size and positive to negative ratios as shown in table 7.1, ranging from 90 known DTIs with 1:15 as in the NR group to 2926 DTIs with 1:100 in the E group. These properties of the datasets affect the effectiveness of both training and evaluating models that use them. For example, the NR DTIs group have the largest positive to negative ratio among all the groups in the Yamanishi_08 dataset and therefore they are the easiest for predictive models in terms of evaluation. Contrary to that, the state-of-the-art models show the worst evaluation results on the NR group compared to other groups. This happens due to the low number of available DTIs training instances, which affects the models' generalisation on the training data.

7.2.2 New KEGG based benchmarking dataset

The Yamanishi_8 benchmarking dataset was published in 2008, and it contained drug target interactions from various sources including the KEGG BRITE, BRENDA, and SuperTarget databases (Yamanishi et al. 2008). In recent years, these sources have witnessed multiple developments (modifications, deletions, additions of many brand new records to their data (Placzek et al., Hecker & et. al. 2017, 2012)). These modification have directly affected the Yamanishi_08 dataset, where a subset of the identifiers of both its drugs and targets has been modified through these developments. This affects the ability to link these drugs and targets to their corresponding properties *e.g.* associated pathways, diseases, or other biological entities in the recent versions of biological knowledge bases. These modifications have also included various newly discovered drug target interactions that are not included in the Yamanishi_08 dataset. For example, the KEGG database alone contains 12112 drug target interactions, while the total number of drug target interactions in the Yamanishi_08 dataset is only 5127.

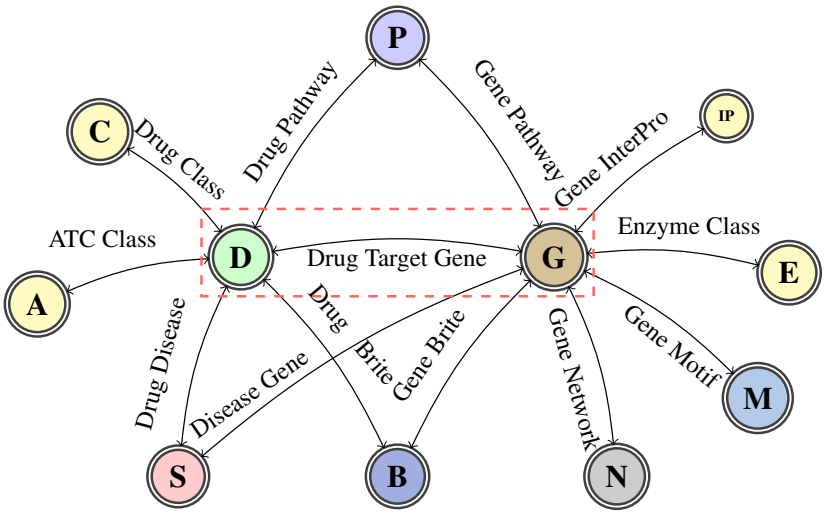

Figure 7.1 – A graph schema for a knowledge graph about drugs, their target genes, pathways, diseases and gene networks extracted from KEGG and UniProt databases.

To overcome these limitations, we propose a new drug target interaction benchmarking dataset that depends on recent versions of biological knowledge bases and includes a larger set of drug target interactions than the Yamanishi_08 dataset. We propose KEGG_MED, a dataset which is collected by extracting all the drug target interactions from the KEGG medicus database [2]. The KEGG_MED dataset contains 4284 drugs and 945 targets which are connected with 12112 drug target interactions. Table 7.1 shows a summary of statistics of the content on the dataset. Later in this chapter, we report our results on this new suggested benchmark (in addition to the comparative validation on DrugBank_FDA) so that future approaches can be compared to our model.

7.2.3 Supporting knowledge graphs

Link prediction with knowledge graph embedding models require data to be modelled in a graph form, where the objective is to predict new links between graph entities. In the case of drug target discovery, we use supporting data from biomedical knowledge bases to generate informative graphs around drug target interactions. We generate a knowledge graph for each dataset to provide descriptive features for both drugs and targets. These knowledge graphs are extracted from different sources like KEGG (Kanehisa et al. 2017), DrugBank (Wishart et al. 2006), InterPro (Mitchell & Attwood 2019) and UniProt (Consortium 2017). In our study we use a customised set of knowledge assertions about both drugs and targets. Appendix 1 and Table 1 in the supplementary material contain more information about the relation types present in each knowledge graph, and about their construction. For further information about the construction of such knowledge bases we refer to the work of (Himmelstein et al. 2017) that provides a study of systematic integration of biological knowledge for learning drug-target

[2] https://www.genome.jp/kegg/medicus.html

interactions.

We generate a group-specific knowledge graph of information extracted from KEGG and UniProt for each DTI groups in the Yamanishi_8 dataset, while we use the DrugBank with UniProt knowledge bases to model information about DTIs of the DrugBank_FDA dataset. The information extracted in both cases is modelled as a graph of interconnected biological entities (schema shown in Fig. 7.1).

7.3 Methods

The knowledge graph embedding models we use follow a generative approach to learn low-rank embedding vectors for knowledge entities and relations as shown in Chapter 4. For learning the embeddings, multiple techniques can be used, such as tensor factorisation (c.f. the DistMult model (Bordes et al. 2013)) or latent distance similarity (c.f. the TransE model (Yang et al. 2015*a*)). The goal of all these techniques is to model possible interactions between graph embeddings and to provide scores for possible graph links..

7.3.1 Embeddings representation

The TriVec model is a knowledge graph embedding model based on tensor factorisation that extends the DistMult (Yang et al. 2015*a*) and ComplEx (Trouillon et al. 2016) models. It represents each entity and relation using three embedding vectors such that the embedding of entity i is $\Theta_E(i) = \{e_i^1, e_i^2, e_i^3\}$ where all embedding vectors have the same size K (a user-defined embeddings size). Similarly, the embedding of relation j is $\Theta_R(j) = \{w_j^1, w_j^2, w_j^3\}$. e^m and w^m denote the m part of the embeddings of the entity or the relation, and $m \in \{1,2,3\}$ represents the three embeddings parts.

In the context of drug target prediction, the TriVec model convert drug target facts into triplets in the form of (drug_x, drug-target-relation, protein_y). It then uses the embeddings of the drug_x, drug-target-relation and protein_y to represent them, where the embedding of each component consists of three numerical vectors.

The embeddings in the TriVec model are initially with random values generated by the Glorot uniform random generator (Glorot & Bengio 2010). The embedding vectors are then updated during the training procedure to provide optimised scores for the knowledge graph facts.

7.3.2 Training procedure

The TriVec is a knowledge graph embedding model that follows the multi-phase procedure discussed in Chapter 5 to effectively learn a vector representation for entities and relation of a knowledge graph. First, the model initialises its embeddings with random noise. It then updates them by iterative learning on the training data. In each training iteration *i.e.* epoch,

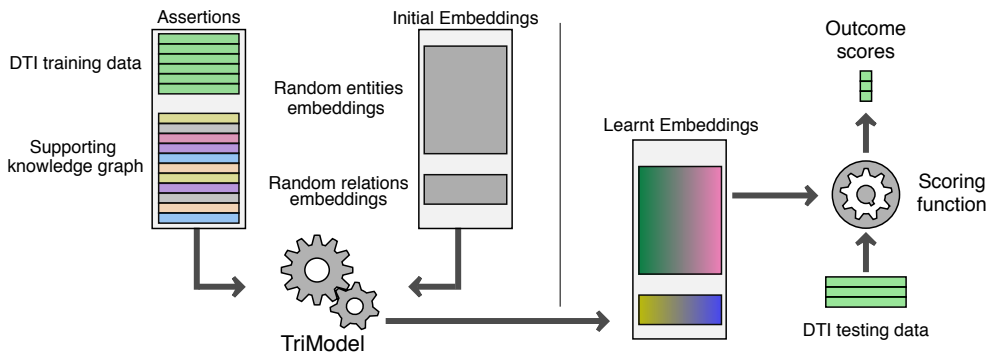

Figure 7.2 – A diagram of the training pipeline of the TriVec model. Both drug target interactions and supporting knowledge graph assertions are combined and used as input to the model along with initial random embeddings for both entities and relations. The outcome of the training procedure is learnt embeddings which is used to score any drug target interaction data of drugs and proteins processed during the training processes.

the model splits the training data into mini-batches and executes its learning pipeline over each batch. The learning pipeline of the model learns the embeddings of entities and relations by minimising a negative softmax log-loss Eq 4.6. The scores of the TriVec model are computed using the embeddings interaction function (scoring function) defined in Eq. 5.6. It uses a set of three interactions: one symmetric interaction: $(e_s^2 w_p^2 e_o^2)$ and two asymmetric interactions: $(e_s^1 w_p^1 e_o^3)$ and $(e_s^3 w_p^3 e_o^1)$ for a convenient graphical explanation of the interaction. This approach models both symmetry and asymmetry in simple form similar to the DistMult (Yang et al. 2015*a*) model where the DistMult model can be seen as a special case of the TriVec model if the first and third embeddings parts are equivalent $(e^1 = e^3)$.

Fig. 7.2 presents and illustration of the training and prediction pipeline of our approach with the drug target interaction and supporting knowledge graph data.

7.4 Results

In this section we describe the configuration of the data used in the experimentation, the evaluation protocol, the setup of our experiments and the results and findings of our experiments. We also compare the predictive accuracy of our model to selected existing approaches, including the state-of-the-art one.

7.4.1 Evaluation protocol

In order to facilitate comparison with the state-of-the-art models, we use a 10-fold cross validation (CV) to evaluate our model on the Yamanishi_08 and DrugBank_FDA datasets. First, we split the drug target interaction data into 10 splits *i.e.* folds. We then evaluate the model 10 times on each split, where the model is trained on the other 9 splits. This procedure is repeated

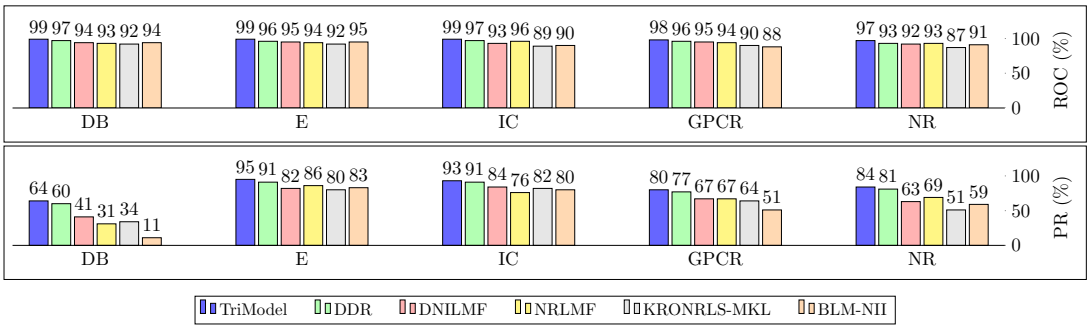

Figure 7.3 – Bar chart for the values of the area under the roc curve (AUC-ROC) and area under the precision recall curve (AUC-PR) for the TriModel compared to other state-of-the-art models on standard benchmarking datasets. All values are rounded to two digits and multiplied by 100 to represent a percentage (%). DB represents the DrugBank_FDA dataset.

5 times and average results across these runs are reported. This is to further minimise the impact of data variability on the result stability.

In each training configuration we use the known drug target interactions as positives, and all other possible combinations between the investigated dataset drugs and protein targets as negatives. This yields different positive to negative ratios since the datasets have different number of drugs, targets, and drug target interactions (see Table 7.1 for exact statistics of the ratios for each dataset).

We use the area under the ROC and precision recall curves (AUC-ROC and AUC-PR respectively) as an indication of the predictive accuracy of our model. We compute both metrics on the testing data (DTIs), where we divide the testing data into three groups: (1) S_p, containing testing drug target interactions where both the drug and the target are involved in known drug target interactions in the training data, (2) S_d, containing testing drug target interactions which contain drugs that have no known drug target interactions in the training data, (3) S_t, containing testing data of targets that has not involved in any known drug target interactions in the training data. The main reason for splitting the data this way was that one of the methods could not be compared with the others on the S_t, S_p data. The largest S_p group, however, generally exhibits least fluctuations across particular cross-validation runs as it has the highest ratio across all testing splits.

We also compute aggregated weighted AU-ROC, AU-PR scores for comparing the different models regardless the data group. These scores are defined as follows:

$$M = \sum_g \omega_g \cdot M_g, \tag{7.1}$$

where $g \in \{S_p, S_d, S_t\}$, M represents the aggregated score (AUC-ROC or AUC-PR), M_g is the specific score value for the group g, and ω_g is the weight of the particular data group computed by dividing the number of instances in g by the total number of instances in $S_p \cup S_d \cup S_t$.

7.4.2 Experimental setup

We use the supporting knowledge graph to perform a grid search to learn the model's best hyperparameters. In all of our experiments we initialise our model embeddings using the Glorot uniform random generator (Glorot & Bengio 2010) and we optimise the training loss using the AMSGrad optimiser (Reddi et al. 2018), where the the learning rate (lr) $\in \{0.01, 0.02, 0.03\}$, embeddings size (K) $\in \{50, 100, 150, 200\}$ and batch size (b) $\in \{128, 256, 512, 1024, 4000\}$. The rest of the grid search hyper parameters are defined as follows: the regularisation weight (λ) $\in \{0.1, 0.3, 0.35, 0.01, 0.03, 0.035\}$, dropout ($d$) $\in \{0.0, 0.1, 0.2, 0.01, 0.02\}$. The number of training epochs is fixed to 1000.

We use Tensorflow framework (GPU) along with Python 3.5 to perform our experiments. All experiments were executed on a Linux machine with processor Intel(R) Core(TM) i70.4790K CPU @ 4.00GHz, 32 GB RAM, and an nVidia Titan Xp GPU. We include the training runtime of the TriVec model for each cross-validation iteration for all the investigated benchmarks in Fig. 1 in the supplementary materials.

7.4.3 Comparison with state-of-the-art models

We evaluate our model on the Yamanishi_08 and DrugBank_FDA datasets, and we compare our results to the following state-of-the-art models: DDR (Olayan et al. 2017), NRLMF (Hao et al. 2017), NRLMF (Liu et al. 2015), KRONRLS-MKL (Nascimento et al. 2016), COSINE (Lim et al. 2016), and BLM-NII (Mei et al. 2012). The comparison is made using the metrics of area-under-the-ROC (AUC-ROC) and precision-recall (AUC-PR) curves.

Fig. 7.3 presents overall results in terms of the AUC-ROC and AUC-PR scores for all compared models. The overall scores are combined across all testing configurations (S_p, S_d, S_t) for each dataset, where each specific score is computed as described in Eq. 7.1.

The results show that the TriVec model outperforms all other models in terms of AUC-ROC and AUC-PR on every benchmarking dataset. The TriVec model achieves a better AUC-PR score with a margin of 4%, 2%, 3%, 3%, 4% on E, IC, GPCR, NR, and DrugBank_FDA datasets respectively. It should be noted that we did not include the COSINE method in Fig. 7.3 as it is specifically designed to predict new drugs that do not have DTIs in the training phase. As such, the description of the method only reports accuracy on the new drug configuration (S_d), while the presented combined scores require values of all three evaluation configurations.

Table 7.2 shows a detailed comparison of the TriVec model and state-of-the-art models on all the standard benchmarking datasets for the the different evaluation settings S_p, S_d, and S_t. It also shows the relative number (in per cent) of drug-target statements available for each of the three validation settings.

The results in Table 7.2 show that the TriVec model outperforms other state-of-the-art models on 13 out of 15 different AUC-ROC experimentation configurations. In case of AU-PR, our

Table 7.2 – A comparison with state-of-the-art models on standard datasets using multiple configurations (S_p, S_d, S_t). The state-of-the-art results were obtained from (Olayan et al. 2017). The count (%) represents the percentage of the configuration instances, and the DB and KM columns represent DrugBank_FDA and KEGG_MED respectively. All the experimental configurations on all the datasets are evaluated using a 10-fold cross validation which is repeated 5 times. Column M. represents metrics and column Ft. represent model's feature type. The *structure* feature type represents protein and drug structure based features and *Ext.* denotes extensive prior knowledge features. Underlined scores represent the best scores in their feature category while the overall best results are bold and highlighted in green.

M.	Model	Ft.	E			IC			GPCR			NR			DB			KM		
	Config.		S_d	S_t	S_p	S_d	S_t	S_p	S_d	S_t	S_p	S_d	S_t	S_p	S_d	S_t	S_p	S_d	S_t	S_p
	Count (%)		4%	5%	91%	4%	1%	95%	5%	4%	91%	10%	4%	86%	4%	11%	85%	5%	3%	92%
AUC-ROC	BLM-NII	Structure	0.73	0.89	0.96	0.83	0.89	0.91	0.85	0.87	0.88	0.88	0.85	0.91	0.71	0.75	0.90	-	-	-
	COSINE		0.80	-	-	0.82	-	-	0.88	-	-	0.89	-	-	0.77	-	-	-	-	-
	KRONRLS-MKL		0.71	0.88	0.93	0.77	0.86	0.90	0.81	0.84	0.91	0.79	0.76	0.87	0.79	0.81	0.88	-	-	-
	NRLMF		0.75	0.90	0.95	0.80	0.93	0.98	0.87	0.92	0.95	0.88	0.83	0.93	0.89	0.80	0.93	-	-	-
	DNILMF		0.81	0.92	0.96	0.81	0.92	0.94	0.86	0.92	0.96	0.83	0.83	0.92	0.90	0.82	0.95	-	-	-
	DDR	Ext.	0.84	0.92	0.97	**0.94**	0.97	0.98	0.91	**0.93**	0.96	**0.90**	**0.88**	0.92	0.91	0.86	0.96	-	-	-
	TriVec		**0.95**	**0.96**	**0.99**	0.93	**0.98**	**0.99**	**0.92**	0.86	**0.99**	0.89	0.85	**0.99**	**0.94**	**0.94**	**0.99**	**0.81**	**0.58**	**0.99**
AUC-PR	BLM-NII	Structure	0.22	0.73	0.86	0.37	0.61	0.83	0.35	0.37	0.53	0.35	0.41	0.62	0.03	0.05	0.12	-	-	-
	COSINE		0.35	-	-	0.36	-	-	0.40	-	-	0.56	-	-	0.30	-	-	-	-	-
	KRONRLS-MKL		0.07	0.07	0.87	0.23	0.23	0.86	0.31	0.37	0.67	0.49	0.46	0.51	0.22	0.18	0.35	-	-	-
	NRLMF		0.28	0.76	0.89	0.30	0.61	0.79	0.36	0.55	0.69	0.49	0.45	0.72	0.28	0.23	0.32	-	-	-
	DNILMF		0.30	0.76	0.85	0.30	0.61	0.87	0.31	0.56	0.70	0.41	0.52	0.66	0.24	0.21	0.42	-	-	-
	DDR	Ext.	0.73	0.82	0.92	0.69	0.80	0.92	0.63	0.61	0.79	0.71	0.64	0.83	0.44	0.39	0.61	-	-	-
	TriVec		**0.78**	**0.83**	**0.96**	**0.76**	**0.87**	**0.95**	**0.81**	**0.73**	**0.80**	**0.87**	**0.77**	**0.84**	**0.59**	**0.62**	**0.64**	**0.18**	**0.18**	**0.94**

model is better 14 out of 15 configurations. The results also show that the experimental configurations where our model is not the best represent a small portion of the total number of DTIs, while the TriVec model provides consistently better results for the largest S_p partition of the validation data.

Table 7.2 also show the results of the TriVec model on our proposed KEGG_MEDD dataset, where the model's AUC-PR scores are 0.18, 0.18, and 0.94 and its AUC-ROC scores are 0.81, 0.58, and 0.99 on the configurations S_d, S_t, and S_p respectively. No comparison with existing tools has been performed as their published versions cannot be directly applied to this data set.

7.4.4 Limitations

Despite the very promising results achieved by the prior knowledge-based models like DDR and TriVec , their predictive capabilities are best suited to finding new associations between well-studied drugs and targets (useful for instance in the drug repurposing context). If one needs predictions for de novo drug discovery, the models that utilise drug structure and target sequence similarities (e.g. BLM-NII, COSINE, KRONRLS-MKL, NRLMF or NRLMF) will likely deliver better results.

7.4.5 Web application for exploring the TriVec predictions

To let users explore our results, we have designed a web application [3]. The application allows for searching the predictions of the TriVec model. One can look for predictions using either drugs or targets as queries. Queries concerning multiple entities are possible simply by appending new terms to the search query. The results are presented as a table of the TriVec model scores of all the possible drug-target associations of the searched term.

The predictions provided by the web application are learnt by training the TriVec model on all the Yamanishi_08 dataset. The prediction scores are then computed for all possible drug-target combinations induced by the dataset. The scores of known drug interactions in the Yamanishi_08 dataset are set to 1, while the scores of all other drug target interactions are the normalised outcome of the TriVec predictions. The table of predictions in the application indicates the origin of each score, where a unique label "*Experimental Evidence*" is given to known DTIs and another label "*Model Prediction*" is assigned to the predicted scores.

7.5 Discussion

In the following we discuss possible reasons for the improved performance of our approach when compared to existing methods. We also review the limitations of the current DTI pre-

[3] Hosted at: http://drugtargets.insight-centre.org.

diction benchmarks and discuss impact of data stratification on the predictive power of the models. Last but not least, we present tentative results in expert-based validation of predictions of our model that are not covered by the benchmark datasets. These results show high promise in terms of actual new discoveries predicted by our model.

7.5.1 Distinctive features of the presented approach

The relative success of the TriVec model can be attributed to two distinctive features not present in the state-of-the-art models. Firstly, we model input for the training as knowledge graphs. This allows for encoding multiple types of associations within the same graph and thus utilising more complex patterns. Other models that use graph-based data are limited in this respect as they only employ networks with single relation type. Secondly, the TriVec model uses a generative approach to learn efficient representations for both drugs and their targets. This approach enables scalable predictions of large volumes of drug-target interactions as it uses linear training time (Nickel, Murphy, Tresp & Gabrilovich 2016*a*) and constant prediction time, which is not the case of the existing works. Furthermore, the TriVec model is able to predict other biological associations within the training data (*e.g.* drug and target pathways) with no extra computational effort. This shows substantial promise for further development of this technique.

7.5.2 Impact of data stratification on the predictive power

The Yamanishi_08 dataset is divided into four groups of DTIs according to the functionality of the target proteins. The groups are enzymes (E), ion-channels (IC) G-protein-coupled receptors (GPCR), and nuclear receptors (NR). The objective of this categorisation is to distinguish between models specifically tailored to predicting targets associated with a particular drug class (Yamanishi et al. 2008). Olayan et al. (2017) confirmed that organising the drug target interactions into groups according to the target's biological functionality enhances the predictive accuracy of models trained on such stratified data.

Based on our observations, we suggest a different explanation. The differences in performance appear to correlate with the relative numbers of negative examples in the grouped and full dataset configuration. Table 7.1 shows that the full Yamanishi_08 dataset configuration has a 0.66% positive to negative ratio, while the groups E, IC, GCPR, and NR have 1%, 3.57%, 3.03%, and 6.67% respectively. These differences can explain the variability of model performance quite well, since predicting positive instances is generally harder with more negatives present in the data (Liu et al. 2009). In addition, dividing the DTI information gives rise to groups like the GPCR and NR groups. These contain only a small number of true DTIs (635 and 90 DTIs respectively), which further hampers the ability of models to generalise well (as we show in Section 7.2).

7.5.3 Validating the discovery potential of TriVec

Good performance of a model in benchmark tests is no doubt important. For various reasons like overfitting or training data imbalances, however, good benchmark results may not necessarily mean that the model can effectively support new discoveries.

Laboratory validation can ultimately confirm the model predictions as actual discoveries, but this is costly and time-consuming to be done at large scale. One can, however, perform alternative validations of the predictions using data that was not used for training the model. Such complementary validation can provide stronger foundations for claiming a model has high generalisation power.

A domain expert have performed a complementary validation of the TriVec 's predictions by manual analysis of top-10 drug-target associations per each of the examined benchmarking datasets. To decide whether or not the associations are true positives, the expert reviewed each drug target interaction assertion and validated them using available literature article which mention these interactions as an evidence of interaction. We only validated the predictions that were not part of the training data. The validation outcome shows that the TriVec model achieves 7 out of 10, 7 out of 10, 8 out of 10, 7 out of 10 and 6 out of 10 true predictions on the E, IC, GPCR, NR, DB datasets respectively. The results of our validation is presented in Table. 7.3.

One can easily see that our model puts actual drug-target introductions (some of which were only recently discovered) high up in the result list. This is very promising for further development of the model and its deployment in clinical application scenarios.

Dataset	#	Drug Name	Drug Id	Target Name	Target ID	Score	Valid	Evidence
E	1	Halothane	D00542	CYP2E1	hsa:1571	8.820	YES	PubMed:19442086
	2	Aminocaproic acid	D00160	PROC	hsa:5624	8.601	UNK	-
	3	Imatinib mesylate	D01441	MAPK1	hsa:5594	8.355	YES	PubMed: 22089930
	4	Methoxsalen	D00139	CYP1A1	hsa:1543	8.323	YES	PubMed: 7702611
	5	Isoflurophate	D00043	ELANE	hsa:1991	8.311	UNK	-
	6	Imatinib mesylate	D01441	MAPK3	hsa:5595	8.295	YES	PubMed: 15100154
	7	Metyrapone	D00410	CYP1A1	hsa:1543	8.275	YES	PubMed: 9512490
	8	Salicylic acid	D00097	PTGS2	hsa:5743	8.184	No	-
	9	Nifedipine	D00437	CYP2C9	hsa:1559	8.140	YES	PubMed: 9929518
	10	Aminoglutethimide	D00574	CYP21A2	hsa:1589	8.132	YES	PubMed: 8201961
IC	1	Nicotine	D03365	CHRNA4	hsa:1137	6.486	YES	PubMed:17590520
	2	Zonisamide	D00538	SCN5A	hsa:6331	6.468	YES	PubMed:20025128
	3	Benzocaine	D00552	SCN5A	hsa:6331	6.380	YES	PubMed:19661462
	4	Nimodipine	D00438	CACNA1S	hsa:779	6.297	YES	PubMed:16675661
	5	Metoclopramide	D00726	CHRNA5	hsa:1138	6.285	UNK	-
	6	Isoflurane	D00545	GLRA2	hsa:2742	6.262	UNK	-
	7	Diazoxide	D00294	ABCC9	hsa:10060	6.198	YES	PubMed: 21428460
	8	Prilocaine	D00553	SCN10A	hsa:6336	5.992	YES	PubMed:17139284
	9	Verapamil hydrochloride	D00619	CACNA1F	hsa:778	5.961	YES	PubMed:19125880
	10	Nimodipine	D00438	CACNA2D1	hsa:781	5.940	UNK	PubMed: 29176626
GPCR	1	Isoetharine	D04625	ADRB2	hsa:154	7.148	YES	PubMed:21948594
	2	Octreotide acetate	D02250	SSTR1	hsa:6751	6.752	YES	PubMed:16438887
	3	Clonidine hydrochloride	D00604	ADRA1B	hsa:147	6.650	YES	PubMed: 17584443
	4	Metoprolol	D02358	ADRB2	hsa:154	6.499	YES	PubMed:19637941
	5	Epinephrine	D00095	ADRA1D	hsa:146	6.489	YES	PubMed:20954794
	6	Theophylline	D00371	ADORA2A	hsa:135	6.407	YES	PubMed:16357952
	7	Denopamine	D02614	ADRB2	hsa:154	6.388	NO	PubMed: 22505670
	8	Risperidone	D00426	DRD2	hsa:1813	6.386	YES	PubMed:17059881]
	9	Bosentan	D01227	AGTR1	hsa:185	6.347	UNK	-
	10	Epinephrine	D00095	ADRA1B	hsa:147	6.306	YES	PubMed:20954794
NR	1	Medroxyprogesterone acetate	D00951	ESR1	hsa:2099	6.314	YES	PubMed:17094978
	2	Mometasone furoate	D00690	NR3C1	hsa:2908	6.066	YES	PubMed:8439518
	3	Ethinyl estradiol	D00554	ESR2	hsa:2100	6.038	NO	PubMed: 15878629
	4	Dydrogesterone	D01217	ESR1	hsa:2099	5.968	YES	PubMed: 22878119
	5	Norethindrone	D00182	ESR1	hsa:2099	5.893	YES	PubMed: 27245768
	6	Etretinate	D00316	RORB	hsa:6096	5.848	UNK	-
	7	Mifepristone	D00585	ESR1	hsa:2099	5.841	YES	PubMed: 15001543
	8	Tretinoin	D00094	RORA	hsa:6095	5.679	YES	CheMBL
	9	Tazarotene	D01132	RORC	hsa:6097	5.463	UNK	-
	10	Testosterone	D00075	ESR1	hsa:2099	5.453	YES	PubMed:12676605
DB	1	Methysergide	DB00247	HTR1D	P28221	6.421	YES	PubMed: 7984267
	2	Phenoxymethylpenicillin	DB00417	SLC15A2	Q16348	6.295	UNK	-
	3	L-Valine	DB00161	BCAT2	O15382	6.263	YES	PubMed: 6933702
	4	Corticorelin ovine triflutate	DB09067	GHRHR	Q02643	6.237	UNK	-
	5	Acarbose	DB00284	GANC	Q8TET4	6.232	YES	KEGG
	6	Halothane	DB01159	GRIA1	P42261	6.226	YES	PubMed: 14739810
	7	Hydroxocobalamin	DB00200	GIF	P27352	6.226	UNK	-
	8	Quazepam	DB01589	GABRB2	P47870	6.215	YES	PubMed:6738302
	9	Nintedanib	DB09079	KIT	P10721	6.202	UNK	-
	10	Miglitol	DB00491	SI	P14410	6.201	YES	CheMBL

Table 7.3 – Validation of the top 10 scored combination for each of the investigated datasets. The DrugBank, CheMBL and KEGG DTIs are used as evidence the different interactions, where the PubMed ID is listed when possible. UNK represents unknown interactions.

8 Case Study: Predicting Tissue-Specific Protein Functions

8.1 Overview

Proteins are complex molecules that are involved in almost all biological processes. They are widely expressed in all parts of the human body where they interact together and execute multiple biological functions. These functions are essential to sustain the biological activities of the living system. Proteins are usually expressed in specific tissues within the body where their precise interactions and biological functions are frequently dependant on their tissue context (Fagerberg et al., Greene et al. 2014, 2015). The disorder of these interactions and functions results in diseases (D D'Agati, Cai & Petrov 2008, 2010). Thus, the deep understanding of tissue-specific protein activities is essential to elucidate the causes of diseases and their possible therapeutic treatments.

Although direct lab-based assay of tissue-specific functions of proteins remains infeasible in many human tissues (Greene et al. 2015), computational approaches can be used to infer this information on a large scale. These approaches work by analysing the protein interactome and known tissue-specific protein functions (Zitnik & Leskovec 2017); they then provide scores for possible new unknown protein functions.

The early computational approaches for predicting protein functions have used sequence alignment similarity between proteins to infer their functions (Marcotte et al. 1999). These models have worked under the assumption that similar sequences are correlated with similar functions. However, recent studies have shown that this correlation is weak (Clark & Radivojac 2011), and sequence alignment alone is not sufficient for predicting protein functions (Radivojac et al. 2013). Therefore, further methods have utilised an extended set of features including protein structure (Pazos & Sternberg 2004), protein-protein interactions (Letovsky & Kasif 2003) and gene expression data (Enault et al. 2005). These methods have shown a significant improvement in terms of the predictive accuracy over the traditional sequence alignment methods (Radivojac et al. 2013). However, all these approaches address the problem of pre-

dicting protein function in a generic context, where they have assumed that protein functions are the same in all the tissues and cell linages that they are expressed in. Radivojac et. al. (Radivojac et al. 2013) have provided a large-scale study of these approaches, where they have analysed the differences between their underlying concepts and utilised features. They have also performed a comparative empirical evaluation between these approaches on standard benchmarks.

Recently, Zitnik et. al. (Zitnik & Leskovec 2017) developed a new technique, the OhmNet model, for tissue-specific protein function prediction. The technique models tissue-specific protein interactome and functions as a hierarchical multilayer network, where the tissues' hierarchy is used to support the hierarchical network-based learning architecture. They have then used unsupervised feature learning to represent and score possible protein functions for each tissue. Although the empirical evaluation of the OhmNet model shows that it outperforms all other state-of-the-art techniques, the model still suffers from a high rate of false positives (Zitnik & Leskovec 2017).

Despite the high availability of generic protein function associations (Consortium 2003), these associations occur only within their corresponding specific tissues and cell-lines in living systems. Moreover, the available curated data on tissue-specific protein function associations is limited (Greene et al. 2015). This affects the predictive capabilities of computational methods that operate on this data. Furthermore, the large numbers of proteins in the human body and their possible associated biological functions increase the sparsity of available data, therefore, increase the difficulty of the problem. Thus, the development of computational methods in this regard aim at developing methods that can operate on limited data with high sparsity.

In this chapter, we try to re-address the problem by formalising it as a tensor completion task. The tissue-specific protein functions and interactome in this context can be naturally modelled using three dimensional tensors of proteins, functions, and their corresponding tissues. Then, the probability that a protein x has a function y in the tissue z is modelled by the adjacent tensor cell (x, y, z) corresponding to the protein, function, and tissue. In this context, known values can be used to populate tensor initial values, and a learning method can then be effectively used to complete the missing score values. This problem then represents a basic tensor completion problem, where empirical evidence has shown that tensor factorisation methods provide state-of-the-art results in terms of both accuracy and scalability of their predictions (Trouillon et al., Lacroix et al. 2016, 2018).

In their work, Zitnik et. al. (Zitnik & Leskovec 2017) compared their OhmNet model to other models including the RESCAL tensor decomposition model (Nickel et al. 2011), where the RESCAL model showed the worst results in the comparison. However, we believe that tensor

decomposition based techniques can outperform other techniques in predicting protein functions due to the following reasons:

1. The RESCAL model used in the OhmNet experiments is one of the earliest developed tensor decomposition techniques, and more recent models such as the ComplEx model (Trouillon et al. 2016) are known to provide significantly better predictive accuracy.

2. The training procedure of the tensor factorisation models require an extensive grid search, and its predictive accuracy is sensitive to the hyper parameter search space (Kadlec et al. 2017). Therefore, careless training of tensor decomposition models can lead to significantly poor predictive accuracy.

3. Tensor decomposition models can be designed with different loss objectives, where logistic based loss objectives are known to provide better results compared to the squared error based loss objective of the RESCAL model (Nickel et al., Trouillon et al., Mohamed, Novácek, Vandenbussche & Muñoz 2011, 2016, 2019).

4. The latent vector representation, *i.e.* embeddings, of the tensor elements have a significant effect on tensor decomposition based models, where the multi-vector representations are known to provide better results than simple representations of the RESCAL models (Trouillon et al., Mohamed & Novácek 2016, 2019).

In this chapter, we re-introduce the use of tensor decomposition models in predicting protein function by addressing the above mentioned issues. First, we assess the predictive accuracy of the ComplEx model—which is a more recent tensor decomposition technique that encodes tensor elements using vectors in the complex space (Trouillon et al. 2016). We also show by an experimental evaluation that the ComplEx model outperforms the OhmNet model in terms of both the area under the ROC and precision recall curves. Secondly, we introduce the use TriVec model (*cf.* Chapter 5), which is a tensor decomposition model that encodes the tensor elements in a hyper complex space. We then show by an experimental evaluation that it outperforms other state-of-the-art models including the OhmNet and ComplEx models in terms of both the area under the ROC and precision recall curves.

8.2 Background

In this section, we discuss the problem of predicting protein targets, the tensor decomposition procedure, and the current state-of-the-art tensor decomposition models. In this section, we also define the notations that we use throughout this chapter which differs from the notations used in Chapter 4 and Chapter 5 as we study the knowledge graph embedding models as pure tensor decomposition approaches where we use the notion of tensors instead of graphs.

8.2.1 Tissue-Specific protein functions

Proteins are large biomolecules that consist of a sequence of amino acids. They perform a wide range of functions within living systems, including catalysing metabolic reactions, DNA replication, responding to stimuli, providing structure to cells and organisms, and transporting molecules from one location to another (Nelson et al. 2008). Proteins are expressed with different levels in different parts of the human body—where they have high expression levels in some parts and low or absent expressions in others. These human parts are categorised in a hierarchy of tissues that represent cell tissues, organs and functional systems. For example, the cerebellum tissue is part of the brain tissue, which is also a part of the nervous system tissues. The exact biological functions and interactions of proteins vary depending on the tissues they are expressed in, where their functions in a specific tissue can vary in other tissues. Therefore, the task of predicting precise protein function is frequently associated to a specific tissue.

Computationally, the problem can be defined as follows: given a set of proteins, biological functions, tissues, and a set of known tissue-specific protein functions associations, predicting tissue-specific protein functions require providing scores for each (protein, tissue, function) combination such that the score of any given true combination is greater than the score of all other false combinations. The evaluation of the learnt scores can then be evaluated using standard classification or ranking metrics. In this chapter, we assess the predictive accuracy of all the methods using the area under ROC (AUC-ROC) and precision recall (AUC-PR) curves as established by previous works (Zitnik & Leskovec 2017).

8.2.2 Tensor decomposition

Scalars are singular numerical values, vectors are one dimensional numerical arrays, matrices are two dimensional numerical arrays, and tensors are numerical arrays with three or more dimensions. The objective of the procedure of tensor decomposition *i.e.* tensor factorisation, is to complete all cell values in an incomplete tensor using a set of initial known cell values.

Let $\mathbf{M}$ be a three dimensional tensor, where the three dimensions represent objects of different sets X, Y, Z. Any element (i, j, k) in the tensor represents the interaction between the components $i \in X$, $j \in Y$, and $k \in Z$. We denote the weight of this interaction using $\eta^{\mathbf{M}}(i, j, k)$. In this chapter, we use a tensor M with elements of the three sets: proteins ($\mathbf{P}$), functions ($\mathbf{F}$), and tissues ($\mathbf{T}$). The objective of tensor decomposition then is to complete the tensor values such that the weight of any interaction for a true known protein function in a specific tissue is larger than all other known false combinations.

where $p \in \mathbf{P}$, $f \in \mathbf{F}$, $t \in \mathbf{T}$, (p, f, t) is any known true combination of a protein function and tissue such that the protein p has function f in the tissue t, and combinations $(p, f, t)'$ represent

any other false combinations. This objective is achieved using a multi-phase procedure as discussed in Chapter 4.

Canonical tensor decomposition

The canonical tensor decomposition model (Hitchcock 1927) is the earliest approach for factorising tensors using sums of products of their objects. It uses multiple embedding matrices such that every tensor dimension is associated to an embedding matrix. Each object in this dimension is then associated to a vector in the matrix, where the weight of any combination of objects in the tensor is computed as the sum of the product of their corresponding vectors as follows:

$$\eta^{\mathbf{M}}_{(x,y,z)} = \sum_{k} \overset{1}{\mathbf{e}}_x \otimes \overset{2}{\mathbf{e}}_y \otimes \overset{3}{\mathbf{e}}_z \tag{8.1}$$

where $\overset{1}{\mathbf{e}}_x$ represents the embedding vector corresponding to object x sampled from the embedding matrix of dimension 1, $\otimes$ is the vector component-wise product operator, and k represents the size of the embedding vectors.

The RESCAL model

The RESCAL model (Nickel et al. 2012) on the other hand was developed to deal with 3D tensors that represent knowledge on linked data, where two tensor dimensions represent knowledge entities and the other dimension model the relations between these entities. It only uses one embedding matrix for the entities dimensions, and an embedding tensor for the relations dimension where each relation is represented by an embedding matrix. The weight of tensor combinations is then defined as follows:

$$\eta^{\mathbf{M}}(x,y,z) = \overset{1}{\mathbf{e}}_x \overset{2}{\mathbf{W}}_y \overset{1}{\mathbf{e}}_z \tag{8.2}$$

where $\overset{2}{\mathbf{W}}_y$ is a $k \times k$ embedding matrix of the relation y sampled for the embeddings of dimension 2.

Complex tensor decomposition

Trouillon et. al. (Trouillon et al. 2016) proposed a new technique for tensor decomposition that represents the object embeddings using complex vectors. Each embedding is then modelled by two vectors: real and imaginary. Such a technique allowed the modelling of non-symmetric interactions between tensor combinations where it used the asymmetric *Hermitian* dot product of its combinations, The embedding interaction function of the ComplEx model

is then defined as follows:

$$\eta^{\mathbf{M}}(x, y, z) = \quad \mathrm{Re}(\overset{1}{\mathbf{e}}_x \odot \overset{2}{\mathbf{e}}_y \odot \overline{\overset{1}{\mathbf{e}}_z}), \tag{8.3}$$

where $\odot$ denotes the *Hermitian* complex product, $\overline{\overset{1}{\mathbf{e}}_z}$ is the complex conjugate of the complex embeddings $\overset{1}{\mathbf{e}}_z$, and $\mathrm{Re}(x)$ denotes the real part of the complex number x.

8.2.3 Related Work

In this section, we present the state-of-the-art related works in predicting protein functions, and general tensor decomposition models.

Predicting protein functions

Computational methods are widely used for modelling complex biological networks of protein (Wei et al. 2017), complexes (Ma & Gao 2012), pathways (Maji et al. 2017), and other biological entities. They provide support for analysing and inference on biological data (Mrozek et al. 2017). They are used to infer different types of associations between biological entities such as protein-protein interactions (Sun et al. 2018), gene-disease associations (Lei & Zhang 2019), protein-complex relations, drug-targets interactions (Mohamed, Nounu & Novácek 2019) and other general links between different biological concepts. In this chapter, we focus on computational approaches that learns associations between proteins and their functions in the human body.

The tissue-specific network propagation model (Magger et al. 2012) is one of the earliest attempts for learning tissue-specific predictions over genes. This approach depends on propagating initial known gene scores associated with a known query function to similar function entities in the networks. The method was firstly designed to predict tissue-specific disease gene associations. The model was later used for predicting tissue-specific protein functions (Zitnik & Leskovec 2017). Similarly, the network-based tissue-specific support vector machine model (Net-SVM) (Guan et al. 2012) was firstly developed to predict tissue-specific disease gene associations and gene phenotypes. I was then used for predicting tissue-specific protein functions (Zitnik & Leskovec 2017). The GeneMania model (Warde-Farley et al. 2010) suggested another propagation based approach for predicting tissue-specific protein functions. In this method, the tissue-specific networks are firstly combined into one weighted network. Known protein function associations' weights are then propagated to allow predicting other unknown protein functions.

The Minimum Curvilinear Embedding (MCE) model (Cannistraci et al. 2013) is one of the earliest embeddings-based models for predicting protein functions in the human body. The

method utilises network embeddings of protein-protein interaction networks to learn an efficient representation of proteins within these networks. It was first used to predict protein-protein interactions; then it was extended to allow prediction of protein function associations.

Zitnik et. al. have examined the LINE model (Tang et al. 2015) in the task of predicting new tissue-specific protein functions, where it showed better scores than the MCE model in terms of the area under the precision recall curve. The model uses a composite learning technique where it learns half of the embeddings' dimensions from the direct neighbour nodes, and the other half from the second hop connected neighbours.

Furthermore, the Node2vec model (Grover & Leskovec 2016) is another approach that works by generating network embeddings using biased random walks as features. It uses a mixture of width and depth based network search to generate flexible views of network nodes to learn their embeddings. The Node2vec model was able to predict tissue-specific protein functions with better area under the ROC and precision recall curves than the MCE and LINE models (Zitnik & Leskovec 2017).

Recently, Zitnik et. al. have developed the state-of-the-art model, the OhmNet model (Zitnik & Leskovec 2017), a hierarchy-aware unsupervised learning method for multi-layer networks. It models each tissue information as a separate network, and learns efficient representations for proteins and functions by generating their embeddings using the tissue-specific protein-protein interactome and protein functions. The experimental evaluation of the OhmNet model outperforms all other state-of-the-art models in terms of both the area under the roc and precision recall curves in predicting tissue-specific protein functions.

Tensor and Matrix Decomposition

Tensor and matrix decomposition methods are generative learning methods that operate by learning low-rank representation of elements in a tensor (Ji et al., Lu, Lai, Xu, You, Li & Yuan, Jiang et al. 2016, 2016, 2018). They have been widely adopted to achieve multiple tasks such as link prediction (Trouillon et al., Lacroix et al. 2016, 2018), recommendation systems and different classification (Zou et al., Hong & Jung 2015, 2018) and clustering tasks (Buono & Pio, Lu, Zhao, Zhang & Li 2015, 2016). Early factorisation methods were developed as representation learning technique for tensors and they were expressed as a sum of products (Hitchcock 1927). They have then evolved to utilise multiple forms of products including dot products, bilinear products and complex products of the representations of tensor elements (Yang et al., Trouillon et al. 2015*b*, 2016).

In this chapter, we focus on current developments of tensor factorisation models that utilise

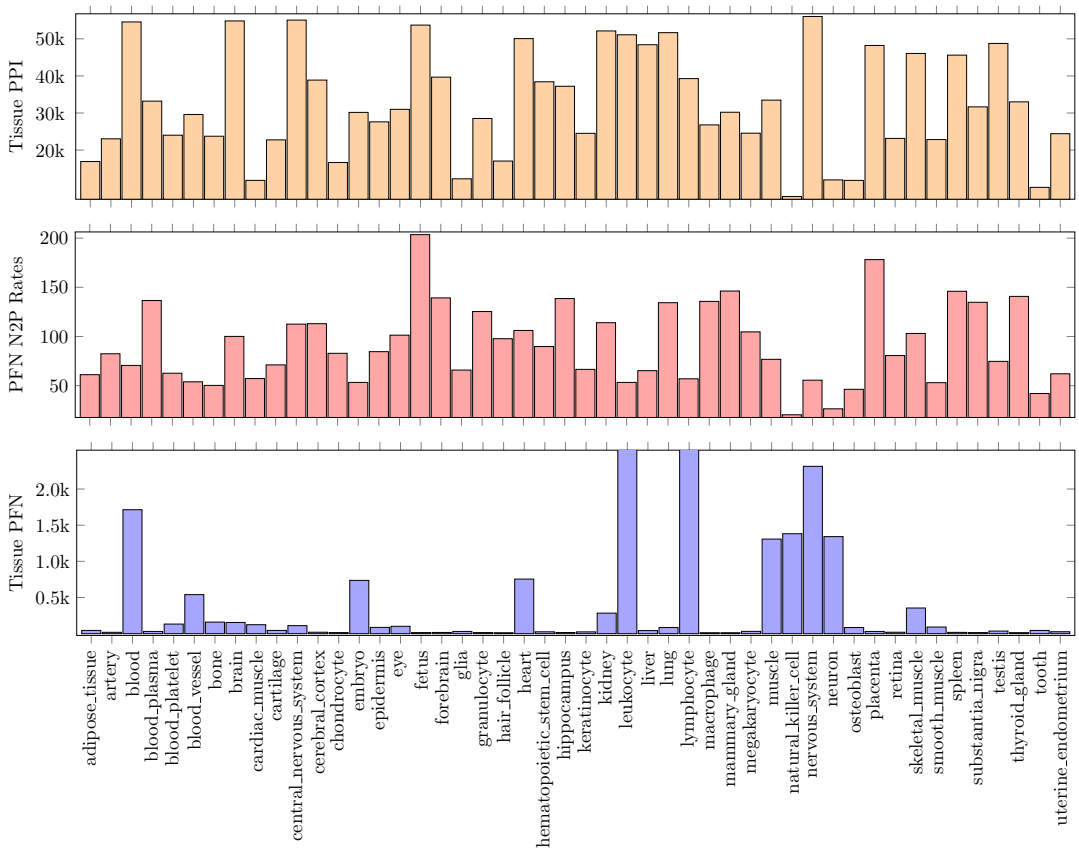

Figure 8.1 – Multiple plots for the number of training instances of protein protein interactions, the negative to positives rates of protein functions for each tissue and protein function links of tissues in the investigated dataset. The set of presented tissues are a subset of all the available tissues that correspond to the list of tissues available in the testing set. PPI refers to protein-protein interactions and PFN refers to protein functions.

the dot and complex products; these models are mainly used to learn efficient representations of elements in graphs, networks and general tensors (Yang et al., Trouillon et al., Lacroix et al. 2015*b*, 2016, 2018). We study both the DistMult (Yang et al. 2015*b*) and ComplEx (Trouillon et al. 2016) models for tensor factorisation. We also explore linear latent translation based tensor completion methods such as the translating embedding model (TransE) (Bordes et al. 2013). These methods model the interaction between the tensor combination as a linear translation between the different combination elements to learn efficient vector representations.

8.3 Materials

In our experiments we used the tissue-specific dataset compiled by Zitnik et. al (Zitnik & Leskovec 2017), where they compiled protein-protein interactions and protein functions of

144 tissue types[1]. The dataset contains information on three different types:

1. *Tissue hierarchy*: the hierarchy of the investigated tissues, where they have used the acyclic graph structure of the BRENDA Tissue Ontology (Gremse et al. 2010) to generate hierarchical relations between tissues.

2. *Tissue-specific protein-protein interactions*: a collection of protein-protein interactions for each of the investigated tissue types which is compiled from different sources (Orchard et al., Rolland & et. al, Chatr-Aryamontri et al. 2013, 2014, 2014). The protein-protein interactions information include 342353 interactions of 21557 proteins across all the investigated tissues.

3. *Tissue-specific protein functions*: a collection of protein tissue-specific functions that covers 48 out of 144 of the investigated tissues. The number of all known protein function associations is 20619, where these associations link proteins to a set of 584 different biological functions such as *odontogenesis*, *regulation of smooth muscle cell migration*, etc. The dataset also contains both negative and positive instances for protein functions for each tissue. The negative to positive ratio, however, is variable for each tissue where the average negative to positive ratio in the dataset is 61:1.

The distribution of both protein-protein interactions, negative to positive ratios and protein functions for each tested tissue is shown in Fig. 8.1. In our experiments, we only used the tissue-specific protein-protein interactions and protein function information. The protein-protein interactions are only used in the training process of our models. We then used the tissue-specific protein function data in two configurations: holdout test and k-fold cross validation. In our holdout test setting, the tissue-specific protein functions were divided between training and testing with a test to train ratio of 1:10 as established by previous works (Zitnik & Leskovec 2017). The division procedure was applied on each tissue alone. First, we compiled all the positive and negative protein function associations into two groups respectively with a random shuffle of instances for each group. We then split both the positive and negative groups into 10 equal splits. Finally, we selected one random split for each group to represent the positive and negative training instances while the rest of the splits represent the training data.

In the k-fold cross validation configuration, we used a 5 fold cross validation which is averaged over 5 runs where we split the data using the same previous approach but with a 1:5 test to train ratio (5 folds). In each fold one split was used as a test split and the others were used for training. This procedure was applied 5 times to learn the average performance of the model.

[1]The dataset is downloaded from: http://snap.stanford.edu/ohmnet/

8.4 Methods

In this section we discuss the use of the TriVec model in our task where we examine its design and its training procedure.

The TriVec model introduced in Chapter 5 is a tensor factorisation based knowledge graph embedding model. It uses a three dimensional tensor to model tissue-specific protein functions, where each (protein x, tissue t, function f) combination is modelled using the tensor combination (i, j, k) such that i, j, k are the corresponding indices of the protein x, tissue t, and function f respectively. These scores are learnt by computing the interactions of embeddings of objects for each combination, where these embeddings are optimised using the general tensor decomposition training procedure discussed in Section 8.2 and Chapter 4. To begin with, the model initialises all object embeddings as random noise. It then applies a multi-phase training procedure to update these embeddings to an optimal state. In this state, the score obtained from interactions of object embeddings of correct combinations should be higher than the scores of all other incorrect combinations.

For example, the training procedure of the known combination "the *ADM* protein performs *regulation of vasoconstriction* in the *blood* tissues" is executed as follows. First, the dataset models such a fact using the combination (*hsa:133*, *blood*, *GO:0019229*), where *hsa:133* is the code of the *ADM* protein and *GO:0019229* is the gene ontology code for the function "*regulation of vasoconstriction*". The TriVec model translates the combination into a numerical combination (i, j, k) which represents the tensor 3D indices of *hsa:133*, *blood*, and *GO:0019229* respectively. The model then generates a set of negative training samples by uniform sampling from all possible proteins and functions. This set is a subset of the set of all possible corruptions of the combination (i, j, k) which is defined as follows:

$$\mathcal{N}(i,j,k) = \bigcup_{k' \in \mathbf{F}} (i,j,k') \quad \cup \quad \bigcup_{i' \in \mathbf{P}} (i',j,k),$$

where $\mathbf{F}$ is the set of indices of all functions, $\mathbf{P}$ is the set of indices of all proteins, i' represents the corresponding index of any random protein and k' represents the corresponding index of any random function. In practice, some of the the sampled corruptions can be true. However, since the number of proteins and functions is very high and the number of sampled corruptions is usually low, the probability of sampling true combinations is insignificant (Bordes et al. 2013) and does not affect the effectiveness of the training procedure. The model then uses both the true combination and its corruptions to define a training objective function that is defined as follows:

$$\mathcal{J} = \sum_{i,j,k} [n \cdot \mathcal{L}(\eta^{\mathbf{M}}(i,j,k)^{+}) + \sum_{d=1}^{n} \mathcal{L}(\eta^{\mathbf{M}}(i,j,k)^{-}_{d})], \tag{8.4}$$

where n denotes the number of negative samples, $\eta^{\mathbf{M}}(i,j,k)^{-}_{d}$ denotes the model score of

the d-th sampled negative corruption, $\eta^{\mathbf{M}}(i,j,k)^{+}$ denotes the score of the true combination (i,j,k), $\mathcal{L}(s,l)$ is the model loss of the score s with its true label l. In this context, 1 and -1 represent the labels for true and corrupted facts respectively. The objective loss is then minimised using the stochastic gradient descent by updating the corresponding embeddings. After iterative training of all known combinations, the model embeddings are updated towards the optimal state such that the computed scores of the true tensor combinations are greater than scores of other random combinations.

In addition to the protein function combinations, the TriVec model also optimises embeddings of protein by training on the tissue-specific protein-protein interaction using the same approach, where both protein functions and interactome are learnt jointly. In the following we discuss the TriVec model embedding interaction function *i.e.* scoring function and its training loss.

8.4.1 The embeddings representation

Tensor decomposition models learn scores for different tensor object combinations by factorising their corresponding embedding representations. Traditionally, these embeddings are modelled using a vector or a matrix. In the RESCAL model, embeddings of entities are modelled using vectors while embeddings of their inter-relations are modelled with matrices. The ComplEx model on the other hand models both entities and relations using only vectors. However, its vectors are complex such that each entity and relation is represented using two vectors: real and imaginary.

In our approach, we model all tensor objects using embedding vectors, where each entity is represented by three real embedding vectors. We then represent different tissue-specific protein functions as combination in a tensor as shown in Fig. 8.2. We then define the score of any tensor combinations of objects as follows:

$$\eta^{\mathbf{M}}(i,j,k) = \sum_{m} [\overset{1}{\mathbf{e}}_{i1} \otimes \overset{2}{\mathbf{e}}_{j1} \otimes \overset{1}{\mathbf{e}}_{k3}] + [\overset{1}{\mathbf{e}}_{i2} \otimes \overset{2}{\mathbf{e}}_{j2} \otimes \overset{1}{\mathbf{e}}_{k2}] + [\overset{1}{\mathbf{e}}_{i3} \otimes \overset{2}{\mathbf{e}}_{j3} \otimes \overset{1}{\mathbf{e}}_{k1}] \tag{8.5}$$

where $\overset{1}{\mathbf{e}}_{i1}$, $\overset{1}{\mathbf{e}}_{i2}$, and $\overset{1}{\mathbf{e}}_{i3}$ represent the first, second, and third embedding vectors of the correspondence to the object of the index i, m denotes the size of all embedding vectors.

8.4.2 Training loss

Tensor decomposition models utilise different ranking loss function techniques for modelling their training objectives. For example, the RESCAL model (Nickel et al. 2011) used square error loss defined in Eq. 4.1. On the other hand, in the TriVec model, we use 1 and -1 to label its true and corrupted combinations respectively. We then adopt the pointwise ranking-based

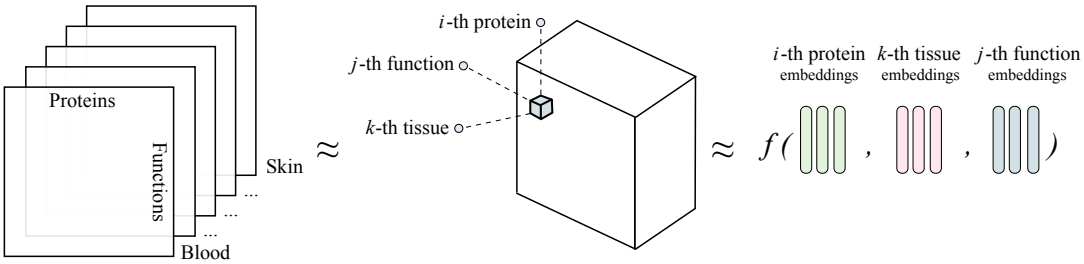

Figure 8.2 – An illustration for the tissue-specific protein functions tensor, where each tissue represents a matrix. Tissue-specific protein function scores are represented by tensor cells, where the cell (i, j, k) represents the score of the i-th protein linked with the j-th function in the k-th tissue.

logistic loss function 4.3 discussed in Chapter 4 which is known to provide efficient training for tensor factorisation models with linear time and space complexity (Trouillon et al. 2016).

8.4.3 Multi-class based training procedure

Lacroix et. al. (Lacroix et al. 2018) showed that a multi-class based training procedure can significantly enhance the predictive accuracy of tensor decomposition models like the ComplEx model. Their approach suggested replacing the corruption sampling procedure by using a 1-vs-all negative log softmax based loss. In this approach, the scores are computed for all possible first and third object corruptions of each of the true combinations. The softmax of the true combination is then maximised. This automatically leads to the minimisation of the corruptions by the nature of the softmax. Let (i, j, k) be a true combination. Then, the objective function of this approach for this combination is defined as follows:

$$\mathcal{J}_{\texttt{mc}} = \mathcal{J}_{\texttt{softmax}}(i, j, k') + \mathcal{J}_{\texttt{softmax}}(i', j, k) + \|(i, j, k)\|_{N3},$$

where $\|(i, j, k)\|_{N3}$ is the tensor nuclear norm (Lacroix et al. 2018) that is defined as

$$\|(i, j, k)\|_{N3} = \frac{\lambda}{3} \sum_{m=1}^{M} \sum_{d=1}^{3} (|\overset{1}{\mathbf{e}}_{id}| + |\overset{2}{\mathbf{e}}_{jd}| + |\overset{1}{\mathbf{e}}_{kd}|).$$

The objective term $\mathcal{J}_{\texttt{softmax}}(i, j, k')$ denotes the negative-log softmax loss of the corruptions of the right hand side element k', which is defined as follows:

$$\begin{aligned}\mathcal{J}_{\texttt{softmax}}(i, j, k') &= -\log\left(\frac{exp(\eta^{\mathbf{M}}_{\{i,j,k\}})}{\sum_{k' \in E} exp(\eta^{\mathbf{M}}_{\{i,j,k'\}})}\right) \\ &= -\eta^{\mathbf{M}}(i, j, k) + \log\left(\sum_{k'} \eta^{\mathbf{M}}(i, j, k')\right)\end{aligned}.$$

The objective term $\mathcal{J}_{\texttt{softmax}}(i', j, k)$ is defined similarly, where it models the loss of the left

hand side element corruptions i'. The final objective loss for all combinations is then defined as follows:

$$\begin{aligned}\mathcal{J}_{\mathtt{mc}} = \sum_{i,j,k} [-2 \cdot \eta^{\mathbf{M}}(i,j,k) + \log(\sum_{i'} \eta^{\mathbf{M}}(i',j,k)) + \log(\sum_{k'} \eta^{\mathbf{M}}(i,j,k')) \\ + \frac{\lambda}{3} \sum_{m=1}^{M} \sum_{d=1}^{3} (|\overset{1}{\mathbf{e}}_{id}| + |\overset{2}{\mathbf{e}}_{jd}| + |\overset{1}{\mathbf{e}}_{kd}|)],\end{aligned} \tag{8.6}$$

where the term $\frac{\lambda}{3} \sum_{m=1}^{M} \sum_{d=1}^{3} (|\overset{1}{\mathbf{e}}_{id}| + |\overset{2}{\mathbf{e}}_{jd}| + |\overset{1}{\mathbf{e}}_{kd}|)$, the nuclear 3 norm, is a regularisation term where λ is a configurable weight, i' and k' represent all possible corruptions of entity objects.

This approach is known to provide state-of-the-art results in Link Prediction using tensor decomposition due to utilising the whole vocabulary as negatives in the softmax procedure. (Lacroix et al. 2018). It also provides exponentially normalised scores that help the model enlarge the score margins between true and positive combinations.

Despite the enhancements reported by Lacroix et. al. (Lacroix et al. 2018) for this approach, it is considered a quadratic space complexity procedure. Therefore, it can have scalability issues especially on large volumes of data. We have experimented our model with the multi-class loss objective and our results have shown that the traditional ranking objective procedures provide best results in terms of both area under the ROC and precision recall curves. More details about these results are presented in Section 8.6.4.

8.5 Experiments

In this section we discuss the design details of the experimental data, the model training pipeline, and the evaluation protocol.

8.5.1 Experimental setup

In the experiments, we evaluated the state-of-the-art tensor decomposition models including the RESCAL (Nickel et al. 2011), DistMult (Yang et al. 2015*b*), and ComplEx (Trouillon et al. 2016) models along with the OhmNet model and other protein function prediction methods investigated by Zitnik et. al., and we compared them to our proposed model.

A grid search was performed to obtain best hyper parameters for each model in our experiments, where the set of investigated parameters are: embeddings size $K \in \{50, 100, 150, 200\}$, margin $m \in \{1, 2, 3, 4, 5\}$ for the DistMult model, and the number of negative samples $n \in \{2, 4, 6, 10\}$. All the embedding vectors of our models were initialised using the uniform Xavier random initializer (Glorot & Bengio 2010). For all the experiments, we used batches of size

5000, with a maximum of 1000 training iterations *i.e.* epochs. The gradient update procedure is performed using the Adagrad optimiser with a fixed learning rate lr = 0.1.

8.5.2 Evaluation protocol

We evaluated the experimented models on a set of 48 tissues using both the area under the roc (AUC-ROC) and precision recall (AUC-PR) curves as established by previous works. The number of the true known tissue-specific protein function testing instances represented 10% of all known protein function for each tissue. The negative to positive ratio is variable in different tissues, where the average negative to positive ratio is 61:1 (cf. Figure 8.1).

8.5.3 Implementation

We used the Tensorflow framework (GPU) along with Python 3.5 to perform our experiments. All experiments were executed on a Linux machine with processor Intel(R) Core(TM) i70.4790K CPU @ 4.00GHz, 32 GB RAM, and an nVidia Titan Xp GPU.

8.6 Results

In this section we present the outcome results of our experiments. We compare our proposed TriVec model to other state-of-the-art tissue-specific protein function prediction models in term of the area under the ROC and precision recall curves. We also compare different strategies for modelling the training objective for our model and other tensor factorisation models.

8.6.1 Comparison with the state-of-the-art using holdout test

Table 8.1 provides a comparison between the experimental results of the TriVec model and other state-of-the-art-models in terms of both the area under ROC and precision recall curves in a holdout test setting. The results show that the TriVec model achieves scores of 0.858 and 0.442 in terms of the area under ROC and precision recall curves respectively. This shows that the TriVec model outperforms the state-of-the-art model, the OhmNet model, with a margin that is 13% and 32% higher in terms of the area under ROC and precision recall curves respectively. The results also show that the TriVec model outperforms the network tissue-specific SVM model with better scores of 22% and 57% in terms of the area under ROC and precision recall curves respectively. The TriVec model also provides a better area under the ROC curve score with margins of 27%, 34%, 29%, 32%, 23%, and 26% for the Minimum Linear Embedding, Induced LINE, Collapsed LINE, Induced Node2Vec, Collapsed Node2Vec, and GeneMania models respectively. The results also show that the TriVec model outperforms the same models in terms of the area under the precision recall curve with a ratio of 25%, 26%, 27%, 28%, 30%, and 28% respectably.

Model	AUC-ROC	AUC-PR
RESCAL Tensor Decomposition ⋆	0.674	0.235
Minimum Curvilinear Embedding ⋆	0.674	0.248
Induced LINE ⋆	0.642	0.261
Collapsed LINE ⋆	0.663	0.271
Induced Node2Vec ⋆	0.649	0.283
Collapsed Node2Vec ⋆	0.697	0.298
GeneMania ⋆	0.683	0.281
Network Tissue-Specific SVM ⋆	0.701	0.281
Tissue-Specific Network Propagation ⋆	0.675	0.265
OhmNet ⋆	0.756	0.336
TransE	0.796	0.230
DistMult - Neg Log Softmax	0.796	0.230
DistMult - SE Loss	0.808	0.219
DistMult - Log Loss	0.629	0.055
ComplEx - Neg Log Softmax	0.835	0.344
ComplEx - SE Loss	0.857	0.337
ComplEx - Log Loss	**0.863**	0.411
TriVec - Neg Log Softmax (ours)	0.826	0.402
TriVec - SE (ours)	0.857	0.337
TriVec - Log Loss (ours)	0.858	**0.442**

Table 8.1 – Summary of results for the holdout test experiments of the TriVec model compared to other state-of-the-art models in terms of area under the ROC and precision recall curves. The notion ⋆ represents the results which are obtained from (Zitnik & Leskovec 2017)

In comparison to other tensor decomposition models, the results show that the TriVec model outperforms the RESCAL tensor decomposition model with a rate of 27% and 88% in terms of the area under the ROC and precision recall curves respectively. The results also show that the TriVec model outperforms the ComplEx model with a rate of 6% and 10% in terms of the area under the ROC and precision recall curves respectively.

8.6.2 Cross validation test results

Figure 8.3 shows the results of the k-fold cross validation test. The results show that the TriVec model outperformed all other models in terms of both the area under the precision recall and ROC curves. The results also show that the TriVec model achieved a 85% and 39% accuracy in terms of both the area under the precision recall and ROC curves respectively. These scores outperform the ComplEx model scores with margins of 3% and 5% in terms of the AUC-ROC and AUC-PR respectively. Similarly, the TriVec model scores outperform the scores of the DistMult model (80%, 16%) with a margin of 5% and 24% in terms of the AUC-ROC and AUC-PR respectively. The results also show that the random baseline model achieves 50% and

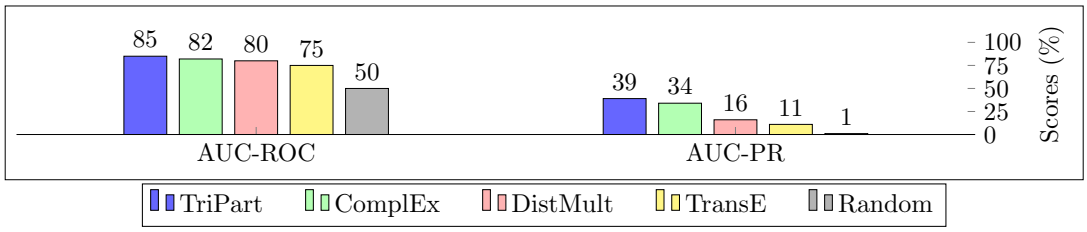

Figure 8.3 – Summary of the area under the ROC and precision recall curve scores of the TriVec model compared to other tensor completion models in the 5-fold cross validation averaged over 5 runs.

	Tissue	TriVec	OhmNet	ComplEx	DistMult	TransE
1	Natural Killer Cell	0.916	0.834	**0.923**	0.893	0.867
2	Placenta	**0.989**	0.830	0.938	0.825	0.754
3	Spleen	0.510	**0.803**	0.611	0.370	0.257
4	Liver	0.676	**0.803**	0.541	0.645	0.637
5	Forebrain	**0.983**	0.796	0.862	0.915	0.821
6	Macrophage	0.605	0.789	**0.939**	0.650	0.672
7	Epidermis	**0.880**	0.785	0.788	0.568	0.647
8	Hematopoietic Stem Cell	0.783	0.784	**0.861**	0.838	0.720
9	Blood Plasma	**1.000**	0.784	0.990	0.987	0.986
10	Smooth Muscle	**1.000**	0.778	0.997	0.953	0.910
	Average	0.834	0.799	**0.845**	0.764	0.727

Table 8.2 – A comparison of the area under the ROC curve scores of the TriVec and OhmNet models on the top ten accurately predicted tissues by the OhmNet model.

1% scores in terms of the AUC-ROC and AUC-PR respectively.

8.6.3 A detailed comparison with other models

Table 8.2 presents the area under the ROC curve scores of the TriVec model compared to other studied methods for a selected set of 10 tissues[2]. The selected tissues represent the tissues where the OhmNet model achieved the best scores in terms of the area under the ROC curve. The results show that the TriVec model achieves an average area under the ROC curve score of 0.834, and it outperforms the average score achieved by the OhmNet model which achieved an average score of 0.799. However, The ComplEx model achieved the best average AUC-ROC scores with an average of 0.845. The results also show that the TriVec model achieved the best AUC-ROC scores in 5 out of 10 investigated tissues. On the other hand, the OhmNet and ComplEx models achieved the best scores on the other five tissues. The results also show that the TriVec model achieved perfect area under the ROC curve scores for 2 out of 10 of the investigated tissues.

[2]We have selected the set of 48 testing tissues compiled by Zitnik et. al. (Zitnik & Leskovec 2017) to be able to compare then to their reported scores.

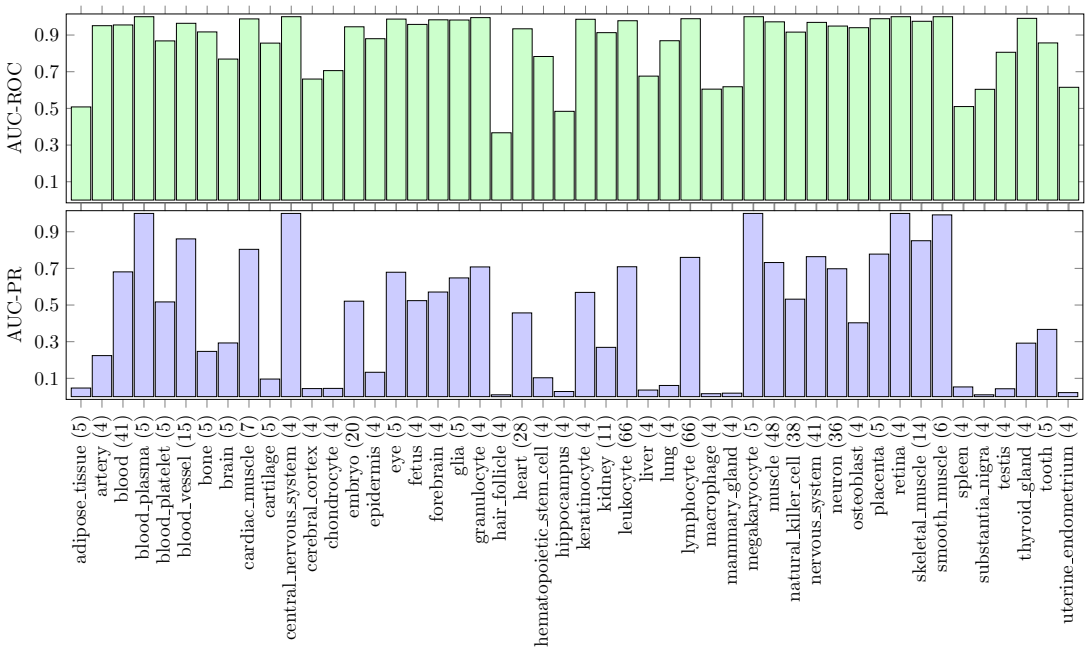

Figure 8.4 – Summary of the area under the ROC and precision recall curve scores of the TriVec model on the investigated 48 tissues. The number numbers next to the tissues represent the number of true known protein functions testing instances for each tissue.

Fig. 8.4 presents the area under the ROC and precision recall curves scores of the TriVec model for all the 48 investigated test tissues. It also includes the number of true known protein function instances for each tissue. The figure shows that the scores have a high variance between the different tissues. Further discussions regarding the relation between the size of the training data and the outcome scores is included in Sec. 8.7.1.

8.6.4 Optimal training objective

The results in Table 8.1 also presents a comparison between the ComplEx and TriVec tensor decomposition models with different training loss functions in terms of the area under the ROC and precision recall curves. The results show that the training objective functions have an effect on the scores of both models where the best area under the ROC curve score is achieved by the ComplEx model with square error loss and the best area under the precision recall curve score is achieved by the TriVec model with the logistic loss. The results also show that the TriVec model achieves the best scores for each objective loss in 5 out of 6 combinations (score type/configuration combinations). The results also show that the scores associated with standard ranking training objectives outperform the multi-class negative logistic softmax loss in both the ComplEx and TriVec model in terms of both the area under the ROC and precision recall curves.

8.7 Discussion

In this section, we provide a discussion on the outcome results of our model where we analyse patterns found in these outcome results. We then discuss the efficiency and scalability of the explored models in this chapter. We also explore the benefits and limitations of our approach compared to other state-of-the-art approaches. Finally, we discuss the impact of our study on biological research.

8.7.1 Analysis of the evaluation results

The results shown in Figure 8.4 show that the TriVec model achieves high scores in term of both the AUC-ROC and AUC-PR metrics. However, it also shows that there is an apparent variance of the outcome scores specifically in the case of the area under the precision recall curve. We have thus performed extra analyses on the outcome results to have a better understanding of this variance. In this regard, we studied the effects of the size of both training and test data on the outcome metric scores for both the area under the ROC and precision recall curves.

Figure 8.5 shows a matrix plot of five tissue-specific data features for the TriVec models results: the area under the ROC curve, the area under the precision recall curve, the negative to positive rate, the true training data size and the true testing data size. Each point in the plot represents information corresponding to a specific tissue where all the data points were initially labelled with blue. We have then applied red labelling to tissues that have the lowest scores of 16% or lower in terms of the AUC-ROC. The histogram of the different features shows that the lowest scores correspond to the tissues with the lowest training and testing data sizes. It also shows that they also correspond to the lowest scores in terms of the area under the ROC curve. Other comparative scatter plots that compare combinations of pairs of features also show a positive correlation between the data training and testing sizes and the outcome metric scores in terms of both the area under the ROC and precision recall curves. The plot also shows that there is no correlation between the different negative to positive ratios and the outcome metric scores, where the lowest scored tissues have a wide range of evenly distributed negative to positive ratios.

8.7.2 Efficiency and scalability

The tensor factorisation models generally have different time and space complexities. For example, the RESCAL tensor factorisation model (Nickel et al. 2011) has a quadratic $\mathcal{O}(K^2)$ time and space complexity, where K is the size of the vector representation (Nickel et al. 2011). On the contrary, other methods such as the DistMult, ComplEx and TriVec models have a linear time and space complexity $\mathcal{O}(K)$ (Yang et al., Trouillon et al. 2015*b*, 2016). Therefore, they are capable of producing more efficient and scalable predictions. The implementation of tensor factorisation methods is also easily portable to GPUs; they therefore, benefit from the efficiency and scalability of their architectures.

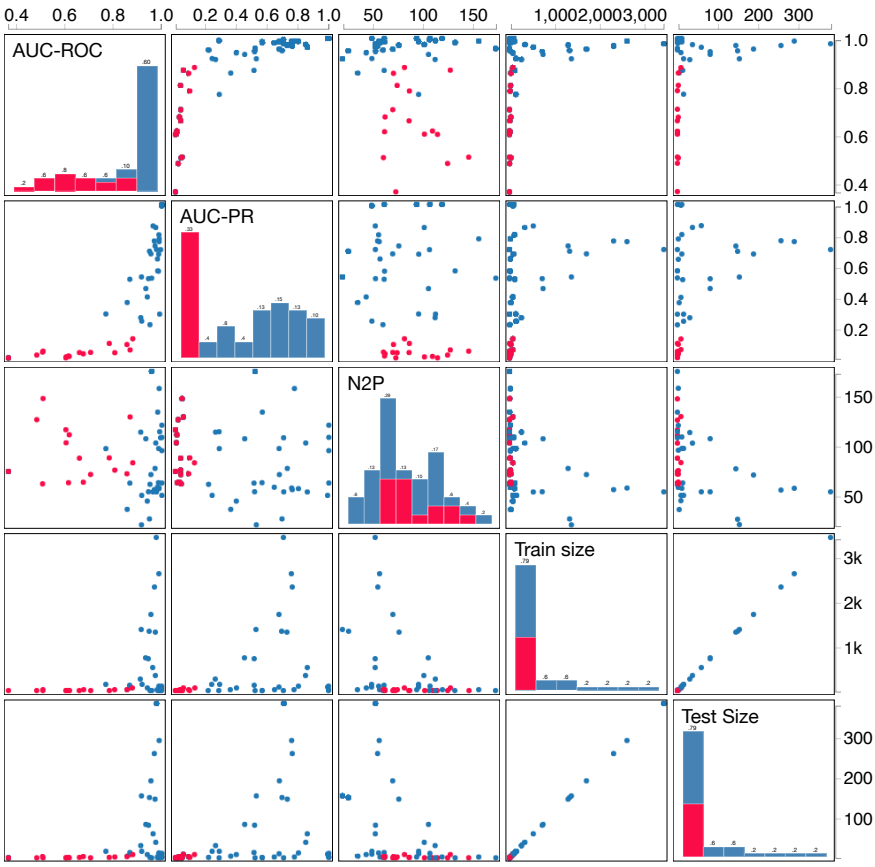

Figure 8.5 – Matrix plot of the model's metric values compared to the training and testing data sizes for each tissue. Red labelled instances represent tissues with the lowest AUC-PR scores. N2P denotes the negative to positive ratio of the testing data. The plot is generated using the data visualisation platform (DVP) software (Yousef et al. 2019) .

In context of protein function predictions, the OhmNet model works in a multi-phase procedure where it builds a network of protein interactions and protein functions for each tissue. It then learns embeddings of proteins and their functions within each tissue and applies a hierarchical propagation routine to merge embeddings of different tissues (Zitnik & Leskovec 2017). Despite the ability to port the implementation of the embedding learning phase of this procedure to GPUs, other phases are dependent on CPU. This results from the dependence on path searches for generating initial features for proteins and functions in the tissue-specific network.

Figure 8.6 shows a comparison between the TriVec model and other models used in this chapter in terms of the runtime required to learn embeddings of proteins within a set of specified tissues [3]. The comparison shows that the tensor factorisation methods have different runtime values dependent on their training objective function. Figure 8.6 also shows that the

[3]We have used the set of brain sub tissues publicly available at:https://github.com/marinkaz/ohmnet. All experiments are also executed on CPU only for fair comparison.

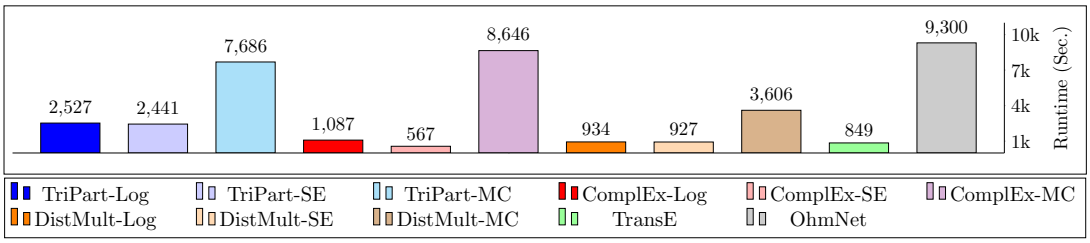

Figure 8.6 – A comparison of the runtime of the TriVec model compared to other models in learning the embedding of protein interactome in the brain related tissues.

TriVec model requires significantly less time than the OhmNet model. It, however, has higher runtime compared to other tensor completion techniques.

8.7.3 Limitations

The outcomes of our experiments show that our approach outperforms other approaches in terms of predictive accuracy. Our approach however, requires more computational time compared to other tensor completion methods as shown in Figure 8.6. The outcome vector representations of the OhmNet, TransE, DistMult models are single vectors with real values. They can therefore be easily consumed by different learning and analytical techniques such as embedding visualisation, clustering, classification, etc. Our approach and the ComplEx model, on the other hand, provide multi-part and complex embedding vectors respectively. They are therefore not consumable by most of the current embedding processing mechanisms.

8.7.4 Implications on biological research

Generally, the use of computational approaches in predicting protein functions is useful as they are free from human bias, and are therefore not influenced by prior knowledge and opinions-unlike laboratory-based methods. These approaches also bypass the need to spend a long amount of time in the laboratory and can be used to provide guidance on the direction of research within the laboratory, therefore saving both time and money. Follow-up experiments can then be carried out in the laboratory for confirmation of the computer predictions. The development of computational tissue-specific protein functions predictors are however developing slowly due to limited available supporting data (Greene et al., Zitnik & Leskovec 2015, 2017). This study hereof provides an incremental step towards building more efficient computational predictors which can provide even more scalable and accurate predictions with the currently limited available data.

We do not believe that computational methods can replace laboratory experiments in the context of our study. However it can significantly speed up laboratory experimentation by suggesting potential protein associated functions. The accuracy of prediction is therefore crucial for ensuring meaningful and beneficial suggestions. We have shown in our study that

the TriVec model outperforms other computational methods in predicting tissue-specific protein functions; it therefore, provides the lowest rates of false positives. This enhancement can thus enhance the quality of the predictions supplied to biologists. It also enables more accurate suggestions of potential tissue-specific protein related functions.

Conclusions Part IV

9 Conclusions and Future work

In this chapter, we discuss the learnt lessons and conclusions of our studies and we discuss the future directions of our work. First, we discuss the current state of the research into graph feature based methods. We also discuss knowledge graph embedding models and the challenges associated with them, and our intended future activities to extend the capabilities of embedding based approaches. We finally discuss our future direction for applying knowledge completion methods to biological use cases.

9.1 Summary

In this thesis, we investigated the problem of knowledge completion using models that rely on both graph features and embeddings. We have also presented a set of use cases for the use knowledge graph embedding models in modelling complex biological systems.

Firstly, we discussed graph feature models and their current limitations in Chapter 3, and we proposed a new graph feature model, the DSP model (Mohamed et al. 2018), which outperformed the currently available graph feature model in term of the predictive accuracy with no extra added computational cost.

Secondly, we investigated the knowledge graph embedding models which are known to achieve state-of-the-art predictive accuracy in the task of link prediction with linear time and space complexity (Trouillon et al. 2016). While these models witnessed rapid updates and developments in the recent years, these updates were mainly limited to one part of their training pipeline, the embedding interaction modelling techniques *i.e.* scoring functions (Nickel, Murphy, Tresp & Gabrilovich, Wang et al. 2016*b*, 2017). In Chapter 4, we discussed other parts of the knowledge graph embedding models such as training objectives, negative sampling techniques and hyperparameters tuning process. We showed that the choice of the training objective and negative sampling have a significant effect on the models' accuracy and scalability. We also showed that knowledge graph embedding models are sensitive to specific hyperparameters which can significantly affect their predictive accuracy.

Thirdly, in Chapter 5, we proposed a new tensor factorisation based knowledge graph embedding model, the TriVec model (Mohamed & Novácek 2019), which models embeddings using multiple vectors. Our model used these multi-vector embedding to compute scores for knowledge graph triplets using a combination of symmetric an asymmetric procedures which allowed encoding both symmetric and asymmetric relations in knowledge graphs. We showed using experimental evaluation on standard benchmarks that our newly proposed model outperforms other state-of-the-art methods in terms of the predictive accuracy on all of the standard benchmarking datasets.

Fourthly, we discussed the use of knowledge graph embedding models in different biological applications in Chapter 6. First, we discussed the evolution of network based methods to model and analyse complex biological systems. We then demonstrated the different predictive and analytical capabilities of knowledge graph embedding models in modelling biological systems. We also executed various experiments to validate these capabilities where we demonstrated the use of knowledge graph embedding models in tasks such as predicting links between biological concepts, measure similarity between biological entities and entity clustering based on the embedding vectors. We also discussed the potential uses and applications of knowledge graph embedding models in the biological domain and their associated risks and limitations.

Finally, in Chapters 7,**??**,8, we introduced the use of knowledge graph embedding models in three different biological use cases: predicting drug targets, predicting polypharmacy side-effects and predicting tissue-specific protein functions respectively. In these chapters, we discussed each use case individually where we investigated previous state-of-the-art predictive techniques and showed their limitation. We also executed computational experiments where we showed that knowledge graph embedding models achieve the best predictive accuracy in terms of the area under the ROC and precision recall curves in all of the three applications. These experiments also showed that our proposed knowledge graph embedding model, the TriVec model, achieved the best predictive accuracy in all the use cases compared to all other models including other knowledge graph embedding models.

9.2 Current State of Knowledge Completion Models

In Chapter 3, we have discussed the state-of-the-art graph feature models such as the PRA and SFE models. We have also shown that these models have limited feature representation of node pairs in knowledge graph as they only depend on connecting paths between nodes as features. We have then proposed a new approach which uses a combination of connecting paths and subgraph paths to model node pairs. We have shown by experimental evaluation that our newly proposed approach outperforms the state-of-the-art approaches in terms of the mean average precision and mean reciprocal rank in knowledge based completion task on a NELL based benchmarking dataset. We have also shown that this enhancement in terms of the predictive accuracy is achieve with equivalent training time complexity to the SFE model.

We have also discussed the interpretability of graph feature models where we shown examples of interpreting model predictions using its input features. In this case, graph feature models use the weights of features learnt during the training process to represent the importance of each feature in the prediction process.

Despite the high interpretability of these models, their dependence on path features affects their scalability as path extraction process is complex and time-consuming. For example, the process of extracting path feature of node pairs on large scale knowledge graph can sometimes be infeasible due to dense node connections. The process of extracting deep path through nested path navigation is also time-consuming and significantly increases the training time of graph feature models. On the other hand, knowledge graph embedding models have linear time and space complexity which makes them superior in terms of scalability. They also excel in the task of link prediction and knowledge graph completion (Toutanova & Chen, Mohamed et al., Mohamed & Novácek 2015, 2018, 2019) where they significantly outperform graph feature models in terms of both scalability and accuracy.

9.3 Towards Explainable Knowledge Graph Embeddings

Despite the accuracy and scalability of knowledge graph embedding models, they have limited interpretability where they operate as black boxes which generate predictive scores that are difficult to relate to original knowledge. This limitation can affect the trust in the predictions of these models, especially in critical domains such as medical informatics research (Holzinger et al. 2019).

This encouraged multiple approaches for enhancing the scalability of graph embeddings using constraining training with a set of predefined rules such as type constraints (Krompass et al. 2015), basic relation axioms (Minervini et al. 2017*b*), etc. These approaches thus enforce the KGE models to learn embeddings that can be partially interpretable by their employed constraints.

In recent studies, researchers have also explored the interpretability of KGE models through new predictive approaches on top of the KGE models. For example, Gusmão et. al. (Gusmão et al. 2018) suggested the use of pedagogical approaches where they have used an alternative graphical predictive model, the SFE model (Gardner & Mitchell 2015), to link the learnt graph embeddings to the original knowledge graph. This approach was able to provide a new way for finding links between the embeddings and the original knowledge; however, the outcomes of these methods are still limited by the expressibility and feature coverage of the newly employed predictive models. The interpreting method in this context, also depends on graph traversal methods which have limited scalability on large knowledge graphs (Mohamed et al. 2018).

9.4 Modelling Evolving Systems

Our knowledge of the surrounding systems evolves everyday. For example, biological systems are rapidly evolving where new chemicals and drugs are introduced and different associations between biological entities are discovered. However, in this context, KGE models are unable to encode the newly introduced entities. This results from their dependence on prior knowledge which does not include the newly discovered entities. The addition of new entities also require full training of the model to learn useful embeddings which is infeasible in rapidly evolving systems with large volumes of data.

In future works, we intend to investigate the use of local retraining to allow learning knowledge graph embeddings for the new entities without retraining the models on the whole knowledge graph. We aim to achieve this by using a sample of neighbour entities in the graph for the newly introduced entities. For example, when adding new entities of locations to a general knowledge graph such as YAGO or DBpedia, the embeddings of these entities can be retrained with other associated locations and concepts only.

In the case of biological systems, this issue can be also addressed by combining knowledge graph embedding scoring procedure with other sequence and structured based scoring mechanisms. This can allow informed prediction on new unknown objects. For example, the protein sequences and chemical structures can be transformed into raw data sequences which can be transformed into embedding using convolutional filters or memory models *e.g.* LSTM. These embeddings can then be used along with knowledge graph embeddings in a unified neural network model to enrich encoded features of under-studied chemicals and proteins. However, such a strategy will affect the scalability of predictions due to the newly introduced components and modules for processing sequence and structure based features.

9.5 Modelling Sparse Biological Networks

Knowledge graph embedding models have proven to be an effective method for modelling complex biological network and predicting new links between biological entities (Mohamed, Nováček & Nounu, Mohamed 2019, 2020). However, the accuracy of KGE models prediction is dependent on the quality of input data where they generate the embedding representations of biological entities according to their prior knowledge. Therefore, the quality and coverage of this knowledge affects the quality of the generated embeddings. For example, there is a high variance in the available prior knowledge on proteins where well studied proteins have significantly higher coverage in most databases (The Uniprot Consortium 2015). This has a significant impact on the quality of the less represented proteins as KGE models will be biased towards the more studied (highly covered) proteins.

In future works, we intend to investigate the use of extra resource about under-represented biological entities to enrich the models knowledge about them. We specifically want to investigate the use of PubMed articles abstracts which mention under-represented entities to

extract new facts about these entities. We can then add these facts to the currently available knowledge graphs to enhance their coverage, therefore, enhance the predictive accuracy of the knowledge graph embedding models operating on them.

Bibliography

Abdelaziz, I., Fokoue, A., Hassanzadeh, O., Zhang, P. & Sadoghi, M. (2017), 'Large-scale structural and textual similarity-based mining of knowledge graph to predict drug-drug interactions', *J. Web Semant.* **44**, 104–117.

Albert, R. (2005), 'Scale-free networks in cell biology.', *Journal of cell science* **118 Pt 21**, 4947–57.

Alshahrani, M., Khan, M. A., Maddouri, O., Kinjo, A. R., Queralt-Rosinach, N. & Hoehndorf, R. (2017), Neuro-symbolic representation learning on biological knowledge graphs, *in* 'Bioinformatics'.

Amrouch, S. & Mostefai, S. (2012), 'Survey on the literature of ontology mapping, alignment and merging', *2012 International Conference on Information Technology and e-Services* pp. 1–5.

Aronson, A. R., Mork, J. G., Gay, C. W., Humphrey, S. M. & Rogers, W. J. (2004), 'The nlm indexing initiative's medical text indexer', *Studies in health technology and informatics* **107 Pt 1**, 268–72.

Bamshad, M. J., Ng, S. B., Bigham, A. W., Tabor, H. K., Emond, M. J., Nickerson, D. A. & Shendure, J. (2011), 'Exome sequencing as a tool for mendelian disease gene discovery', *Nature Reviews Genetics* **12**, 745–755.

Barabási, A.-L. & Oltvai, Z. N. (2004), 'Network biology: understanding the cell's functional organization', *Nature Reviews Genetics* **5**, 101–113.

Bateman, A., Coin, L. J. M., Durbin, R., Finn, R. D., Hollich, V., Griffiths-Jones, S., Khanna, A., Marshall, M., Moxon, S., Sonnhammer, E. L. L., Studholme, D. J., Yeats, C. & Eddy, S. R. (2000), 'The pfam protein families database', *Nucleic acids research* **28 1**, 263–6.

Bauer-Mehren, A., Bundschus, M., Rautschka, M., Mayer, M. A., Sanz, F. & Furlong, L. I. (2011), Gene-disease network analysis reveals functional modules in mendelian, complex and environmental diseases, *in* 'PloS one'.

Berg, H. v. d. (1993), Knowledge graphs and logics; one of two kinds, PhD thesis, University of Twente.

Bizer, C. & Cyganiak, R. (2006), D2r server-publishing relational databases on the semantic web, *in* 'Poster at the 5th international semantic web conference', Vol. 175.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T. & Taylor, J. (2008), Freebase: A collaboratively

created graph database for structuring human knowledge, *in* 'Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data', SIGMOD '08, ACM, New York, NY, USA, pp. 1247–1250.

Bordes, A., Glorot, X., Weston, J. & Bengio, Y. (2014), 'A semantic matching energy function for learning with multi-relational data - application to word-sense disambiguation', *Machine Learning* **94**(2), 233–259.

Bordes, A., Usunier, N., García-Durán, A., Weston, J. & Yakhnenko, O. (2013), Translating embeddings for modeling multi-relational data, *in* 'NIPS', pp. 2787–2795.

Bordes, A., Weston, J., Collobert, R. & Bengio, Y. (2011), Learning structured embeddings of knowledge bases, *in* 'Proceedings of the Twenty-Fifth AAAI 2011, San Francisco, California, USA, August 7-11, 2011'.

Bouchard, G., Singh, S. & Trouillon, T. (2015), On approximate reasoning capabilities of low-rank vector spaces, *in* 'AAAI Spring Syposium on Knowledge Representation and Reasoning (KRR): Integrating Symbolic and Neural Approaches', AAAI Press.

Bowes, J., Brown, A. J., Hamon, J., Jarolimek, W., Sridhar, A., Waldron, G. & Whitebread, S. (2012), 'Reducing safety-related drug attrition: the use of in vitro pharmacological profiling', *Nature reviews Drug discovery* **11**(12), 909.

Brachman, R. J. & Levesque, H. J. (2004), *Knowledge Representation and Reasoning*, Elsevier.

Buono, N. D. & Pio, G. (2015), 'Non-negative matrix tri-factorization for co-clustering: An analysis of the block matrix', *Inf. Sci.* **301**, 13–26.

Cai, J. J. & Petrov, D. A. (2010), 'Relaxed purifying selection and possibly high rate of adaptation in primate lineage-specific genes', *Genome biology and evolution* **2**, 393–409.

Cannistraci, C. V., Alanis-Lobato, G. & Ravasi, T. (2013), Minimum curvilinearity to enhance topological prediction of protein interactions by network embedding, *in* 'Bioinformatics'.

Cao, Z., Qin, T., Liu, T., Tsai, M. & Li, H. (2007), Learning to rank: from pairwise approach to listwise approach, *in* 'ICML', Vol. 227 of *ACM International Conference Proceeding Series*, ACM, pp. 129–136.

Chatr-Aryamontri, A., Breitkreutz, B.-J., Oughtred, R., Boucher, L., Heinicke, S., Chen, D., Stark, C., Breitkreutz, A., Kolas, N., O'donnell, L. et al. (2014), 'The biogrid interaction database: 2015 update', *Nucleic acids research* **43**(D1), D470–D478.

Chen, W., Liu, T., Lan, Y., Ma, Z. & Li, H. (2009), Ranking measures and loss functions in learning to rank, *in* 'NIPS', Curran Associates, Inc., pp. 315–323.

Chen, X., Ji, Z. L. & Chen, Y. Z. (2002), 'Ttd: therapeutic target database', *Nucleic acids research* **30**(1), 412–415.

Cheng, F., Liu, C., Jiang, J., Lu, W., Li, W., Liu, G., Zhou, W.-X., Huang, J. & Tang, Y. (2012), Prediction of drug-target interactions and drug repositioning via network-based inference, *in* 'PLoS Computational Biology'.

Cheng, F., Zhou, Y., Li, W., Liu, G. & Tang, Y. (2012), Prediction of chemical-protein interactions

network with weighted network-based inference method, *in* 'PloS one'.

Cheung, T.-Y. (1983), 'Graph traversal techniques and the maximum flow problem in distributed computation', *IEEE Transactions on Software Engineering* (4), 504–512.

Clark, W. T. & Radivojac, P. (2011), 'Analysis of protein function and its prediction from amino acid sequence.', *Proteins* **79 7**, 2086–96.

Cohen, J. D. & Servan-Schreiber, D. (1992), 'Context, cortex, and dopamine: a connectionist approach to behavior and biology in schizophrenia.', *Psychological review* **99 1**, 45–77.

Consortium, G. O. (2003), The gene ontology (go) database and informatics resource gene ontology consortium.

Consortium, T. G. O. (2019), The gene ontology resource: 20 years and still going strong, *in* 'Nucleic Acids Research'.

Consortium, T. U. (2017), Uniprot: the universal protein knowledgebase, *in* 'Nucleic Acids Research'.

Corbett, A., Pickett, J., Burns, A., Corcoran, J., Dunnett, S. B., Edison, P., Hagan, J. J., Holmes, C., Jones, E., Katona, C. et al. (2012), 'Drug repositioning for alzheimer's disease', *Nature Reviews Drug Discovery* **11**(11), 833.

Cossock, D. & Zhang, T. (2008), 'Statistical analysis of bayes optimal subset ranking', *IEEE Trans. Information Theory* **54**(11), 5140–5154.

D D'Agati, V. (2008), 'The spectrum of focal segmental glomerulosclerosis: new insights', *Current opinion in nephrology and hypertension* **17**(3), 271–281.

Davis, J. & Goadrich, M. (2006), The relationship between precision-recall and roc curves, *in* 'Proceedings of the 23rd international conference on Machine learning', ACM, pp. 233–240.

Davis, R., Shrobe, H. E. & Szolovits, P. (1993), 'What is a knowledge representation?', *AI Magazine* **14**(1), 17–33.

Dettmers, T., Minervini, P., Stenetorp, P. & Riedel, S. (2018), Convolutional 2d knowledge graph embeddings, *in* 'AAAI', AAAI Press, pp. 1811–1818.

Drews, J. (2000), 'Drug discovery: A historical perspective', *Science* **287**(5460), 1960–1964.

Dumontier, M., Callahan, A., Cruz-Toledo, J., Ansell, P., Emonet, V., Belleau, F. & Droit, A. (2014), Bio2rdf release 3: A larger, more connected network of linked data for the life sciences, *in* 'Proceedings of the ISWC 2014', pp. 401–404.

Enault, F., Suhre, K. & Claverie, J.-M. (2005), 'Phydbac "gene function predictor" : a gene annotation tool based on genomic context analysis', *BMC Bioinformatics* **6**, 247–247.

et. al., A. F. (2016), 'The reactome pathway knowledgebase', *Nucleic acids research* **44 D1**, D481–7.

et. al., S. E. O. (2014), The mintact project—intact as a common curation platform for 11 molecular interaction databases, *in* 'Nucleic Acids Research'.

Fagerberg, L., Hallström, B. M., Oksvold, P., Kampf, C., Djureinovic, D., Odeberg, J., Habuka,

M., Tahmasebpoor, S., Danielsson, A., Edlund, K. et al. (2014), 'Analysis of the human tissue-specific expression by genome-wide integration of transcriptomics and antibody-based proteomics', *Molecular & Cellular Proteomics* **13**(2), 397–406.

Färber, M., Bartscherer, F., Menne, C. & Rettinger, A. (2017), 'Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago', *Semantic Web* **9**, 77–129.

Ferrucci, D. A., Brown, E. W. & et. al., J. C. (2010), 'Building watson: An overview of the deepqa project', *AI Magazine* **31**(3), 59–79.

Fraigniaud, P., Gasieniec, L., Kowalski, D. R. & Pelc, A. (2006), 'Collective tree exploration', *Networks: An International Journal* **48**(3), 166–177.

Freund, Y., Iyer, R. D., Schapire, R. E. & Singer, Y. (2003), 'An efficient boosting algorithm for combining preferences', *Journal of Machine Learning Research* **4**, 933–969.

Fu, W., Nair, V. & Menzies, T. (2016), 'Why is differential evolution better than grid search for tuning defect predictors?', *ArXiv* **abs/1609.02613**.

Galárraga, L., Teflioudi, C., Hose, K. & Suchanek, F. M. (2015), 'Fast rule mining in ontological knowledge bases with AMIE+', *VLDB J.* **24**(6), 707–730.

García-Durán, A. & Niepert, M. (2018), Kblrn: End-to-end learning of knowledge base representations with latent, relational, and numerical features, *in* 'UAI'.

Gardner, M. & Mitchell, T. M. (2015), Efficient and expressive knowledge base completion using subgraph feature extraction, *in* 'EMNLP', The Association for Computational Linguistics, pp. 1488–1498.

Gardner, M., Talukdar, P. P., Kisiel, B. & Mitchell, T. M. (2013), Improving learning and inference in a large knowledge-base using latent syntactic cues, *in* 'EMNLP'.

Gardner, M., Talukdar, P. P., Krishnamurthy, J. & Mitchell, T. M. (2014), Incorporating vector space similarity in random walk inference over knowledge bases, *in* '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. 397–406.

Gaulton, A., Hersey, A., Nowotka, M., Bento, A. P., Chambers, J., Mendez, D., Mutowo-Meullenet, P., Atkinson, F., Bellis, L. J., Cibrián-Uhalte, E., Davies, M., Dedman, N., Karlsson, A., Magariños, M. P., Overington, J. P., Papadatos, G., Smit, I. & Leach, A. R. (2017), The chembl database in 2017, *in* 'Nucleic Acids Research'.

Gibrat, J.-F., Madej, T. & Bryant, S. H. (1996), 'Surprising similarities in structure comparison.', *Current opinion in structural biology* **6 3**, 377–85.

Glorot, X. & Bengio, Y. (2010), Understanding the difficulty of training deep feedforward neural networks, *in* 'AISTATS', Vol. 9 of *JMLR Proceedings*, JMLR.org, pp. 249–256.

Greene, C. S., Krishnan, A., Wong, A. K., Ricciotti, E., Zelaya, R. A., Himmelstein, D. S., Zhang, R., Hartmann, B. M., Zaslavsky, E., Sealfon, S. C. et al. (2015), 'Understanding multicellular function and disease with human tissue-specific networks', *Nature genetics* **47**(6), 569.

Gremse, M., Chang, A., Schomburg, I., Grote, A., Scheer, M., Ebeling, C. & Schomburg, D. (2010), 'The brenda tissue ontology (bto): the first all-integrating ontology of all organisms for enzyme sources', *Nucleic acids research* **39**(suppl_1), D507–D513.

Grover, A. & Leskovec, J. (2016), 'node2vec: Scalable feature learning for networks', *KDD: proceedings. International Conference on Knowledge Discovery & Data Mining* **2016**, 855–864.

Guan, Y., Gorenshteyn, D., Burmeister, M., Wong, A. K., Schimenti, J. C., Handel, M. A., Bult, C. J., Hibbs, M. A. & Troyanskaya, O. G. (2012), Tissue-specific functional networks for prioritizing phenotype and disease genes, *in* 'PLoS Computational Biology'.

Günther, S., Kuhn, M., Dunkel, M., Campillos, M., Senger, C., Petsalaki, E., Ahmed, J., Urdiales, E. G., Gewiess, A., Jensen, L. J. et al. (2007), 'Supertarget and matador: resources for exploring drug-target relationships', *Nucleic acids research* **36**(suppl_1), D919–D922.

Guo, S., Wang, Q., Wang, L., Wang, B. & Guo, L. (2016), Jointly embedding knowledge graphs and logical rules, *in* 'EMNLP'.

Gusmão, A. C., Correia, A. H. C., Bona, G. D. & Cozman, F. G. (2018), Interpreting embedding models of knowledge bases: A pedagogical approach, *in* 'Proceddings of WHI'.

Hao, M., Bryant, S. H. & Wang, Y. (2017), 'Predicting drug-target interactions by dual-network integrated logistic matrix factorization', *Scientific reports* **7**, 40376.

Hayashi, K. & Shimbo, M. (2017), On the equivalence of holographic and complex embeddings for link prediction, *in* 'ACL (2)', Association for Computational Linguistics, pp. 554–559.

Hecker, N. & et. al., J. A. (2012), Supertarget goes quantitative: update on drug–target interactions, *in* 'Nucleic Acids Research'.

Himmelstein, D. S., Lizée, A., Hessler, C. S., Brueggeman, L., Chen, S. L., Hadley, D., Green, A. J., Khankhanian, P. & Baranzini, S. E. (2017), Systematic integration of biomedical knowledge prioritizes drugs for repurposing, *in* 'eLife'.

Hitchcock, F. L. (1927), 'The expression of a tensor or a polyadic as a sum of products', *Journal of Mathematics and Physics* **6**(1-4), 164–189.

Holzinger, A., Langs, G., Denk, H., Zatloukal, K. & Müller, H. (2019), 'Causability and explainability of artificial intelligence in medicine', *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **9**.

Hong, M. & Jung, J. J. (2018), 'Multi-sided recommendation based on social tensor factorization', *Inf. Sci.* **447**, 140–156.

Janjic, V. & Przulj, N. (2012), 'Biological function through network topology: a survey of the human diseasome.', *Briefings in functional genomics* **11 6**, 522–32.

Ji, T., Huang, T., Zhao, X., Ma, T. & Liu, G. (2016), 'Tensor completion using total variation and low-rank matrix factorization', *Inf. Sci.* **326**, 243–257.

Jiang, T., Huang, T., Zhao, X., Ji, T. & Deng, L. (2018), 'Matrix factorization for low-rank tensor completion using framelet prior', *Inf. Sci.* **436-437**, 403–417.

Kadlec, R., Bajgar, O. & Kleindienst, J. (2017), Knowledge base completion: Baselines strike back, *in* 'Rep4NLP@ACL', Association for Computational Linguistics, pp. 69–74.

Kanehisa, M., Furumichi, M., Tanabe, M., Sato, Y. & Morishima, K. (2017), 'Kegg: new perspectives on genomes, pathways, diseases and drugs', *Nucleic Acids Research* **45**(D1), D353–D361.

Kanehisa, M. & Goto, S. e. a. (2006), 'From genomics to chemical genomics: new developments in kegg', *Nucleic acids research* **34**(suppl_1), D354–D357.

Kanehisa, M., Sato, Y., Kawashima, M., Furumichi, M. & Tanabe, M. (2016), 'Kegg as a reference resource for gene and protein annotation', *Nucleic Acids Research* **44**(D1), D457–D462.

Kantor, E. D., Rehm, C. D., Haas, J. S., Chan, A. T. & Giovannucci, E. L. (2015), 'Trends in prescription drug use among adults in the united states from 1999-2012.', *JAMA* **314 17**, 1818–31.

Krompass, D., Baier, S. & Tresp, V. (2015), Type-constrained representation learning in knowledge graphs, *in* 'International Semantic Web Conference'.

Kuhn, M., Letunic, I., Jensen, L. J. & Bork, P. (2016), 'The sider database of drugs and side effects', *Nucleic acids research* **44 D1**, D1075–9.

Lacroix, T., Usunier, N. & Obozinski, G. (2018), Canonical tensor decomposition for knowledge base completion, *in* 'ICML', Vol. 80 of *JMLR Workshop and Conference Proceedings*, JMLR.org, pp. 2869–2878.

Landrum, M. J., Lee, J. M., Riley, G. R., Jang, W., Rubinstein, W. S., Church, D. M. & Maglott, D. R. (2014), Clinvar: public archive of relationships among sequence variation and human phenotype, *in* 'Nucleic Acids Research'.

Lao, N. & Cohen, W. W. (2010*a*), 'Relational retrieval using a combination of path-constrained random walks', *Mach. Learn.* **81**(1), 53–67.

Lao, N. & Cohen, W. W. (2010*b*), 'Relational retrieval using a combination of path-constrained random walks', *Machine Learning* **81**(1), 53–67.

Lao, N., Minkov, E. & Cohen, W. W. (2015), Learning relational features with backward random walks, *in* 'ACL (1)', The Association for Computer Linguistics, pp. 666–675.

Lao, N., Mitchell, T. M. & Cohen, W. W. (2011), Random walk inference and learning in a large scale knowledge base, *in* 'EMNLP'.

Lehmann, J. & et. al., R. I. (2014), 'DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia', *Semantic Web Journal* **6**(2), 167–195.

Lei, X. & Zhang, Y. (2019), 'Predicting disease-genes based on network information loss and protein complexes in heterogeneous network', *Inf. Sci.* **479**, 386–400.

Lerer, A., Wu, L., Shen, J., Lacroix, T., Wehrstedt, L., Bose, A. & Peysakhovich, A. (2019), Pytorch-biggraph: A large-scale graph embedding system, *in* 'The 2nd SysML Conference'.

Letovsky, S. & Kasif, S. (2003), 'Predicting protein function from protein/protein interaction data: a probabilistic approach', *Bioinformatics* **19 Suppl 1**, i197–204.

Li, L., Geraghty, O. C., Mehta, Z., Rothwell, P. M. & Study, O. V. (2017), 'Age-specific risks, severity,

time course, and outcome of bleeding on long-term antiplatelet treatment after vascular events: a population-based cohort study', *The Lancet* **390**(10093), 490–499.

Lim, H., Gray, P., Xie, L. & Poleksic, A. (2016), 'Improved genome-scale multi-target virtual screening via a novel collaborative filtering approach to cold-start problem', *Scientific reports* **6**, 38860.

Lipschitz, W. L., Hadidian, Z. & Kerpcsar, A. (1943), Bioassay of diuretics.

Liu, H., Sun, J., Guan, J., Zheng, J. & Zhou, S. (2015), 'Improving compound–protein interaction prediction by building up highly credible negative samples', *Bioinformatics* **31**(12), i221–i229.

Liu, T. (2011), *Learning to Rank for Information Retrieval*, Springer.

Liu, T.-Y. et al. (2009), 'Learning to rank for information retrieval', *Foundations and Trends® in Information Retrieval* **3**(3), 225–331.

Lu, M., Zhao, X.-J., Zhang, L. & Li, F. (2016), 'Semi-supervised concept factorization for document clustering', *Inf. Sci.* **331**, 86–98.

Lu, Y., Lai, Z., Xu, Y., You, J., Li, X. & Yuan, C. (2016), 'Projective robust nonnegative factorization', *Inf. Sci.* **364-365**, 16–32.

Ma, X. & Gao, L. (2012), 'Predicting protein complexes in protein interaction networks using a core-attachment algorithm based on graph communicability', *Inf. Sci.* **189**, 233–254.

Magger, O., Waldman, Y. Y., Ruppin, E. & Sharan, R. (2012), Enhancing the prioritization of disease-causing genes through tissue specific protein interaction networks, *in* 'PLoS Computational Biology'.

Mahdisoltani, F., Biega, J. & Suchanek, F. M. (2015), YAGO3: A knowledge base from multilingual wikipedias, *in* 'CIDR', www.cidrdb.org.

Maji, P., Shah, E. & Paul, S. (2017), 'Relsim: An integrated method to identify disease genes using gene expression profiles and ppin based similarity measure', *Inf. Sci.* **384**, 110–125.

Malone, B., García-Durán, A. & Niepert, M. (2018), Knowledge graph completion to predict polypharmacy side effects, *in* 'DILS'.

Marcotte, E. M., Pellegrini, M., Thompson, M. J., Yeates, T. O. & Eisenberg, D. S. (1999), 'A combined algorithm for genome-wide prediction of protein function', *Nature* **402**, 83–86.

Mattingly, C. J., Colby, G. T., Forrest, J. N. & Boyer, J. (2003), 'The comparative toxicogenomics database (ctd).', *Environmental Health Perspectives* **111**, 793–795.

Mei, J.-P., Kwoh, C.-K., Yang, P., Li, X.-L. & Zheng, J. (2012), 'Drug–target interaction prediction by learning from local information and neighbors', *Bioinformatics* **29**(2), 238–245.

Miller, G. A. (1995), 'Wordnet: A lexical database for english', *Communications of the ACM* **38**(11), 39–41.

Miller, G., Beckwith, R., Fellbaum, C., Gross, D. & Miller, K. (1990), Introduction to WordNet: an on-line lexical database.

Minervini, P., Costabello, L., Muñoz, E., Novácek, V. & Vandenbussche, P. (2017*a*), Regularizing knowledge graph embeddings via equivalence and inversion axioms, *in* 'ECML/PKDD (1)', Vol. 10534 of *Lecture Notes in Computer Science*, Springer, pp. 668–683.

Minervini, P., Costabello, L., Muñoz, E., Novácek, V. & Vandenbussche, P.-Y. (2017*b*), Regularizing knowledge graph embeddings via equivalence and inversion axioms, *in* 'ECML/PKDD'.

Minoda, Y. & Kharasch, E. D. (2001), 'Halothane-dependent lipid peroxidation in human liver microsomes is catalyzed by cytochrome p4502a6 (cyp2a6).', *Anesthesiology* **95 2**, 509–14.

Minsky, M. (1974), A framework for representing knowledge, *in* 'MIT-AI Laboratory Memo 306'.

Mitchell, A. L. & Attwood, T. K. e. a. (2019), 'Interpro in 2019: improving coverage, classification and access to protein sequence annotations', *Nucleic Acids Research* **47**(D1), D351–D360.

Mitchell, T. M., Cohen, W. W. & et. al., E. R. H. J. (2015), Never-ending learning, *in* 'AAAI', Vol. 61, AAAI Press, pp. 2302–2310.

Mnih, A. & Kavukcuoglu, K. (2013), Learning word embeddings efficiently with noise-contrastive estimation, *in* 'NIPS', pp. 2265–2273.

Mohamed, S. K. (2020), 'Predicting tissue-specific protein functions using multi-part tensor decomposition', *Inf. Sci.* **508**, 343–357.

Mohamed, S. K., Muñoz, E., Novácek, V. & Vandenbussche, P. (2017), Identifying equivalent relation paths in knowledge graphs, *in* 'LDK', Vol. 10318 of *Lecture Notes in Computer Science*, Springer, pp. 299–314.

Mohamed, S. K., Nounu, A. & Novácek, V. (2019), Drug target discovery using knowledge graph embeddings, *in* 'SAC', SAC '19, ACM, pp. 11–18.

Mohamed, S. K. & Novácek, V. (2019), Link prediction using multi part embeddings, *in* 'ESWC', Vol. 11503 of *Lecture Notes in Computer Science*, Springer, pp. 240–254.

Mohamed, S. K., Novácek, V. & Vandenbussche, P. (2018), Knowledge base completion using distinct subgraph paths, *in* 'SAC', SAC '18, ACM, pp. 1992–1999.

Mohamed, S. K., Novácek, V., Vandenbussche, P. & Muñoz, E. (2019), Loss functions in knowledge graph embedding models, *in* 'DL4KG@ESWC', Vol. 2377 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 1–10.

Mohamed, S. K., Nováček, V. & Nounu, A. (2019), 'Discovering protein drug targets using knowledge graph embeddings', *Bioinformatics* .

Mrozek, D., Kasprowski, P., Malysiak-Mrozek, B. & Kozielski, S. (2017), 'Life sciences data analysis', *Inf. Sci.* **384**, 86–89.

Muñoz, E., Minervini, P. & Nickles, M. (2019), Embedding cardinality constraints in neural link predictors, *in* 'SAC', ACM, pp. 2243–2250.

Muñoz, E., Novácek, V. & Vandenbussche, P. (2016), Using drug similarities for discovery of possible adverse reactions, *in* 'AMIA 2016', AMIA.

Muñoz, E., Novácek, V. & Vandenbussche, P.-Y. (2017), 'Facilitating prediction of adverse drug reactions by using knowledge graphs and multi-label learning models', *Briefings in*

bioinformatics .

Nascimento, A. C., Prudêncio, R. B. & Costa, I. G. (2016), 'A multiple kernel learning algorithm for drug-target interaction prediction', *BMC bioinformatics* **17**(1), 46.

Neelakantan, A., Roth, B. & McCallum, A. (2015), Compositional vector space models for knowledge base completion, *in* 'Proceedings of the 53rd ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers', pp. 156–166.

Nelson, D. L., Lehninger, A. L. & Cox, M. M. (2008), *Lehninger principles of biochemistry*, Macmillan.

Ngomo, A.-C. N. & Auer, S. (2011), Limes - a time-efficient approach for large-scale link discovery on the web of data, *in* 'IJCAI'.

Nguyen, D. Q., Nguyen, T. D., Nguyen, D. Q. & Phung, D. Q. (2018), A novel embedding model for knowledge base completion based on convolutional neural network, *in* 'NAACL-HLT (2)', Association for Computational Linguistics, pp. 327–333.

Nickel, M., Murphy, K., Tresp, V. & Gabrilovich, E. (2016*a*), 'A review of relational machine learning for knowledge graphs', *Proceedings of the IEEE* **104**(1), 11–33.

Nickel, M., Murphy, K., Tresp, V. & Gabrilovich, E. (2016*b*), 'A review of relational machine learning for knowledge graphs', *Proceedings of the IEEE* **104**(1), 11–33.

Nickel, M., Rosasco, L. & Poggio, T. A. (2016), Holographic embeddings of knowledge graphs, *in* 'AAAI', AAAI Press, pp. 1955–1961.

Nickel, M., Tresp, V. & Kriegel, H. (2011), A three-way model for collective learning on multi-relational data, *in* 'ICML', Omnipress, pp. 809–816.

Nickel, M., Tresp, V. & Kriegel, H. (2012), Factorizing YAGO: scalable machine learning for linked data, *in* 'WWW', ACM, pp. 271–280.

Nie, B. & Sun, S. (2019), 'Knowledge graph embedding via reasoning over entities, relations, and text', *Future Generation Comp. Syst.* **91**, 426–433.

Olayan, R. S., Ashoor, H. & Bajic, V. B. (2017), 'Ddr: efficient computational method to predict drug–target interactions using graph mining and machine learning approaches', *Bioinformatics* **34**(7), 1164–1173.

Orchard, S., Ammari, M., Aranda, B., Breuza, L., Briganti, L., Broackes-Carter, F., Campbell, N. H., Chavali, G., Chen, C., Del-Toro, N. et al. (2013), 'The mintact project—intact as a common curation platform for 11 molecular interaction databases', *Nucleic acids research* **42**(D1), D358–D363.

Overington, J. P., Al-Lazikani, B. & Hopkins, A. L. (2006), 'How many drug targets are there?', *Nature Reviews Drug Discovery* **5**, 993–996.

Papalexakis, E. E., Faloutsos, C. & Sidiropoulos, N. D. (2016), 'Tensors for data mining and data fusion: Models, applications, and scalable algorithms', *ACM TIST* **8**, 16:1–16:44.

Pazos, F. & Sternberg, M. J. E. (2004), 'Automated prediction of protein function and detection of functional sites from structure.', *Proceedings of the National Academy of Sciences of the*

United States of America **101 41**, 14754–9.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. (2011), 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830.

Perozzi, B., Al-Rfou', R. & Skiena, S. (2014), Deepwalk: online learning of social representations, *in* 'KDD', Vol. abs/1403.6652.

Perros, I., Papalexakis, E. E., Wang, F., Vuduc, R. W., Searles, E., Thompson, M. & Sun, J. (2017), Spartan: Scalable parafac2 for large & sparse data, *in* 'KDD'.

Placzek, S., Schomburg, I., Chang, A., Jeske, L., Ulbrich, M., Tillack, J. & Schomburg, D. (2017), Brenda in 2017: new perspectives and new tools in brenda, *in* 'Nucleic Acids Research'.

Pohl, J. E., Thurston, H. F. & Swales, J. D. (1972), The antidiuretic action of diazoxide.

Pujara, J., Augustine, E. & Getoor, L. (2017), Sparsity and noise: Where knowledge graph embeddings fall short, *in* 'EMNLP'.

Qian, R. (2013), 'Understand your world with bing'. Bing Blogs.
URL: *http://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing/*

Radivojac, P., Clark, W. T., Oron, T. R., Schnoes, A. M., Wittkop, T., Sokolov, A., Graim, K., Funk, C., Verspoor, K., Ben-Hur, A. et al. (2013), 'A large-scale evaluation of computational protein function prediction', *Nature methods* **10**(3), 221.

Raman, K. (2010), Construction and analysis of protein–protein interaction networks, *in* 'Automated experimentation'.

Razniewski, S., Suchanek, F. M. & Nutt, W. (2016), But what do we actually know?, *in* 'Proceedings of the 5th Workshop on Automated Knowledge Base Construction, AKBC@NAACL-HLT 2016, San Diego, CA, USA, June 17, 2016', pp. 40–44.

Reddi, S., Kale, S. & Kumar, S. (2018), On the convergence of adam and beyond, *in* 'ICLR', OpenReview.net.

Ribeiro, M. T., Singh, S. & Guestrin, C. (2016), "why should i trust you?": Explaining the predictions of any classifier, *in* 'HLT-NAACL Demos', Vol. abs/1602.04938.

Rolland, T. & et. al, M. T. (2014), 'A proteome-scale map of the human interactome network', *Cell* **159**(5), 1212–1226.

Rosdah, A. A., K. Holien, J., Delbridge, L. M., Dusting, G. J. & Lim, S. Y. (2016), 'Mitochondrial fission–a drug target for cytoprotection or cytodestruction?', *Pharmacology research & perspectives* **4**(3), e00235.

Rothwell, P. M., Wilson, M., Elwin, C.-E., Norrving, B., Algra, A., Warlow, C. P. & Meade, T. W. (2010), 'Long-term effect of aspirin on colorectal cancer incidence and mortality: 20-year follow-up of five randomised trials', *The Lancet* **376**(9754), 1741–1750.

Rungruangsak-Torrissen, K., Carter, C. G., Sundby, A., Berg, A. E. & Houlihan, D. F. (1999), 'Maintenance ration, protein synthesis capacity, plasma insulin and growth of atlantic

salmon (salmo salar l.) with genetically different trypsin isozymes', *Fish Physiology and Biochemistry* **21**, 223–233.

Scarselli, F., Tsoi, A. C. & Hagenbuchner, M. (2004), Computing personalized pageranks, *in* 'Proceedings of the 13th international conference on World Wide Web - Alternate Track Papers & Posters, WWW 2004, New York, NY, USA, May 17-20, 2004', pp. 382–383.

Schlichtkrull, M. S., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I. & Welling, M. (2018), Modeling relational data with graph convolutional networks, *in* 'ESWC', Vol. 10843 of *Lecture Notes in Computer Science*, Springer, pp. 593–607.

Schomburg, I., Chang, A., Ebeling, C., Gremse, M., Heldt, C., Huhn, G. & Schomburg, D. (2004), 'Brenda, the enzyme database: updates and major new developments', *Nucleic acids research* **32**(suppl_1), D431–D433.

Singhal, A. (2012), 'Introducing the knowledge graph: things, not strings'. Google Official Blog. **URL:** *"https://googleblog.blogspot.ie/2012/05/introducing-knowledge-graph-things-not.html"*

Sleno, L. & Emili, A. (2008), 'Proteomic methods for drug target discovery', *Current opinion in chemical biology* **12**(1), 46–54.

Sneader, W. (2005), *Drug discovery: a history*, John Wiley & Sons.

Snoek, J., Larochelle, H. & Adams, R. P. (2012), Practical bayesian optimization of machine learning algorithms, *in* 'NIPS'.

Socher, R., Chen, D., Manning, C. D. & Ng, A. (2013), Reasoning with neural tensor networks for knowledge base completion, *in* 'Advances in neural information processing systems', pp. 926–934.

Solis, F. J. & Wets, R. J.-B. (1981), 'Minimization by random search techniques', *Math. Oper. Res.* **6**, 19–30.

Sowa, J. F. (2006), Semantic networks, *in* 'Encyclopedia of Cognitive Science'.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014), 'Dropout: A simple way to prevent neural networks from overfitting', *Journal of Machine Learning Research* **15**, 1929–1958.

Stark, C., Breitkreutz, B.-J., Chatr-aryamontri, A., Boucher, L., Oughtred, R., Livstone, M. S., Nixon, J., Auken, K. V., Wang, X., Shi, X., Reguly, T., Rust, J. M., Winter, A. G., Dolinski, K. & Tyers, M. (2007), 'The biogrid interaction database: 2011 update', *Nucleic acids research* **39 Database issue**, D698–704.

Su, C., Tong, J., Zhu, Y., Cui, P. & Wang, F. (2018), 'Network embedding in biomedical data science.', *Briefings in bioinformatics* .

Sun, P. G., Quan, Y.-N., Miao, Q. & Chi, J. (2018), 'Identifying influential genes in protein-protein interaction networks', *Inf. Sci.* **454-455**, 229–241.

Szklarczyk, D., Morris, J. H., Cook, H. V., Kuhn, M., Wyder, S., Simonovic, M., Santos, A., Doncheva, N. T., Roth, A., Bork, P., Jensen, L. J. & von Mering, C. (2017), The string database

in 2017: quality-controlled protein–protein association networks, made broadly accessible, *in* 'Nucleic Acids Research'.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J. & Mei, Q. (2015), Line: Large-scale information network embedding, *in* 'WWW'.

Tatonetti, N. P., Ye, P., Daneshjou, R. & Altman, R. B. (2012), 'Data-driven prediction of drug effects and interactions.', *Science translational medicine* **4 125**, 125ra31.

Terstappen, G. C., Schlüpen, C., Raggiaschi, R. & Gaviraghi, G. (2007), 'Target deconvolution strategies in drug discovery', *Nature Reviews Drug Discovery* **6**(11), 891.

The Uniprot Consortium (2015), Uniprot: a hub for protein information, *in* 'Nucleic Acids Research'.

Toutanova, K. & Chen, D. (2015), Observed versus latent features for knowledge base and text inference, *in* 'Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)', ACL, pp. 57–66.

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P. & Gamon, M. (2015), Representing text for joint embedding of text and knowledge bases, *in* 'EMNLP', The Association for Computational Linguistics, pp. 1499–1509.

Trouillon, T. & Nickel, M. (2017), 'Complex and holographic embeddings of knowledge graphs: A comparison', *CoRR* **abs/1707.01475**.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É. & Bouchard, G. (2016), Complex embeddings for simple link prediction, *in* 'ICML', Vol. 48 of *JMLR Workshop and Conference Proceedings*, JMLR.org, pp. 2071–2080.

Uhlén, M., Fagerberg, L., Hallström, B. M., Lindskog, C., Oksvold, P., Mardinoglu, A., Sivertsson, Å. K., Kampf, C., Sjöstedt, E., Asplund, A. S., Olsson, I., Edlund, K., Lundberg, E., Navani, S. S., Szigyarto, C. A.-K., Odeberg, J., Djureinovic, D., Takanen, J. O., Hober, S., Alm, T., Edqvist, P. H. D., Berling, H., Tegel, H., Mulder, J., Rockberg, J., Nilsson, K. P. R., Schwenk, J. M., Hamsten, M. C., von Feilitzen, K., Forsberg, M. F., Persson, L., Johansson, F., Zwahlén, M., von Heijne, G., Nielsen, J. & Ponten, F. (2015), 'Tissue-based map of the human proteome', *Science* **347**.

van der Maaten, L. (2014), 'Accelerating t-sne using tree-based algorithms', *Journal of Machine Learning Research* **15**, 3221–3245.

Verster, J. C. & Volkerts, E. R. (2004), 'Clinical pharmacology, clinical efficacy, and behavioral toxicity of alprazolam: a review of the literature.', *CNS drug reviews* **10 1**, 45–76.

Vrandecic, D. & Krötzsch, M. (2014), 'Wikidata: a free collaborative knowledgebase', *Commun. ACM* **57**(10), 78–85.

W3C (1997), "world wide web consortium publishes public draft of resource description framework", *in* 'W3C Consortium Press Release. Cambridge, MAS. 1997-10-03. http://www.w3.org/Press/RDF'.

Wan, F., Hong, L., Xiao, A., Jiang, T. & Zeng, J. (2019), 'Neodti: neural integration of neighbor

information from a heterogeneous network for discovering new drug-target interactions', *Bioinformatics* **35 1**, 104–111.

Wang, Q., Mao, Z., Wang, B. & Guo, L. (2017), 'Knowledge graph embedding: A survey of approaches and applications', *IEEE Trans. Knowl. Data Eng.* **29**(12), 2724–2743.

Wang, Z., Zhang, J., Feng, J. & Chen, Z. (2014), Knowledge graph embedding by translating on hyperplanes, *in* 'AAAI', AAAI Press, pp. 1112–1119.

Warde-Farley, D., Donaldson, S. L., Comes, O., Zuberi, K., Badrawi, R., Chao, P., Franz, M., Grouios, C., Kazi, F., Lopes, C. T., Maitland, A., Mostafavi, S., Montojo, J., Shao, Q., Wright, G., Bader, G. D. & Morris, Q. (2010), The genemania prediction server: biological network integration for gene prioritization and predicting gene function, *in* 'Nucleic Acids Research'.

Weber, L., Minervini, P., Münchmeyer, J., Leser, U. & Rocktäschel, T. (2019), Nlprolog: Reasoning with weak unification for question answering in natural language, *in* 'ACL (1)', Association for Computational Linguistics, pp. 6151–6161.

Wei, L., Tang, J. & Zou, Q. (2017), 'Local-dpp: An improved dna-binding protein prediction method by exploring local evolutionary information', *Inf. Sci.* **384**, 135–144.

Whirl-Carrillo, M., McDonagh, E. M., Hebert, J., Gong, L., Sangkuhl, K., Thorn, C., Altman, R. B. & Klein, T. E. (2012), 'Pharmacogenomics knowledge for personalized medicine', *Clinical Pharmacology & Therapeutics* **92**(4), 414–417.

Wishart, D. S., Knox, C., Guo, A. C., Cheng, D., Shrivastava, S., Tzur, D., Gautam, B. & Hassanali, M. (2008), 'Drugbank: a knowledgebase for drugs, drug actions and drug targets', *Nucleic Acids Research* **36**, D901–D906.

Wishart, D. S., Knox, C., Guo, A. C., Shrivastava, S., Hassanali, M., Stothard, P., Chang, Z. & Woolsey, J. (2006), 'Drugbank: a comprehensive resource for in silico drug discovery and exploration', *Nucleic Acids Research* **34**, D668–D672.

Xia, F., Liu, T., Wang, J., Zhang, W. & Li, H. (2008), Listwise approach to learning to rank: theory and algorithm, *in* 'ICML', Vol. 307 of *ACM International Conference Proceeding Series*, ACM, pp. 1192–1199.

Xiang, E. W., Liu, N. N., Pan, S. J. & Yang, Q. (2009), Knowledge transfer among heterogeneous information networks, *in* '2009 IEEE International Conference on Data Mining Workshops'.

Xie, L., Xie, L., Kinnings, S. L. & Bourne, P. E. (2012), 'Novel computational approaches to polypharmacology as a means to define responses to individual drugs', *Annual review of pharmacology and toxicology* **52**, 361–379.

Xu, B., Guan, J., Wang, Y. & Wang, Z. (2017), 'Essential protein detection by random walk on weighted protein-protein interaction networks', *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **16**, 377–387.

Yamanishi, Y., Araki, M., Gutteridge, A., Honda, W. & Kanehisa, M. (2008), 'Prediction of drug–target interaction networks from the integration of chemical and genomic spaces', *Bioinformatics* **24**(13), i232–i240.

Yang, B., Yih, W., He, X., Gao, J. & Deng, L. (2015*a*), Embedding entities and relations for learning and inference in knowledge bases, *in* 'ICLR', Vol. abs/1412.6575.

Yang, B., Yih, W., He, X., Gao, J. & Deng, L. (2015*b*), Learning multi-relational semantics using neural-embedding models, *in* 'ICLR'.

Yousef, W. A., Abouelkahire, A. A., Marzouk, O. S., Mohamed, S. K. & Alaggan, M. N. (2019), 'Dvp: Data visualization platform', *ArXiv* **abs/1906.11738**.

Zeng, X., Ding, N., Rodríguez-Patón, A. & Zou, Q. (2017), Probability-based collaborative filtering model for predicting gene–disease associations, *in* 'BMC Medical Genomics'.

Zhu, Y., Elemento, O., Pathak, J. & Wang, F. (2018), 'Drug knowledge bases and their applications in biomedical informatics research.', *Briefings in bioinformatics* .

Zitnik, M., Agrawal, M. & Leskovec, J. (2018), Modeling polypharmacy side effects with graph convolutional networks, *in* 'Bioinformatics'.

Zitnik, M. & Leskovec, J. (2017), Predicting multicellular function through multi-layer tissue networks, *in* 'Bioinformatics'.

Zitnik, M. & Zupan, B. (2016), 'Collective pairwise classification for multi-way analysis of disease and drug data', *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing* **21**, 81–92.

Zou, B., Li, C., Tan, L. & Chen, H. (2015), 'GPUTENSOR: efficient tensor factorization for context-aware recommendations', *Inf. Sci.* **299**, 159–177.