From entity to relation, a comparative study about relation prediction models

Master Thesis

presented by
Dung Duc Huynh
Matriculation Number 1711573

submitted to the
Data and Web Science Group
Prof. Dr. Rainer Gemulla
Daniel Ruffinelli
University of Mannheim

June, 2022

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, most real-world information, such as social and biological information, can be well encoded into graphs (Lü & Zhou, 2011). Each vertex represents an entity, an object, or a biological element (proteins, genes, etc.), while each edge represents a relation or an interaction between entities (Chang et al., 2020). Those graphs can be thought of as Knowledge Graphs (KGs), and each connection between two entities is a *fact* (Nickel et al., 2016a). For example, ”Berlin is the capital of Germany” is a fact where *Berlin* is a subject entity (i), *Germany* is an object entity (j), and their relation/predicate (k) is *”is capital of.”* By applying KG, different applications have been proved that we can achieve better results like question-answering (Bordes et al., 2014a;b), semantic parsing (Berant et al., 2013; Heck et al., 2013), and named entity disambiguation (Damljanovic & Bontcheva, 2012; Zheng et al., 2012).

Unfortunately, many KGs, such as biological networks (Amaral, 2008; Stumpf et al., 2008; Yu et al., 2008), contain numerous undiscovered relations between two entities. This is essential since existing knowledge graphs often miss many facts, and some of the edges they have are incorrect (Angeli & Manning, 2013). There are two approaches to adding missing relation: (1) predicting a missing entity based on a given entity and relation - *entity prediction* (Wang et al., 2017), and (2) directly predicting the relation between two entities - *relation prediction* (Lin et al., 2015a; Xie et al., 2016b). However, in most KG studies, adding missing relations between two entities is typically referred to as entity prediction or *entity ranking* (Lin et al., 2015a). The relation prediction and entity prediction are generally called *link prediction*.

Several models have been proposed to perform the link prediction, but recently, those given more attention are models based on the learning representation of the

KG - *knowledge graph embedding (KGE)* (Wang et al., 2017). They have been successfully applied to knowledge graph completion (Nickel et al., 2016a) as well as in downstream tasks and applications such as recommender systems (Wang et al., 2017).

Problem Statement. Although most KGE models can perform both entity and relation predictions, (Wang et al., 2017) namely RESCAL (Nickel et al., 2011), TransE (Bordes et al., 2013b), ComplEx (Trouillon et al., 2016), TuckER (Balazevic et al., 2019), RotatE (Sun et al., 2019), and many more, previous KGE studies have only been limited to entity prediction for training, evaluating, and testing the performance of models (Yang et al., 2014; Wang et al., 2014; Trouillon et al., 2016; Shang et al., 2018; Sun et al., 2019). The reason could be that the number of candidates for entity prediction is usually extremely larger than the number of candidates for predicting relation. Despite this, the importance of relation prediction in training and evaluation should not be neglected.

Using a biomedical Knowledge Graph, Chang et al. (2020) pointed out the importance of relation prediction in evaluating KGE models and encouraged analyzing the performance of models on relation prediction. By evaluating relation prediction, relation representations learned by models could directly be evaluated based on model prediction (Chang et al., 2020). For example, two relations ”place of birth” and ”lived place” could be similar at a certain level, but models must understand that people can live in more than one place throughout their life. If that person is an artist, the ”origin place of artist” relation should be similar to the ”place of birth” relation as well as the ”lived place” relation.

Additionally, Chen et al. (2021) showed that by combining relation prediction with entity ranking, the models could perform better than using only entity ranking as a training objective. However, Chen et al. (2021) reported on the evaluation of entity ranking, and models were selected based on the Mean Reciprocal Rank (MRR) of entity ranking. The MRR and Hits@K (Hit ratios of the top-K ranked results) for predicting relations were neglected.

Contributions. Based on those observations, it is essential to conduct a study of relation prediction not only in training objectives but also in evaluation protocol. Thus, it is crucial to (1) conduct a comparative study to analyze the impact of relation prediction on training objectives and on evaluation; (2) evaluate the performance of KGE models on relation prediction in a similar experimental setting.

As mentioned before, Chen et al. (2021) have studied the impact of relation prediction on training objectives and reported the results only for entity prediction.

Their approach might have been more interesting if the relation prediction performance had also been studied. Therefore, in this thesis, we attempted to reproduce their results and analyze the impact of combining two training objectives on relation prediction performance.

Outline of thesis. The thesis is structured as follows. Chapter 2 provides a common knowledge about Knowledge Graph, Knowledge Graph Embedding, as well as current well-known training objectives, loss function, and evaluation protocols. Additionally, in this chapter, we establish the terminologies as well as the definition we used throughout this thesis. Finally, chapter 2 discusses the relevant papers regarding relation prediction.

Chapter 3 describes step-by-step how we reproduced the results from Chen et al. (2021)'s study; we also discuss the differences between two codebases in the chapter as well as to conduct the ablation study to identify the main difference between the two codebases.

Chapter 4 presents the results of a comparative study. We will discuss the impact of relation prediction not only on training objectives but also on evaluation protocol. Based on the results of the choice models, this chapter also demonstrates the market simulations and measures the willingness to pay in the competitive setting before drawing some managerial implications.

Based on the results in chapter 4, chapter 5 is where we analyze the results as well as explain the reason why models can achieve those results.

Chapter 6 concludes with a discussion on the main findings and limitations of the study.

Chapter 2

Background and relation prediction studies

2.1 Knowledge Graph embedding: Models, Training, Evaluation

Given a set $\mathcal{E}$ of entities and a set $\mathcal{R}$ of relations, a knowledge graph $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ contains a set of subject-predicate-objects triples (i, k, j), where each triplet is considered as a fact (a *positive example/triple*) of the knowledge graph $\mathcal{G}$. A triple represents the relationship $k \in \mathcal{R}$ between subject $i \in \mathcal{E}$ and object $j \in \mathcal{E}$ of that triple.

Knowledge Graph Embedding Models. Generally, KGE models are trained and evaluated in the context of multi-relational link prediction for the knowledge graph (Ruffinelli et al., 2020). The goal of multi-relational link prediction is to ”complete the KG,” i.e., predicting the true but unobserved facts based on the information in $\mathcal{G}$. To perform the prediction, a KGE model learns how to represent entities $i \in \mathcal{E}$ and relations $k \in \mathcal{R}$ as vectors $e_i \in \mathbb{R}^{d_e}$ and $r_k \in \mathbb{R}^{d_r}$ in a low-dimensional vector space respectively. Those vectors are referred to as representation vectors/embedding vectors. The $d_e, d_r \in \mathbb{N}^+$ are the *embedding size* of the embedding vectors.

Each model uses particular score function $s : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \mapsto \mathbb{R}$ to associate a score $s(i, k, j)$ of a triplet with the likelihood that the triple (i, k, j) is in the knowledge graph $\mathcal{G}$. The higher the score of triple is, the more likely the triple considered to be a fact (i.e., a positive triple) in that knowledge graph $\mathcal{G}$.

The score function of a KGE model takes form $s(i, k, j) = f(e_i, r_k, e_j)$ where e_i, r_k, e_j are embedding vectors of subject, predicate and object respectively. Function f represented the model architecture could be a fixed function or the learned function (Ruffinelli et al., 2020).

Training objectives. There are three commonly used approaches to train a KGE model: Negative Sampling (NegSamp) (Bordes et al., 2013b), 1vsAll, (Lacroix et al., 2018) and KvsAll (Dettmers et al., 2018). The main difference between those three training objectives is in generating the *negative examples*, i.e., the triples that are not considered incorrect compared to their associated true example.

Given a triple $t = (i, k, j)$ in training data, *NegSamp* generates the (pseudo-) negative triples by randomly replacing the subject, predicate, or object of the triple t with other entities/predicates (and verifying that obtained triples not in KG). While *1vsAll* omits to sample and considers all triples generated by perturbing the subject and object positions as negative examples (even if those tuples exist in the KG). Finally, *KvsAll*[1] constructs batches from non-empty rows $(i, k, *)$ or $(*, k, j)$ instead of from individual triples, and labels all such triples as either positive (occurs in training data) or negative (otherwise).

In most studies, the negative triples are generated by replacing the subject and object entities from KG and keeping the true relations in the negative examples. We refer this those triples as *negative entity triples*. Besides perturbing subjects and object positions only, we can also generate negative examples by replacing the true relations with another relation in KG. We refer to those triples as *negative relation triples*.

In order to incorporate relation prediction into the commonly-used 1vsAll training objective as proposed by Chen et al. (2021), we include triples generated perturbing the predicate positions (negative relation triples) into the set of negative triples of t.

In this thesis, we used 1vsAll as the primary training objective, so as to be able to compare with Chen et al. (2021)'s results. Thus, to distinguish between original 1vsAll and 1vsAll including negative relation examples, throughout the thesis, we will use the term ***standard training*** to refer to original 1vsAll and ***hybrid training*** to refer to 1vsAll including negative relation examples. Finally, 1vsAll objective containing only negative examples generated by perturbing the predicate positions is termed as ***relation training***.

[1] KvsAll is the term from Ruffinelli et al. (2020), the term the used by Dettmers et al. (2018) is 1-N.

Loss function. There are several loss functions that have been proposed so far. Mean squared error (MSE) is the loss between the score of each triple and its label (positive - 1 or negative - 0). Pairwise margin ranking with hinge loss (MR) is only applicable with 1vsAll and NegSamp because it makes the score of positive triple higher than its negative examples by at least the margin η. The binary cross-entropy (BCE) loss proposed by Trouillon et al. (2016) applies the sigmoid to the score of each triple (positive and negative) and uses cross-entropy between the resulting probability and that triple's label to calculate the loss. BCE is suitable for multi-class and multi-label classification (Ruffinelli et al., 2020). **Finally, cross-entropy (CE) between the model distribution (softmax distribution over scores) and the data distribution (labels of corresponding triples, normalized to sum to 1). CE is more suitable for multi-class classification (as in NegSamp and 1vsAll) but has also been used in the multi-label setting (KvsAll) (Ruffinelli et al., 2020).**

Evaluation. The performance of the model could be determined by evaluating extitentity prediction and/or *relation prediction* tasks. Given a triple (i, k, j), the difference between entity prediction and relation prediction would be the model's questions: for entity prediction, the model tries to answer $(i, k, ?)$ question - *object prediction* and $(?, k, j)$ question - *subject prediction*. In contrast, the *relation prediction* model tries to answer the $(i, ?, j)$ question.

To do so, all potential answered triples are first ranked by their score, the higher score the triple has, the lower rank the triple gets. All answered triples, except the give triple (i, k, j), that are in training, validation or test data are discarded so that only new predictions are evaluated (Ruffinelli et al., 2020). The reason for this is those triplets may be ranked above the test triplet, but this should not be counted as an error because those triplets are valid triples, i.e., they are in KG (Bordes et al., 2013b). Finally, the lower the true answer is ranked, the better the model is; thus, mean reciprocal rank (MRR), mean rank (MR) or the average Hits@K are computed. The lower the rank is, the higher MRR and Hits@K are. While the lower the rank is, the lower the mean rank is. From now on, all of metrics will be reported as filtered metrics.

MRR or Hits@K could be computed separately for each of the question types, i.e., subject, object, relation predictions, and we can aggregate them by computing *the micro average* of metrics (Equation 2.1). To simplify, let denote the MRR of subject and object predictions be ***entity MRR*** and MRR of relation prediction be ***relation MRR***. It's the same for Hits@K, i.e., ***entity Hits@K*** and ***relation Hits@K*** respectively. The MRR for the subject, object, and relation predictions can be called as ***overall MRR***, same for Hits@K, i.e., ***overall Hits@K*** (Equation 2.2).

We formally define the evaluation metrics overall filtered MRR and overall filtered Hits@K. Given a triple (i, k, j), we denote the rank$(i|k, j)$ as the filtered rank of object j, that is the rank of model score $s(i, k, j)$ among the collection of all pseudo-negative object scores

$$\{s(i, k, j) : i' \in \mathcal{E} \text{ and } (i', k, j) \text{ does not appear in training, validation or test data}\}.$$

If a tie exists, the final rank is the mean rank of all triplets with the same score as $s(i, k, j)$. Define the rank$(i|k, j)$ likewise and denote the $\mathcal{K}^{\text{exam}}$ the set of exam triples. Then

$$\begin{aligned}\text{MRR}_{\text{overall}} = \frac{1}{3|\mathcal{K}^{\text{exam}}|} \sum_{(i,k,j)\in\mathcal{K}^{\text{exam}}} \Bigg(& \frac{1}{\text{rank}(i|k,j)} + \\ & \frac{1}{\text{rank}(j|i,k)} + \\ & \frac{1}{\text{rank}(k|i,j)} \Bigg)\end{aligned} \tag{2.1}$$

$$\begin{aligned}\text{Hits@K}_{\text{overall}} = \frac{1}{3|\mathcal{K}^{\text{exam}}|} \sum_{(i,k,j)\in\mathcal{K}^{\text{exam}}} \Bigg(& \mathbb{1}(\text{rank}(i|k,j) \leq K) + \\ & \mathbb{1}(\text{rank}(j|i,k) \leq K) + \\ & \mathbb{1}(\text{rank}(k|i,j) \leq K) \Bigg)\end{aligned} \tag{2.2}$$

where $\mathbb{1}(E)$ is a indicator such that $\mathbb{1}(E)$ is 1 if E is true while $\mathbb{1}(E)$ is 0 if E is false.

2.2 Literature on relation prediction.

The insufficiency of relation prediction studies. There is a vast amount of literature on adopting entity prediction as to the main evaluation task (i.e., models were evaluated on answering $(i, k, ?)$ and $(?, k, j)$ questions) and training objectives (i.e., only negative entity examples are included in the negative set) for KGE methods (Bordes et al., 2013b; Lin et al., 2015b; Nickel et al., 2016b; Wang et al., 2014). Unfortunately, in terms of relation prediction, few studies have been published on studying relation prediction both in training objective (including negative relation examples to negative set) and evaluation protocol (i.e., models were evaluated on answering $(i, ?, j)$ question).

Table 2.1 shows ten papers that I considered related to relation prediction. All those papers were published in top conferences, namely AAAI, NAACL, AKBC, etc., and the rank of the conference was consulted from *The Computing Research and Education Association of Australasia (CORE Inc)*[2].

	Type	Paper	Year	Conference	Relation Prediction	Motivation
1	**Objective**	**Relation Prediction as an Auxiliary Training Objective for Improving Multi-Relational Graph Representations**	**2021**	**AKBC**	**As Auxiliary Training Object**	**To Improve performance on standard Entity Ranking**
2	**Evaluation**	**Benchmark and Best Practices for Biomedical Knowledge Graph Embeddings**	**2020**	**ACL**	**Should be included with standard entity ranking**	**A new standard way to evaluate KGE models**
3	Application	Strong Baselines for Simple Question Answering over Knowledge Graphs with and without Neural Networks	2018	NAACL	A subtask in Question and Answering problem	To Identify the relation being queried in QA
4	**Prediction**	**Type-augmented Relation Prediction in Knowledge Graphs**	**2021**	**AAAI**	**Type information and instance-level information are encoded**	**To improve performance on relation prediction**
5	Evaluation	Representation Learning of Knowledge Graphs with Entity Descriptions	2016	AAAI	A subtask in evaluation protocol	As a part of the graph completion task
6	Evaluation	Representation Learning of Knowledge Graphs with Hierarchical Types	2016	IJCAI		
7	Evaluation	Modeling Relation Paths for Representation Learning of Knowledge Bases	2015	EMNLP		
8	Evaluation	ProjE: Embedding Projection for Knowledge Graph Completion	2017	AAAI		
9	Evaluation	Shared Embedding Based Neural Networks for Knowledge Graph Completion	2018	CIKM		
10	Evaluation	KG-BERT: BERT for Knowledge Graph Completion	2019	AAAI*		

Table 2.1: List of ten papers from top conferences related to relation prediction. *Type*: indicates the where relation prediction was mentioned in the paper with three values: *objective* means in training objective, *evaluation* means in evaluation task, and *prediction* means in model prediction. The *relation prediction* tells what authors have done with relation prediction. *Motivation* shows us the reason why they considered the relation prediction. The *bold* indicates the paper mainly focuses on relation prediction

As shown in Table 2.1, six out of ten papers (from paper number five to paper number ten) considered relation prediction as an additional task along with entity prediction in the evaluation protocol. For example, Yao et al. (2019) evaluated their proposed model - KGE-BERT, by performing relation prediction on FB15K dataset along with entity prediction and triple classification. Similarly, Shi

[2] CORE Inc: http://portal.core.edu.au/conf-ranks/

& Weninger (2017) also performed relation prediction and entity prediction tasks when evaluating the performance of their proposed model - ProjE (Bordes et al., 2013a). In the same way as other scholars, Xie et al. (2016a;b) also decided to utilize relation and entity prediction as to their two main evaluation tasks for TKRL and DKRL models. In summary, relation prediction is not the primary task to propose new models in most papers; it has often been considered an additional task to evaluate with the entity prediction (Chang et al., 2020).

Attempts to study relation prediction. Although the number of papers studying relation prediction is limited, papers one, two, and four have attempted to study the impact of relation prediction on the training objective, evaluation protocol, and even to improve model performance in relation prediction.

Cui et al. (2021), authors of the TaRP model, successfully utilized the description of entities and relations to achieve better performance in terms of relation prediction. To select their best models, Cui et al. (2021) used relation Hits@1 as the main metric. In the same year, Chen et al. (2021), in an attempt to improve the model performance in entity ranking, suggested incorporating negative relation examples to the negative set instead of only negative entity examples. Finally, Chang et al. (2020) stressed the importance of including the relation prediction task in the evaluation protocol alongside entity ranking.

Although there is a lack of attention to relation prediction, we still observed some attempts to study the impact, and some scholars tried to propose a new model for relation prediction.

Dataset for relation prediction. Table 2.2 lists the datasets that have been used to evaluate and analyze the performance of the model in relation prediction. As can be seen, FB15K is the most popular dataset, followed by FB15K-237, UMLS, WN18, and WN18RR. The remaining datasets are the extensions of those popular datasets. Further noticed that most popular datasets (i.e., FB15K, FB15K-237, UMLS, WN18, and WN18RR) contain a small number of relations except FB15K and FB15K-237, while the remaining datasets which contain a large number of relation were not really considered in most of studies.

Training objectives and evaluation metrics. Noticeably, in those ten studies, the most common training objective is NegSamp which contains negative triples generated by randomly replacing the subject, object, or predicate (Xie et al., 2016a;b; Lin et al., 2015a; Shi & Weninger, 2017). Those negative triples are verified not to appear in the KG. Alongside with NegSamp training objective, margin ranking

ID	Dataset	Entities	Relations	Appeared in (papers)
1	FB15k (Bordes et al., 2013a)	14,951	**1,345**	9
2	FB15k-237 (Toutanova et al., 2015)	27,395	237	3
3	UMLS (Kok & Domingos, 2007)	135	49	3
4	WN18 (Dettmers et al., 2018)	40 559	18	2
5	WN18RR (Dettmers et al., 2018)	40,943	11	2
6	SemMedDB	1,6M	**117K**	1
7	SIMPLEQUESTIONS (facts from FB2M)	45,335	**1,703**	1
8	Aristo-v4 (Dalvi et al., 2017)	44,950	**1,605**	1
9	FB40K	39,528	**1,336**	1
10	FB20K	19,923	**1,336**	1
11	DB111K-174 (Hao et al., 2019)	98,336	298	1
12	Nations (Kok & Domingos, 2007)	125	57	1
13	YAGO26K-906 (Hao et al., 2019)	26,078	34	1
14	Kinships (Kok & Domingos, 2007)	104	26	1

Table 2.2: The dataset in relation prediction papers with their corresponding number of entities and relations as well as the number of papers used them.

loss was mainly adopted to calculate the loss between each positive triple and its negative examples (Xie et al., 2016a;b; Lin et al., 2015a).

Despite Mean Rank's shortcomings since it is sensitive to outliers (Nickel et al., 2016b), MR is the most popular metric used to evaluate the models' performance in relation prediction (Cui et al., 2021; Xie et al., 2016a;b; Lin et al., 2015a; Shi & Weninger, 2017; Guan et al., 2018; Yao et al., 2019). Besides, MRR and Hits@{1, 3, 10} were also used to evaluate the models' performance (Chang et al., 2020; Chen et al., 2021). Those metrics were reported separately between relation prediction and entity prediction.

Model selection. Many studies have decided to select their best models based on filter entity MRR, entity Hits@K, (Xie et al., 2016a;b; Lin et al., 2015a; Shi & Weninger, 2017; Guan et al., 2018; Yao et al., 2019) or relation Hits@K only (Cui et al., 2021). We believe that no other authors have studied the effect of selecting the best model on overall MRR so far.

2.3 Conclusion

Despite the interest in relation prediction, research has focused on entity ranking rather than relation prediction. Table 2.1 demonstrates the lack of studies that truly focus on relation prediction performance. Most papers only considered relation prediction (i.e., models were evaluated on answering $(i, ?, j)$ question) as an auxiliary task in model evaluation. Unexpectedly, no paper studies selecting the best models that can achieve the best performance on both relation prediction and entity prediction simultaneously (overall MRR).

To the best of our knowledge, Chen et al. (2021) are the first authors who studied the effect of including the negative relation examples alongside negative entity examples in the 1vsAll training objective. Their empirical experiments showed an improvement in entity prediction when including negative relation examples in the 1vsAll training objective. Their paper research might have been more interesting if the relation prediction performance of models had been reported when studying hybrid training.

In the next chapter (Chapter 3), to allow us to study the impact of hybrid training on relation prediction performance, we attempted to reproduce their results using LibKGE (Broscheit et al., 2020) to assure that we have the same models that they had. The reason for this effort is to have fairness in the comparative study: where we comprehensively studied the impact of hybrid training and overall MRR on models' performance in relation prediction tasks (Chapter 4).

Chapter 3

Reproducibility of LibKGE framework

In their analysis of hybrid training, Chen et al. (2021) showed that by including negative relation examples to the 1vsAll training objective, models could perform better in entity prediction tasks. Therefore, this chapter will describe our attempt to reproduce Chen et al. (2021)'s results using the LibKGE framework.

Table 3.1 shows the performance of their best models on entity prediction on FB15K-237 and WN18RR. In the table, the *"Entity"* indicates that the negative examples are negative entity examples generated by perturbing the subject and object position. At the same time, the *"Relation"* indicated that negative examples are negative relation examples. The *"Yes"* value determines the existence of those corresponding types of negative examples in 1vsAll, while the *"No"* value means there is an absence of those types in 1vsAll.

Table 3.1 clearly highlights the perquisite of incorporating negative relation examples with negative entity examples into the 1vsAll training objective (hybrid training objective). By including negative relation examples, the ComplEx model can achieve a 2% higher entity MRR on the FB15K-237 dataset. We also observed the improvement of the model's performance in entity Hits@{1, 3, 10}. However, on WN18RR, the improvements were not significant, ComplEx model can achieve a 0.1% higher in all of the metrics.

Although their finding is remarkable, the models' performance on relation prediction was neglected, and the impact of hybrid training on overall MRR (i.e., micro-average of entity MRR and relation MRR) was not examined. Furthermore, how models perform on both entity ranking and relation prediction if we use overall MRR in model selection alongside with hybrid training objective is also an

Dataset	Entity	Relation	MRR	Hits@1	Hits@3	Hits@10
FB15K-237	Yes	No	36.6	27.1	40.1	55.7
	Yes	Yes	**38.8**	**29.8**	**42.5**	**56.8**
	No	Yes	26.3	18.7	28.7	41.1
WN18RR	Yes	No	48.7	44.1	50.1	58.0
	Yes	Yes	**48.8**	**44.3**	**50.5**	**57.8**
	No	Yes	25.8	21.2	29.0	33.9

Table 3.1: The performance of ComplEx models in Entity Ranking on Test data. The results are taken from (Chen et al., 2021)'s paper.

interesting question we attempt to study.

Therefore, to have fairness in the comparative study (Chapter 4), it is essential to reproduce the models found by Chen et al. (2021) using LibKGE (Broscheit et al., 2020). Using LibKGE python-library (Broscheit et al., 2020) for implementation, we can utilize its modular design, giving us flexibility when designing/running experiments (Ruffinelli et al., 2020). Furthermore, LibKGE also provides the framework for KGE models; therefore, we can save time and effort on re-implementation.

3.1 Preliminary results on reproducing

Experimental setup. In their study, Chen et al. (2021) conducted experiments with the ComplEx model, the nuclear 3-norm (N3) regularize, and the Adagrad optimizer. The model was trained for 400 epochs with a binary cross-entropy (BCE) loss function and 1vsAll. Hyperparameter optimization was conducted using a grid search with five different hyperparameters: embedding size (*d*), learning rate (*lr*), batch size (*bsz*), the weight of regularizer (*reg*), and weight of relation prediction (λ). The first four hyper-parameters are shown in Table 3.2. For WN18RR, the weight of relation prediction (λ) was searched over {0.005, 0.001, 0.05, 0.1, 0.5, 1}. For FB15k-237, Chen et al. (2021) searched over {0.125, 0.25, 0.5, 1, 2, 4}. The best configuration for each of the datasets was chosen based on entity MRR. Table 3.3 shows the best configuration found by Chen et al. (2021).

The nuclear 3-norm regularizer was written by Lacroix et al. (2018) as $\|u_r^{(d)}\|_p^3 =$

[1] The embedding size was written follow LibKGE.

Dataset	*d* [1]	*lr*	*bsz*	*reg*
FB15k-237	{200, 1000, 2000}	{0.1, 0.01}	{100, 500, 1000}	{0.0005, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 0}
WN18RR	{200, 1000, 2000}	{0.1, 0.01}	{100, 500, 1000}	{0.005, 0.01, 0.05, 0.1, 0.5, 1}

Table 3.2: Hyperparameter values used by Chen et al. (2021) to find best models

Dataset	**Entity**	**Relation**	*d*	*lr*	*bsz*	*reg*	λ	**MRR**
FB15K-237	Yes	No	2000	0.10	100	0.05	NA	0.372305
	Yes	Yes	2000	0.10	1000	0.05	4.00	0.393722
	No	Yes	2000	0.10	1000	0.0005	NA	0.262888
WN18RR	Yes	No	2000	0.10	100	0.10	NA	0.488083
	Yes	Yes	2000	0.10	100	0.10	0.05	0.490053
	No	Yes	2000	0.10	500	0.5	NA	0.257945

Table 3.3: The best ComplEx configurations found by Chen et al. (2021). The MRR is the entity MRR on the validation dataset.

$\sum_{j=1}^{n_d} |u_{r,j}^{(d)}|^3$. The N3 is implemented by using this formulation of the form:

$$\min_{\substack{(u_r^{(d)})_{d=1\dots3} \\ r=1\dots R}} \sum_{(i,j,k)\in S} [l_{i,j,k}(\sum_{r=1}^{R} u_r^{(1)} \otimes u_r^{(2)} \otimes u_r^{(3)}) + \frac{\mathrm{w}}{3} \sum_{r=1}^{R} (|u_{r,i}^{(1)}|^3 + |u_{r,j}^{(2)}|^3 + |u_{r,k}^{(3)}|^3)] \tag{3.1}$$

where w is the weight for regularization. The N3 was implemented in LibKGE under the name L3 regularizer (Ruffinelli et al., 2020).

3.1.1 Similarly setting

We acquired their best configurations (Table 3.3) and used LibKGE to reproduce the best ComplEx models from Chen et al. (2021). Embedding size (*d*), learning rate (*lr*), batch size (*bsz*), and the weight of regularizer (*reg*) were set exactly as they are in Table 3.4. Alongside that, reciprocal technique and dropout technique were not employed, and the embedding initialization method was normal distribution with a mean of 0 and standard deviation of 0.001

To ensure that we have exactly the same models as they reported on, first, we utilized their codebases[2] to reproduce their best models and observed the performance of those models. Then models obtained from LibKGE were cross-checked

[2] Chen et al. (2021)'s Github: https://github.com/facebookresearch/ssl-relation-prediction

against the models obtained using their codebase. Cross-checking was performed under the ”validation dataset” in three key metrics: MRR, Hits@{1, 3, 10} (Table 3.4).

			Entity Ranking from Chen et al. (2021)'s code base				**Entity Ranking from LibKGE**			
Dataset	**Entity**	**Relation**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
FB15K-237	Yes	No	37.3	27.9	41.0	56.3	35.7	26.6	39.2	54.0
	Yes	Yes	39.3	30.2	43.0	57.3	36.2	27.2	39.7	54.6
	No	Yes	26.7	18.9	29.3	41.8	26.1	18.7	28.3	40.8
WN18RR	Yes	No	48.9	44.7	57.5	50.1	47.5	43.8	48.9	55.2
	Yes	Yes	49.1	45.0	57.6	50.5	47.5	43.7	49.0	55.0
	No	Yes	26.4	21.9	29.5	34.5	43.0	38.4	45.4	51.1

Table 3.4: First attempt to reproduce the best models of Chen et al. (2021) using LibKGE (from the best configurations). The metrics are the filtered entity metrics on the validation dataset.

As seen in Table 3.4, there were huge differences between Chen et al. (2021)'s best models and models produced from LibKGE. For example, the model trained on a hybrid training objective from LibKGE achieved an entity MRR of 36.2%, which is lower than the performance of models from Chen et al. (2021), i.e., entity MRR of 39.3%. Furthermore, with the same hyper-parameter configurations, the model trained on the relation objective from LibKGE performed much better than the model from Chen et al. (2021) on WN18RR (i.e., MRR of 43.0% and MRR of 25.8% , respectively).

Those observations indicate that there should be some differences in the implementation between the two codebases. Thus, it is necessary to perform an open search with previous knowledge about the implementation from Chen et al. (2021).

Hyperparameter	**Value(s)/ Range**
Embedding size	{256, 512, 1024, 2048}
Learning rate	[0.0003, 1.0]
Batch size	{128, 256, 512, 1024}
Weight of regularizer	[1.0e-20, 1.0]
Regularizer	{None, L1, L2, L3}
Dropout	[-0.5, 0.5]
Relation weight	[0.1, 8.0]

Table 3.5: Searching space to find AKBC

Instead of setting the best configuration values for embedding size (*d*), learning rate (*lr*), batch size (*bsz*), and the weight of regularizer (*reg*) as in Table 3.2, we searched for the best model on an open hyperparameters space which also includes the hyperparameters tested by Chen et al. (2021) (Table 3.5). Furthermore, we unrestricted the embedding regularizer to {None, L1, L2, L3} instead of only considering the L3 regularizer. We also employed dropout techniques. Table 3.6 shows the performance of the best models we had founded.

			Entity Ranking from Chen et al. (2021)'s code base				**Entity Ranking from LibKGE**			
Dataset	**Entity**	**Relation**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
FB15K-237	Yes	No	37.3	27.9	41.0	56.3	36.6	27.4	40.0	55.1
	Yes	Yes	39.3	30.2	43.0	57.3	37.3	28.2	41.0	55.5
	No	Yes	26.7	18.9	29.3	41.8	25.8	18.6	28.0	40.3
WN18RR	Yes	No	48.9	44.7	57.5	50.1	47.9	44.1	49.2	55.6
	Yes	Yes	49.1	45.0	57.6	50.5	48.1	44.3	49.4	55.4
	No	Yes	26.4	21.9	29.5	34.5	45.8	42.3	47.3	52.6

Table 3.6: Second attempt to reproduce the best models of Chen et al. (2021) using LibKGE (from a semi-open HPC space). The metrics are the filtered entity metrics on the validation dataset.

As can be seen from Table 3.6, we had some improvement in the model's performance; however, those models could not achieve the MRRs reported by Chen et al. (2021). For example, in FB15k-237, the best model found by LibKGE, which was trained on a hybrid training objective achieved an entity MRR of 37.3% which is 2% lower than the performance of models from Chen et al. (2021) - entity MRR of 39.3%. Furthermore, in WN18RR, the best model founded by LibKGE can achieve filtered entity MRR of 47.9% and 48.1% which are 10% lower than filtered entity MRR from Chen et al. (2021)'s models (48.9% and 49.1% respectively).

Unfortunately, with all of those efforts, LibKGE could not identify exactly the models that Chen et al. (2021) had reported. There must be some difference between the two codebases in terms of implementation. In section 3.2, we describe the main difference between Chen et al. (2021)'s codebase and LibKGE.

3.2 The difference between the two codebases

After comparing the two codebases, we identified the two differences: training dataset and regularization implementation. Instead of training ComplEx directly on the original dataset, the models were trained on reciprocal triples included in the dataset (Chen et al., 2021). Furthermore, L3 regularization was implemented differently between two codebases, and embedding vectors are considered as complex vectors while, in libKGE, we consider them as real vectors. The final results after applying those changes are shown in Table 3.7. As can be seen, we successfully reproduced Chen et al. (2021)'s best models.

			Entity Ranking from Chen et al. (2021)'s code base				**Entity Ranking from LibKGE**			
Dataset	**Entity**	**Relation**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
FB15K-237	Yes	No	37.3	27.9	41.0	56.3	37.2	27.8	40.7	56.2
	Yes	Yes	39.3	30.2	43.0	57.3	39.1	30.0	42.8	57.2
	No	Yes	26.7	18.9	29.3	41.8	26.4	18.7	28.7	41.5
WN18RR	Yes	No	48.9	44.7	57.5	50.1	48.5	44.4	49.8	56.7
	Yes	Yes	49.1	45.0	57.6	50.5	49.1	45.1	50.2	57.2
	No	Yes	26.4	21.9	29.5	34.5	00.1	00.0	00.1	00.1

Table 3.7: Final attempt to reproduce the best models of Chen et al. (2021) using LibKGE (from filling the gap between two codebases). The metrics are the filtered entity metrics on the validation dataset.

3.2.1 Training with reciprocal relations

Reciprocal triplets. The reciprocal relations technique was first introduced by Kazemi & Poole (2018) and Lacroix et al. (2018) to decrease computational cost, and it leads to better performance (Lacroix et al., 2018). The key idea is that instead of using the same scoring function to predict object $(s, p, ?)$ and subject $(?, p, o)$, the prediction is made by two separate scoring functions, one for the prediction of objects and one for the prediction of subjects. To do so, both scoring functions share the same entity embeddings; however, they do not share the same relation embedding (i.e., each relation contains two embeddings).

In order to apply the reciprocal relation technique, Chen et al. (2021) intentionally modified the training dataset as follows[3]: (1) the original dataset is duplicated, (2) those object entities of the duplicated dataset were swapped for the subject entities, and finally (3), the relation ids of those triplets are shifted to new ids by adding the number of relations. Therefore, the number of subject and object entities examined in one epoch is double that of the dataset without reciprocal triplets. Due to duplication, their models were trained only on object prediction, i.e., answering $(s, p, ?)$ question.

[3] For the details of implementation, see the codes: https://github.com/facebookresearch/ssl-relation-prediction/blob/main/src/datasets.py#L51

In LibKGE, preserving the training dataset is mandatory; thus, to apply the reciprocal relation technique, we associate one relation with two embedding vectors. Then, the models is trained on entity prediction, including object prediction $(s, p, ?)$ and subject prediction $(?, p, o)$.

These observations imply that entities (subjects and objects) in Chen et al. (2021)'s codebase are regularized twice as much as in LibKGE. Therefore, the weight of the regularizer (*reg*) needs to be double when used in libKGE. For example, instead of 0.05, 0.0005, 0.1, and 0.5 as shown Table 3.3, the weight of regularization for entity embeddings should be 0.1, 0.001, 0.2, and 1.0, respectively.

Relation prediction in reciprocal relational model. When applying the reciprocal relation technique, we introduce two different embeddings for a single relation: the original relation and reciprocal relation embedding. Therefore, to perform the relation prediction $(s, ?, o)$, it is still an open question whether the models should consider all relations as potential candidates or just consider the original relations and reciprocal relations separately. In their codebase, Chen et al. (2021) decided to consider all of the relations as the potential candidates.

Furthermore, due to the duplication, their models were trained on two queries $(s, ?, o)$ and $(o, ?, s)$. The correct relation for $(s, ?, o)$ would be the normal relation, while the correct relation for $(o, ?, s)$ would be the reciprocal relation.

Missing penalty for reciprocal embedding vectors. When performing penalty calculation for each batch, LibKGE has not penalized the reciprocal embedding while Chen et al. (2021)'s codebase does. Therefore, in this thesis, we also penalize the embedding vector of reciprocal relations.

3.2.2 Regularization implementation

L3 regularization. Instead of scaling them one-third as formulated in Equation 3.1, Chen et al. (2021) decided to omit the scaling part and kept only the sum part. However, in LibKGE, the formula is implemented exactly as it is. Thus, this difference in implementation implies that entities and relation embeddings in their codebase are regularized thrice as much as in LibKGE.

Based on these observations from the training dataset and the L3 implementation, the regularization weight of entity embedding and relation embeddings should be six times for entity and three times for relation. Table 3.8 shows the equivalence of regularization weights between two codebases. As can be seen from that table (Table 3.8), those regularization weights were not explored by Ruffinelli et al. (2020). The highest regularization weights for entity embedding and relation embedding were 0.1 Ruffinelli et al. (2020) which is smaller than 0.15 and 0.3.

			Chen et al. (2021)'s emb. weight		**LibKGE's emb. weight**	
Dataset	**Entity**	**Relation**	**Entity**	**Relation**	**Entity**	**Relation**
FB15K-237	Yes	No	0.05	0.05	0.3	0.15
	Yes	Yes	0.05	0.05	0.3	0.15
	No	Yes	0.0005	0.0005	0.003	0.0015
WN18RR	Yes	No	0.1	0.1	0.6	0.3
	Yes	Yes	0.1	0.1	0.6	0.3
	No	Yes	0.5	0.5	3.0	1.5

Table 3.8: Equivalence of regularization weights between two codebases.

Regularizing embedding vectors. L3/L2 regularization of a embedding vector u are a norm of a vector which is defined as:

$$\|u\|_p^3 = \sum_{j=1}^{n_d} |u_j|^3 \tag{3.2}$$

and

$$\|u\|_p^2 = \sum_{j=1}^{n_d} |u_j|^2 \tag{3.3}$$

When an embedding vector is a complex vector $u \in \mathbb{C}^K$, it composed of a real vector component and an imaginary vector component, thus, the size n_d of an embedding vector is doubled. Let denote $u_j = u'_j + iu''_j$, where u'_j is the real part and u''_j is the imaginary. Therefore the norm of a complex number is defined as $|u_j| = \sqrt{u'^2_j + u''^2_j}$. Therefore, the L3 regularization of a complex vector is defined:

$$\|u\|_p^3 = \sum_{j=1}^{n_d} |u_j|^3 = \sum_{j=1}^{n_d} \left(\sqrt{u'^2_j + u''^2_j}\right)^3 \tag{3.4}$$

However, when we use L2 regularization of a complex vector, equation 3.3 will be transformed to

$$\|u\|_p^2 = \sum_{j=1}^{n_d} |u_j|^2 = \sum_{j=1}^{n_d} \left(\sqrt{u'^2_j + u''^2_j}\right)^2 = \sum_{j=1}^{n_d} u'^2_j + \sum_{j=1}^{n_d} u''^2_j \tag{3.5}$$

which is exactly similar as we apply L2 norm to a vector in a real vector space.

This could be considered as the main difference between LibKGE and codebase from Chen et al. (2021). In LibKGE, when applying L3, we consider the entities and relations as embedding vectors in a real vector space[4]. While, Chen et al. (2021) and Lacroix et al. (2018) consider embedding vectors in ComplEx model as a complex vector. However, we have same agreement between two codebases when applying L2 regularization.

3.2.3 Ablation study: Unexplored weights and complex vectors

From those observations, the main difference between two codebases is the how the embedding vectors are considered when calculating penalty. Chen et al. (2021) consider embedding vector in ComplEx model as complex vector while Broscheit et al. (2020) consider them as a normal embedding vector. Additionally, Chen et al. (2021) searched their best models with the weights that have not been explored by Ruffinelli et al. (2020). Therefore, it is essential to studied the effect of each change: (i) unexplored hyperparameter searching space and (ii) considering embedding vectors as complex vectors when calculating penalty term.

The results for ablation study are shown in Figures 3.1 and 3.2 respectively. The orange represents the performance of models trained on hybrid training objective, while the blue represent for models trained on standard training objective. The green is for models trained on relation training objective. The lines illustrate the performance from Chen et al. (2021)'s models. The ”type”-axis represents the changes that were applied to the models. ”Normal weight” means that models were trained with the same-value weights as reported in the papers. ”Scaled weight” means the regularization weights were scaled based on the difference in the implementation between two codebases. ”Complex emb. vector” means that embedding vectors were considered as complex vector when performing penalty calculation.

On FB15K-237, the Figure 3.1 demonstrates, by using reciprocal relation technique, the performance of model on entity prediction (entity MRR) was boosted, i.e., MRR increases 1% or 2% (from 36.2% to 37.7%). Furthermore, with unexplored regularization weights, the models could achieve must more better results (37.7% to 38.6% MRR).

However, the main difference between LibKGE and Chen et al. (2021)'s codebase, i.e., considering embedding vector as complex vectors could slightly improve

[4] https://github.com/uma-pi1/kge/blob/master/kge/model/embedder/lookup_embedder.py#L139

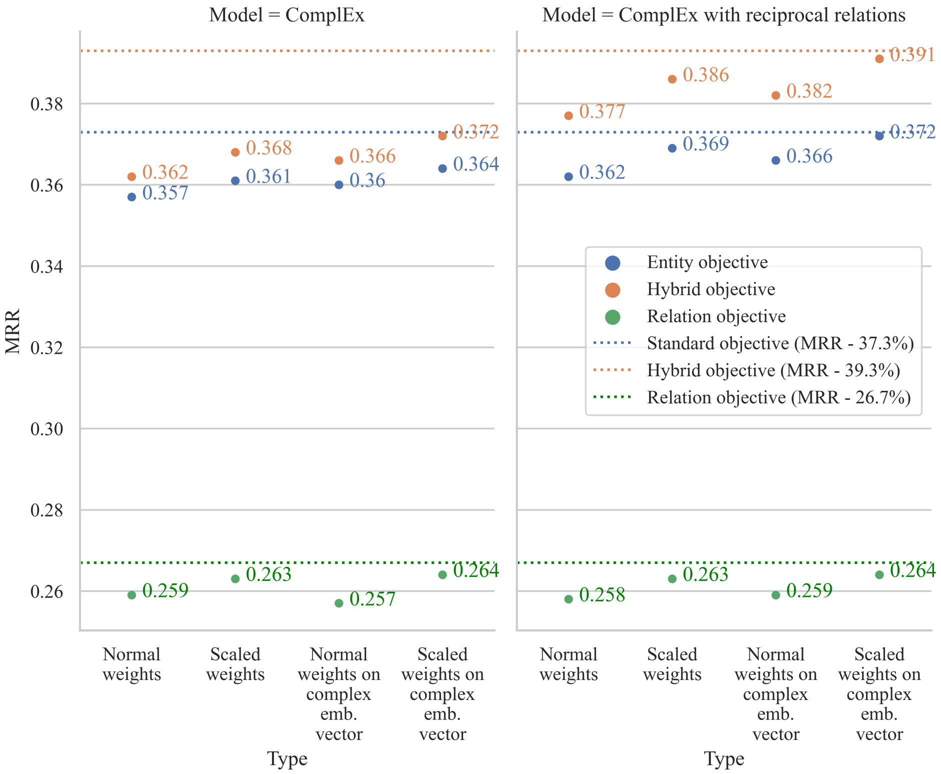

Figure 3.1: Performance of model when applying changes on FB15K-237. *Type:* indicates the changes has been applied

MRR by around 0.5%. Clearly seen that, the main improvement comes from unexplored regularization weights and it does not matter how we treat the embedding vector as complex vector or not.

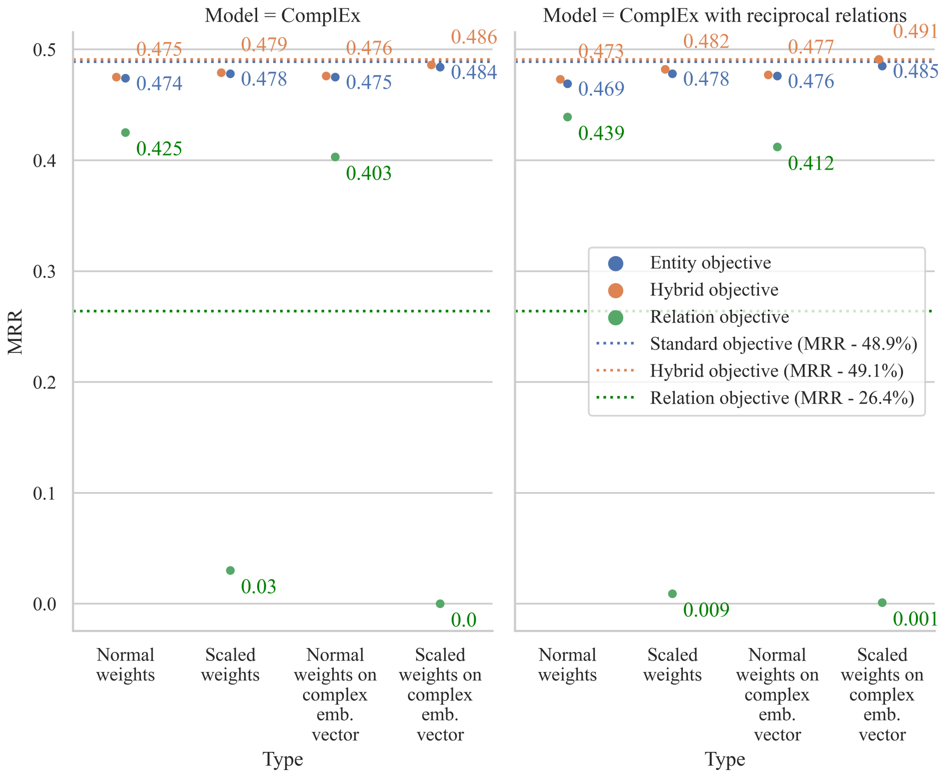

Figure 3.2: Performance of model when applying changes on WN18RR. *Type:* indicates the changes has been applied

On WN18RR, Figures 3.2 states that the improvement is not really clear. However, we can see the same pattern as we observed on FB15K-237, with the unexplored regularization weights, the model trained on standard and hybrid objectives could achieve much more better results in entity MRR, that is 48.6% and 48.4% vs 47.6% and 47.5% without reciprocal technique and 49.1% and 48.5% vs 47.7% and 47.6% with reciprocal relations.

Complex embedding vector in unrestricted HPC space. The HPC space from Chen et al. (2021) is a restricted space with numerous predefined hyperparameters such as regularizer, embedding initialization method or training optimizer. Therefore, we conducted an experiment on an unrestricted HPC space to observe the effect of considering embedding vectors as complex vectors in penalty calculation. The unrestricted HPC space we chose is from Ruffinelli et al. (2020).

Dataset	**Relation**	**Entity**	**Complex vector**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
FB15K-237	No	Yes	Yes	34.9	26.2	38.0	52.3
	No	Yes	No	35.0	26.1	38.3	52.8
	Yes	Yes	Yes	35.0	26.6	38.0	51.5
	Yes	Yes	No	35.1	26.4	38.2	52.9
WN18RR	No	Yes	Yes	47.6	44.5	48.6	54.0
	No	Yes	No	47.6	44.4	48.9	53.8
	Yes	Yes	Yes	46.2	43.5	47.3	51.2
	Yes	Yes	No	46.5	43.9	47.5	51.5

Table 3.9: The performance of models in FB15K-237 and WN18RR with and without considering embedding vectors as complex vectors. The *Complex vector* indicates that considering the embedding vector as complex vector in penalty calculation. *Yes* value means model considered embedding vectors as complex vectors while *No* value means model does not consider embedding vectors as complex vectors.

Table 3.9 reports the models' performance with and without considering embedding vectors as complex vectors on unrestricted HPC space. From the table we can note that there is no difference between with or without the consideration. Therefore the main improvement come from unexplored values in the HPC space.

3.3 Ability of finding Chen et al. (2021)'s best models

In this chapter, we reported the differences between two codebase and we have found exactly the configuration to reproduce the best models from Chen et al. (2021) using LibKGE. However, the main question now is that we can find those models using Random Search approach in LibKGE or those models that are too good to be found.

Due to the huge difference between restricted HPC space from Chen et al. (2021) and unrestricted HPC space from Ruffinelli et al. (2020), we relaxed the hyperparameter bit by bit. Within this scope of the study, we only focus on reproducing the model trained on the hybrid training objective, the best-performance

Trial	Hyperparameter	Values	entity MRR
1	Embedding initialization	Normal	39.1
	Std. deviation (Normal)	0.001	
	Lp regularization	L3	
	Entity emb. weight	[0.0005, 1.0], log scale	
	Relation emb. weight	[0.0005, 1.0], log scale	
2	Embedding initialization	Normal	39.0
	Std. deviation (Normal)	0.001	
	Lp regularization	{L1, L2, L3, None}	
	Entity emb. weight	[0.0005, 1.0], log scale	
	Relation emb. weight	[0.0005, 1.0], log scale	
2	Embedding initialization	Normal	38.9
	Std. deviation (Normal)	0.001	
	Lp regularization	{L1, L2, L3, None}	
	Entity emb. weight	[0.0005, 1.0], log scale	
	Relation emb. weight	[0.0005, 1.0], log scale	

Table 3.10: Model performance when relaxing the restricted HPC space. The *Trial* indicates the id of trial, *Hyperparameters* are the hyperparameters we attempted to relax with their corresponding values in the *Value* column. Entity MRR indicates the MRR that the best model can achieve.

model was selected on entity MRR. The model was trained on FB15K-237. Table 3.10 shows the performance of the best models we found after 3 times relaxing the restricted HPC space (on entity MRR).

As illustrated in the table, so far we still can find the best model from Chen et al. (2021). However, the range for regularization weight is still small $[0.0005, 1.0]$ compared to unrestricted HPC space i.e., close interval $[1.0^{-20}, 1.0^{-01}]$. Therefore, the future study should enlarge the close interval for regularization weight to test the finding ability of LibKGE.

Chapter 4

Comparative study

In this chapter, we reported on the design and results of our comparative study. We mainly compared the performance of the ComplEx model in different model selection and training objectives, but, in a similar setting. This chapter aims to answer two questions: (1) Does hybrid training improve the performance of a model in relation prediction? (2) What is the effect of using overall MRR instead of entity MRR in model selection?

4.1 Experimental setup

Dataset. Despite the popularity of the FB15K dataset, we used the FB15K-237 and WN18RR in our study because of three reasons: (i) they are ”hard” datasets since they were explicitly designed to evaluate multi-relational link prediction, (ii) they are diverse in that models often yield different performance, (iii) they are of reasonable size for a large study (Ruffinelli et al., 2020).

Models. Within the scope of this thesis, we mainly focused on the ComplEx model due to excessive training costs (Trouillon et al., 2016). Focusing on the ComplEx, we can directly compare our results with Chen et al. (2021) and Ruffinelli et al. (2020). Furthermore, this model is also one of the most well-known KGE models, besides RESCAL (Nickel et al., 2011), TransE (Bordes et al., 2013b), and DisMult (Yang et al., 2014).

Evaluation metric. We reported filtered MRR and filtered Hits@$\{1, 3, 10\}$ on entity prediction and relation prediction. We used the validation dataset for analysis.

Loss function. In this thesis, we used the CE loss function so as to be able to compare with Chen et al. (2021)'s results and since Ruffinelli et al. (2020) have shown that the use of CE yields to better configuration than other loss functions.

Relation weight. In their experimental setup, Chen et al. (2021) searched the weight of relation prediction over set $\{0.005, 0.001, 0.05, 0.1, 0.5, 1\}$ for WN18RR, and for FB15k-237, they searched over set $\{0.125, 0.25, 0.5, 1, 2, 4\}$. To have only one comprehensive HPC space for both datasets, we combined both sets of weights and searched relation prediction over a closed interval $[0.005, 4.0]$, and we sampled the weights uniformly instead of using a log scale.

Hyperparameters configuration (HPC) space. Our HPC space bears a close resemblance to the HPC space from Ruffinelli et al. (2020). The reason we chose their HPC space is that this hyperparameter space is not excluded from a prior hyperparameter and contains salient points. However, in our searching space, instead of using all three primary training objectives (NegSamp, 1vsAll, and KvsAll), we only focused on 1vsAll, so that we could directly study the effect of hybrid training proposed by Chen et al. (2021). Furthermore, embedding vectors were considered real embedding vectors instead of complex vectors when performing penalty calculations. The details of the HPC space can be found in Table B.1 in Appendix B.

We aware that Chen et al. (2021)'s HPC space could yield better models than Ruffinelli et al. (2020)'s HPC space in terms of entity ranking. However, due to a lack of resources and time, we had conducted the comparative study simultaneously with reproducing study, therefore the preliminary results from Chen et al. (2021)'s HPC were discussed in Section 4.4. Furthermore, we note that the searching space from Chen et al. (2021) is restricted which could lead to bias, thus, the main focus of this comparative study is the HPC from Ruffinelli et al. (2020).

Training. All models were trained for a maximum of 400 epochs. Models were validated every five epochs on validation and we performed early stopping with patience of 50 epochs which is similar to the setup of Ruffinelli et al. (2020).

Metrics. In this thesis, the filtered MRR and the filtered Hits@$\{1, 3, 10\}$ are the main metrics used to evaluate the models in relation prediction and entity prediction, so we can be able to compare results with Chen et al. (2021) and Ruffinelli et al. (2020). The MR is not adopted, since it is sensitive to outliers (Nickel et al., 2016b).

Ranking in relation prediction task. When applying the reciprocal technique, the number of relations is doubled and the score functions of (i, k, j) and (j, k, i) differs. Therefore, given a true triple (i, k, j), we note that there are two potential answers which depend on the order of subject and object in relation prediction tasks. However, to preserve the validation dataset, we only validated the $(i, ?, j)$ query and ignored the validation for the inverse query $(j, ?, i)$. Furthermore, all the relations including the reciprocal ones are considered during the evaluation protocol.

Model selection. We studied the effect of model selection on three different MRRs: the entity MRR, the relation MRR and the overall MRR which is micro-average of subject, object and relation MRRs. All of them are filtered MRRs.

KGE Framework: All of the following results were conducted by using the LibKGE. In order to run experiments on LibKGE framework, a new relation mechanism has been implemented, for the details please consult Table A.1.

4.2 Training objectives: entity, hybrid, and relation training objectives

The effect of three training objectives: entity, hybrid, and relation training objectives on relation prediction performance (i.e., filtered relation MRR) is the main discussion in this section. We selected the best-performance models based on filtered entity MRR. Table 4.1 shows the performance of the best models on the relation prediction task as well as the entity prediction task. The performance of the models are represented in the filtered MRR and the filtered Hits@{1, 3, 10} metrics.

On the FB15K-237 dataset, as evidenced by Table 4.1, by including negative examples which are obtained by perturbing the relation positions into 1vsAll (i.e., hybrid training objective), the performance of the model in terms of relation prediction was significantly improved. Model trained on hybrid objective achieved relation MRR of 95.4% which is 4 times higher than model trained on standard objective i.e., relation MRR of 21.8%.

However, the effect of the hybrid training objective was not observed on WN18RR data, the performance of the model trained on a hybrid training objective was a re-

			Entity prediction				Relation prediction			
Dataset	**Entity**	**Relation**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
FB15K-237	Yes	No	35.2	26.4	38.5	52.7	21.8	09.8	17.3	58.1
	Yes	Yes	35.0	26.0	38.4	52.9	95.4	93.6	96.9	98.3
	No	Yes	23.7	16.7	25.6	37.9	97.2	95.6	98.9	99.6
WN18RR	Yes	No	46.9	43.8	48.0	52.7	76.0	66.1	80.5	99.9
	Yes	Yes	46.6	43.3	48.4	52.6	67.2	54.5	75.4	85.9
	No	Yes	43.6	41.2	44.5	48.0	63.1	49.5	71.5	82.3

Table 4.1: Model performance in standard, hybrid, and relation training objectives on FB15K-237 and WN18RR. The best models were selected on entity MRR.

lation MRR of 67.2% which is 10% lower than the relation MRR of the model trained with a standard objective i.e., relation MRR of 76.0%. Furthermore, we observed that the training model by hybrid objective lowered the entity MRR from 46.9% to 46.6% - 3% lower.

Table 4.1 additionally shows the models' performances when the models were trained using negative relation examples only (relation training objective). By applying the relation training objective, the ComplEx model can achieve the relation MRR of 97.2% which is 2% higher than the performance of the model trained on hybrid training on FB15K-237. However, the performance on entity ranking dropped significantly to 23.7% on entity MRR from 35.0%.

Furthermore, on WN18RR, there was a decrease in the relation MRR performance when the model was trained on the relation objective (i.e., from 67.2% to 63.1%), the same drop was also observed in the entity MRR (from 46.6% to 43.6%). These trends were also observed in Hits@$\{1, 3, 10\}$ metrics.

One potential explanation for non-improvement when including negative relation examples in WN18RR would be that the dataset contains comparatively a small number of relation $|\mathcal{R}| = 11$ while the number of relations in FB15K-237 $|\mathcal{R}| = 237$. This observation was also reported by Chen et al. (2021). They found that hybrid training brings benefits to datasets with a larger number of predicates (Chen et al., 2021).

Relation prediction performance of Chen et al. (2021)'s models. Table 4.2 shows the performance of Chen et al. (2021)'s best models on relation prediction and entity prediction. The results were produced using the LibKGE framework. The table, once again, nicely illustrates the effect of hybrid training objective in relation prediction.

There is not only the improvement in entity MRR, the table 4.2 also shows the

			Entity prediction				Relation prediction			
Dataset	**Entity**	**Relation**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
FB15K-237	Yes	No	37.2	27.8	40.9	56.3	90.2	85.0	94.7	97.6
	Yes	Yes	39.1	30.0	42.7	57.2	95.4	92.1	98.8	99.6
	No	Yes	26.2	18.7	28.7	40.7	95.0	91.3	98.4	99.5
WN18RR	Yes	No	48.6	44.4	49.8	57.1	76.8	70.3	79.5	89.0
	Yes	Yes	48.6	44.6	49.7	57.3	62.9	42.7	79.0	88.6
	No	Yes	14.1	08.3	18.2	24.3	24.0	00.0	38.4	49.5

Table 4.2: The Chen et al. (2021)'s best model performance on entity prediction and relation prediction. The results were produced by using LibKGE.

improvement in relation MRR when the model was trained on hybrid training objective. Compared with the standard training objective, the hybrid model achieved relation MRR of 95.2% which is 5% higher than the model trained on standard objective (relation MRR of 90.2%).

Although, the hybrid training objective brings benefits to FB15K-237, however, we observed a 14% dropout in WN18RR when we trained model on hybrid training objective (from 76.8% to 62.9%). This observation strengthen the point of view that the hybrid training objective can bring improvement in relation MRR to those datasets which have a significant number of predicates.

Another interesting observation from the Chen et al. (2021)'s models is that those models trained only on relation training objective could not achieve better results than model trained on hybrid training objective. This observation is contradicted to what we observed in unrestricted HPC space (Table 4.1)

Furthermore, when comparing the performance of hybrid training models found in unrestricted HPC space with the Chen et al. (2021)'s best models (Table 4.1 and 4.2), we observed that there is no difference between those two models in relation prediction performance. This indicates that the Chen et al. (2021)'s best models are more suitable for entity prediction than relation prediction.

Positive impact of hybrid training. The evidence from observing the performance of models trained on hybrid training objective points towards the idea that this training objective could bring a nice benefit not only for entity prediction (on Chen et al. (2021)'s HPC space), but also for relation prediction (on both HPC spaces), although we chosen the best models in favor for entity prediction. Therefore, in the next section, we attempt to observe the performance of models when we select the best models in favor for both entity prediction and relation prediction (overall MRR).

4.3 Model selection: entity MRR, overall MRR, relation MRR

In the previous section, we observed that using a hybrid training objective leads to improvement of the model's performance in relation prediction. However, the best models were selected based on the entity MRR which is mainly focusing on the performance of mode in entity ranking. Therefore, in this section, the models will be selected on overall MRR which is the micro-average between subject MRR, object MRR and relation MRR (details in Equation 2.1). Table 4.3 shows the comparison between two model selection approaches, the performance was represented by filter MRRs (for other metrics, see Table C.1).

			Selected on entity MRR		**Selected on overall MRR**	
Dataset	**Entity**	**Relation**	**entity MRR**	**relation MRR**	**entity MRR**	**relation MRR**
FB15K-237	Yes	No	35.2	21.8	33.5	89.4
	Yes	Yes	35.0	95.4	34.6	96.8
	No	Yes	23.7	97.2	24.0	97.4
WN18RR	Yes	No	46.9	76.0	45.2	88.8
	Yes	Yes	46.6	67.2	42.6	80.8
	No	Yes	43.6	63.1	42.7	63.5

Table 4.3: Model performance in three training objectives. *Selected on entity MRR:* The best models were selected based on models' performance in entity MRR metric. *Selected on overall MRR:* The best models were selected based on models' performance in overall MRR metric.

As was mentioned in the previous section (section 4.2), the model trained on standard training objective can only achieve 21.8% of relation MRR. However, as can be seen in Table 4.3, without training from scratch, we can obtain a better model in relation prediction (89.4% in relation MRR) by selecting the best models on overall MRR instead of on entity MRR. Interestingly, this positive effect also can be observed on WN18RR also. The best model can achieve up to relation MRR of 88.8% instead of 76.0%. However, there is a trade-off when selecting best-performance models on overall MRR, the model's performance in entity prediction dropped notably by around 2% on both dataset.

To overcome that trade-off, as shown in 4.3, if we re-train models on hybrid training objective, the drop in entity ranking become smaller, instead of 2%, now it is only about 0.4%. Furthermore, the model can now even achieve higher relation MRR (96.8% instead of 95.4% on FB15K-237).

Unfortunately, that trend was not observed on WN18RR. We still observe the non-improvement even if we select a model based on overall MRR. The performance reduce from 88.8% to 80.8% along the reduce in entity ranking performance also from 45.2% to 42.6%

Based on those observations, we can conclude that the model selection positively affects the model's performance if a model is trained on hybrid training objective on unrestricted HPC space.

Relation MRR: It is easy to see from Table 4.4, selecting a model on relation MRR brings better performance on relation prediction, but, at the same time, the models underperformed on entity prediction.

			Selected on entity MRR		**Selected on overall MRR**		**Selected on relation MRR**	
Dataset	**Entity**	**Relation**	**entity MRR**	**relation MRR**	**entity MRR**	**relation MRR**	**entity MRR**	**relation MRR**
FB15K-237	Yes	No	35.2	21.8	33.5	89.4	30.9	92.9
	Yes	Yes	35.0	95.4	34.6	96.8	32.0	97.3
	No	Yes	23.7	97.2	24.0	97.4	17.9	97.8
WN18RR	Yes	No	46.9	76.0	45.2	88.8	45.6	89.4
	Yes	Yes	46.6	67.2	42.6	80.8	40.0	89.3
	No	Yes	43.6	63.1	42.7	63.5	02.1	90.8

Table 4.4: Model performance in three training objectives. *Selected on entity MRR:* The best models were selected based on models' performance in entity MRR metric. *Selected on overall MRR:* The best models were selected based on models' performance in overall MRR metric. *Selected on relation MRR:* The best models were selected based on models' performance in relation MRR metric.

4.4 Primary results Chen et al. (2021)'s HPC space

Due to the restricted time for a master thesis, we conducted the comparative study simultaneously with the reproducing phase (Chapter 3). Therefore, the awareness about reciprocal techniques and the difference in L3 implementation were absent on the time this experiment conducted. Thus, we studied the Chen et al. (2021)'s HPC space without applying the reciprocal technique. Furthermore, when calculate the regularization for embedding, we consider embedding vectors as real vector. i.e., the vectors in real vector space.

Semi-restricted HPC space. We would like to call this HPC space is semi-restricted HPC space due to the following restrictions. To imitate the Chen et al.

(2021)'s HPC space, we use Adagrad as optimizer, learning rate without log scale, and normal distribution as embedding initialization methods with $\mu = 0$ and $\sigma = 0.001$. Additionally, penalty for the embedding of an object is weighted by the object's frequency in the training data i.e., the frequency weighting is true. For another hyperparameters, we keep them as same as values in Ruffinelli et al. (2020)'s HPC space such as learning scheduler, the number of epochs and the early-stopping criteria (see Table B.1 in Appendix B for more details).

For the important hyperparameters, we merged the values from Ruffinelli et al. (2020)'s HPC space and values from Chen et al. (2021)'s HPC space together. Furthermore, the searching range for relation weight is enlarged from $[0.005, 4.0]$ to $[0.1, 8.0]$. Table 3.5 shows the values for important hyperparameters. This is also the HPC space that we used to find Chen et al. (2021)'s best models (in section 3.1.1) and we also unitized this HPC space to study impact of hybrid training on larger embedding size.

			Selected on entity MRR		**Selected on overall MRR**		**Selected on relation MRR**	
Dataset	**Entity**	**Relation**	**entity MRR**	**relation MRR**	**entity MRR**	**relation MRR**	**entity MRR**	**relation MRR**
FB15K-237	Yes	No	36.6	94.7	36.5	94.9	35.5	95.0
	Yes	Yes	37.3	97.9	37.1	98.0	36.5	98.0
	No	Yes	25.8	96.2	25.8	96.4	17.3	97.8
WN18RR	Yes	No	47.9	83.7	48.0	83.9	48.1	84.8
	Yes	Yes	48.1	86.0	48.1	86.1	36.3	90.2
	No	Yes	45.8	80.9	45.2	87.8	40.0	89.2

Table 4.5: Model performance in three training objectives on semi-restricted HPC space. *Selected on entity MRR:* The best models were selected based on models' performance in the entity MRR metric. *Selected on overall MRR:* The best models were selected based on models' performance in the overall MRR metric. *Selected in relation MRR:* The best models were selected based on models' performance in relation MRR metric. (for other metrics, see Table C.2)

Table 4.5 illustrates the benefit of using hybrid training objective which are consistent with previous results illustrated in Table 4.5 and Table 4.4. Surprisingly, on this semi-restricted HPC space, the effect of model selection is not observed anymore.

We can clearly observe that the performance of models on entity and relation prediction are consistent regardless of model selection. When we trained a model on hybrid training objective on FB15K-237, the model can achieve 37.3% entity MRR and 97.9% relation MRR which are higher than entity MRR and relation MRR standard model can achieve (36.6% and 94.7% resp.).

Unsurprisingly, by comparing Table 4.5 and Table 4.4, the semi-restricted HPC space yield better models than unrestricted HPC space (i.e., the HPC space from Ruffinelli et al. (2020)).

4.5 Conclusion

Our comparative study has led us to conclude that hybrid training objective could bring a positive effect to ComplEx model not only on predicting entity but also relation prediction. Furthermore, model selection could become handy to find the model trained on standard objective that can work well on both relation and entity prediction tasks without retraining from scratch.

Chapter 5

Error Analysis

In this chapter, in an attempt to determine the limitation of the model trained on the standard training objective and improvements that hybrid training brings to relation prediction, we will mainly study the best models trained on hybrid and standard training objectives only. The models that we examined are from the unrestricted HPC space. We note that semi-unrestricted HPC space yielded better models; however, we have not fully explored that HPC space. Therefore, future work should concentrate on that HPC space.

Additionally, we only examine the best models selected based on entity MRR and overall MRR. Those two model selection methods are chosen because we can find the best performance on relation and entity prediction using those two. We could not ignore the importance of entity prediction, although the thesis is about relation prediction.

For simplicity, the *"standard model"* is used to refer to a model trained on a standard training objective. The best standard model which is selected based on entity MRR will be referred to as the *"standard entity model.* "Next, the best standard model selected on overall MRR will be called as *"standard overall model"*. In this chapter, the standard entity model is our *"baseline model"*.

If a model is trained on a hybrid training objective, we will refer them to a *hybrid model*. The *hybrid entity model* will be referred to as the best hybrid model selected on entity MRR and the *hybrid overall model* is used to name the best hybrid model selected on overall MRR.

Within this chapter, the FB15K-237 dataset is used to perform the error analysis. We chose this particular HPC space because we observed the improvement when applying the hybrid training objective.

5.1 Standard entity model and its limitation

To reveal the limitations of baseline model (i.e., standard entity model) on relation prediction task, we attempted to calculate the relation MRR for each relation. More precisely, for a specific relation k, we extracted all triples in a validation data $\mathcal{V}$ (or training data $\mathcal{T}$) that are formed by that relation k i.e.,

$$\mathcal{D}_{j,\mathcal{V}} = \{(i,k,j)|i,j \in \mathcal{E} \vee (i,k,j) \in \mathcal{V}\}$$

or

$$\mathcal{D}_{k,\mathcal{T}} = \{(i,k,j)|i,j \in \mathcal{E} \vee (i,k,j) \in \mathcal{T}\}$$

. Then we calculated the filtered relation MRR for that relation k based on all of those triples in set $\mathcal{D}_{k,\mathcal{V}}$ (or $\mathcal{D}_{k,\mathcal{T}}$). Formally, for example, the relation MRR for that relation k on validation data will be calculated by using this formula:

$$\text{Filtered MRR}_{\text{relation, } j,\mathcal{V}} = \frac{1}{|\mathcal{D}_{j,\mathcal{V}}|} \sum_{(i,k,j)\in\mathcal{D}_{j,\mathcal{V}}} \left(\frac{1}{\text{rank}(k|i,j)}\right) \tag{5.1}$$

We performed the calculation in validation and training data independently.

In their paper, Chang et al. (2020) suggested analyzing the model performance in each relation type (i.e., M-M, M-1, 1-N, and 1-1). The main difference between those four types is the number of entities that a subject or an object can associate with. For example, the basic difference between M-N and M-1 relation types is the number of objects that a subject can associate with, e.g., M-N relation - ”lived place” and M-1 relation - ”place of birth”. A person can live in more than one place, however, there is only one birthplace for a person. Figure 5.1 shows the distribution of filtered relation MRRs for each relation type and model.

Unexpectedly, from Figure 5.1 we can see that the standard entity model showed a wide dispersion in M-1 relations in which the the third quartile MRR is 97.9% and the first quartile MRR is 33.0% in training dataset. However, for the remain types, the baseline model could achieve low MRRs i.e., the median of M-N, 1-N, and 1-1 distributions are 25.8%, 25.0%, and 29.0% respectively in training dataset. Undoubtedly, we also observed the wide dispersion in M-1 relation and low medians in M-N, 1-N, and 1-1 the relations on validation data. Those findings indicates that the model seems to be biased toward the M-1 relations. Therefore, it is necessary to observe the common ranking patterns which the baseline model ranked in four relation types (i.e., M-M, M-1, 1-N, and 1-1).

Table 5.1 shows the top three ranking patterns as well as their percentage. The ranking pattern is a tuple of the relation types from top three ranked relations;

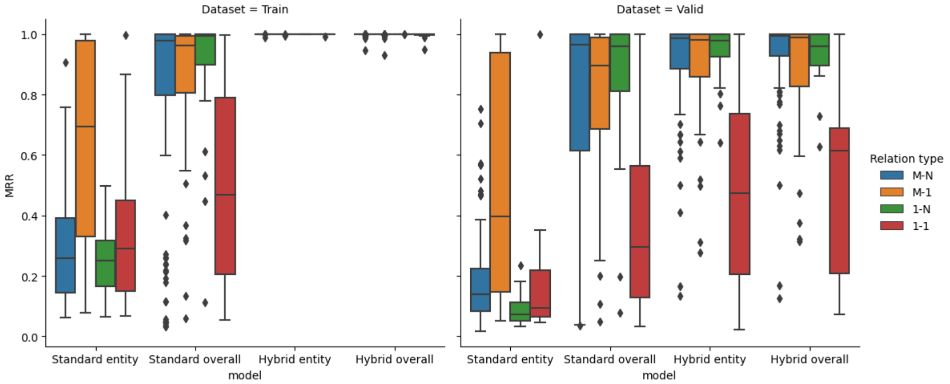

Figure 5.1: Distribution of filtered relation MRR on validation and training data over relations for four relation types (M-M, M-1, 1-N, and 1-1).

for example, the (M-1, 1-1, 1-N) pattern means that the first rank relation is M-1 relation, then the second rank relation is 1-1 relation, and third rank relation is 1-N relation.

True relation type: M-N		**True relation type: M-1**		**True relation type: 1-N**		**True relation type: 1-1**	
Top 3	**%**	**Top 3**	**%**	**Top 3**	**%**	**Top 3**	**%**
(M-1, M-1, M-1)	34.71	(M-1, M-1, M-1)	63.00	(M-1, M-1, M-1)	33.96	(1-1, 1-1, 1-1)	65.74
(1-1, 1-1, 1-1)	14.65	(M-1, M-1, 1-N)	7.97	(M-1, M-1, 1-N)	10.52	(M-1, 1-1, 1-1)	8.71
(M-1, M-1, M-N)	5.59	(M-1, 1-N, M-1)	7.35	(M-1, 1-N, M-1)	6.97	(1-1, M-1, 1-1)	5.33

Table 5.1: Three top-3 predictions for each relation type (training)

Table 5.1 strongly underlines the bias of model toward the M-1 relations. The M-1 relations are, most of the time, on the top of ranking regardless of the relation type of the true relation (except 1-1 relation type). For instance, if the actual relation is a M-N relation, there is a 34.71% chance that we observed the first, second, and third rank relations being M-1 relations. Same for if the actual relation is 1-N relation, there is 33.96% that we observed the first, second, and third rank relations being all M-1 relations. Table 5.2 shows three examples of ranking results from baseline model.

Through this example, we note that there exists some relations that share the same object and/or objects domain together. For instance, the subject for ”lived places” and ”place of birth” relations is the set of name of people, while their set of object is the set of cities or places. Those domain-sharing relations could be

Subject	Relation	Object	Rank	Type
Joe Hisaisahi	/people/person/places_lived./people/place_lived/location	Nagano Prefecture	6	M-N
	/people/person/place_of_birth		1	M-1
	/time/event/instance_of_recurring_event		2	M-1
	/education/educational_institution/campuses		3	1-1
Arnold Schoenberg	/people/person/places_lived./people/place_lived/location	Vienna	8	M-N
	/people/person/place_of_birth		1	M-1
	/people/deceased_person/place_of_death		2	M-1
	/base/biblioness/bibs_location/country		3	M-1
Cancer (Disease or medical condition)	/people/cause_of_death/people	Laurence Olivier	12	1-N
	/base/schemastaging/person_extra/net_worth./measurement_unit/dated_money_value/currency		1	M-1
	/film/person_or_entity_appearing_in_film/films./film/personal_film_appearance/type_of_appearance		2	M-1
	/film/film/estimated_budget./measurement_unit/dated_money_value/currency		3	M-1

Table 5.2: Prediction examples from standard entity model. The first relations on the top of each box are the true relations while the remain ones are the predicted relation. *Joe Hisaishi was born in Nagono and Arnold Schoenberg was also born in Veinna, however, those information was not captured in FB15K-237, so those triples are considered to be negative triples in this study.*

considered as hard-cases for relation predictions.

Formally, a *domain-sharing relations* is a relations such that there exists another relation(s) in knowledge graph which share the same subject and object; for example, Mr.A lived in Ha Noi and Mr.A was born in Ha Noi. On the other hand, *non domain-sharing relations* are those relations which do not have relation sharing the same subject and object in data. In FB15k-237, there are totally 160 domain-sharing relations which takes about 67.5% total relations in dataset. However, there are only 29160 triples which are formed from 160 domain-sharing relations, those triples take 10% of total triple in the training dataset. Those triples could be considered as the corner-cases.

Figure 5.2 shows the distributions of filtered relation MRRs for domain-sharing and non domain-sharing relations on validation and training dataset. Perhaps unsurprisingly, we found from Figure 5.2 that baseline mode tends to perform better on non domain-sharing relations than on domain-sharing relations. The median MRRs of non domain-sharing relation are always higher than median MRRs of domain-sharing relation.

We attempt to measure the similarity between the top-rank relation k' and the actual relation k based on domain similarity. To do so, we calculated the Jaccard similarity between (1) the subject sets between those two relation and between the object sets of those two relations. Then the similarity between two relation is the average of subject and object Jaccard similarities. The set of subjects (or objects) of a relation $\mathcal{E}_{s,k'}$ (or $\mathcal{E}_{o,k'}$) is defined as $\mathcal{E}_{s,k'} = \{i' | i' \in \mathcal{E} \vee (i', k', *) \in \mathcal{T}\}$ (or $\mathcal{E}_{o,k'} = \{j' | j' \in \mathcal{E} \vee (*, k', j') \in \mathcal{T}\}$), where $\mathcal{T}$ is the training dataset. Thus, the

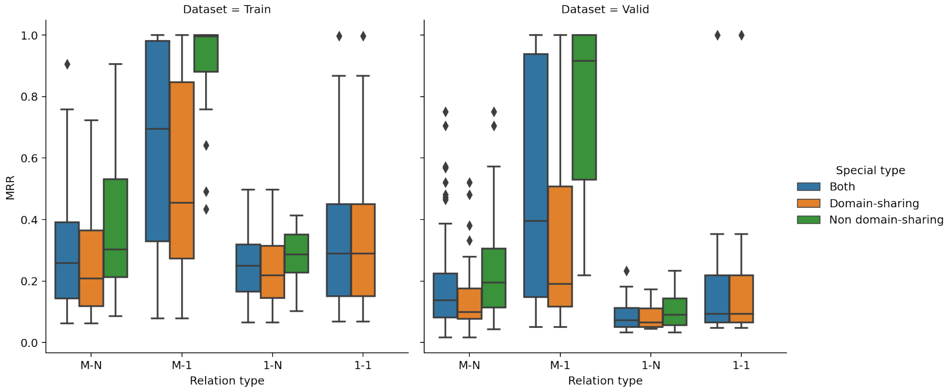

Figure 5.2: Distribution of filtered relation MRR on validation and training data over relations for for non domain-sharing and domain-sharing relations (standard entity model)

similarity between two sets of subject, for example, is calculated using this formula

$$J(\mathcal{E}_{s,k}, \mathcal{E}_{s,k'}) = \frac{|\mathcal{E}_{s,k} \cap \mathcal{E}_{s,k'}|}{|\mathcal{E}_{s,k} \cup \mathcal{E}_{s,k'}|}$$

.

The Figure 5.3 shows the the distribution of average similarity between ranked relation and actual relation for domain-sharing relations and non domain-sharing relations. Predictably, the top-rank relations are not similar to actual relation in term of domain similarity except M-1 relation type. There are, however, some interesting finding across relation types. We found that baseline model could rank relations which are similar to the actual relation higher for non domain-sharing relation, especially for M-1 relations, while, for domain-sharing relations, the similarity between high-rank relations and the corresponding actual relations are lower. Furthermore, the distribution of similarity for M-1 relations are more dispersed.

In sum, based on observation in Figure 5.3 and 5.2, we found that baseline model could not capture the domain similarity, especially domain similarity of domain-sharing relations and the model tends to rank M-1 relations higher than other types regrading the type of actual relation.

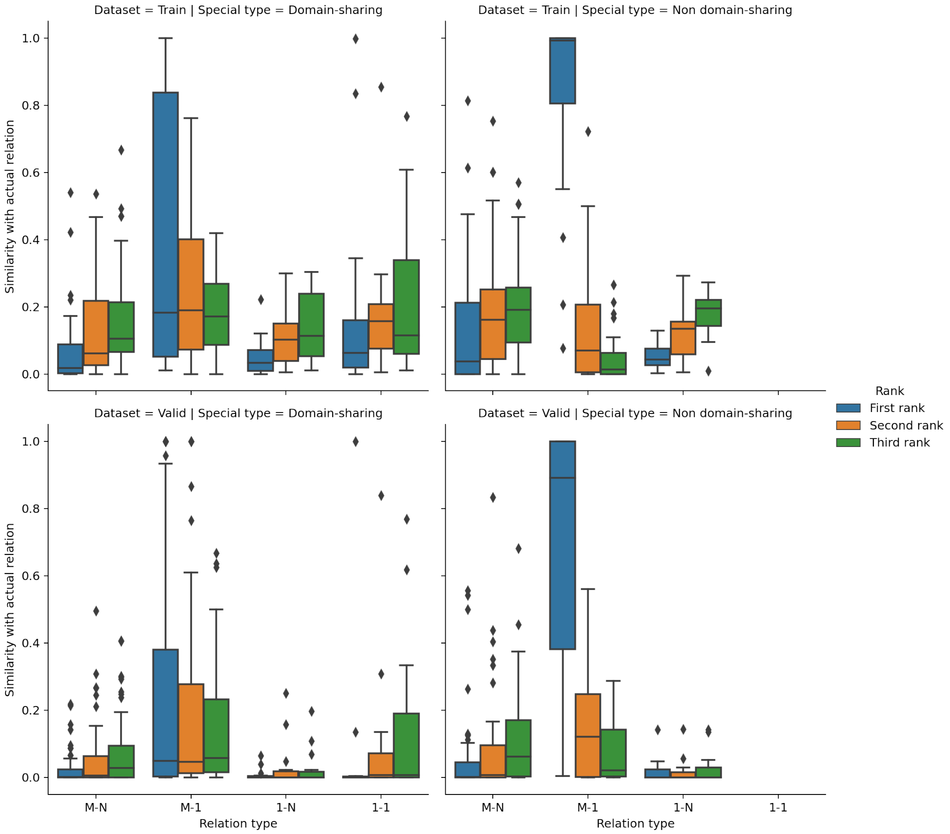

Figure 5.3: Distribution of average similarity between ranked relation and actual relation on validation and training data over relations for non domain-sharing and domain-sharing relations (standard entity model)

5.2 Improvements from selecting on overall MRR

As was mentioned in the Section 4.2, without training from scratch, we can obtain a better standard model for relation prediction if we select the best-performance model base on overall MRR instead of entity MRR with a small trade-off (i.e., the entity performance of model is dropped by around 2.0% MRR - Table 4.3).

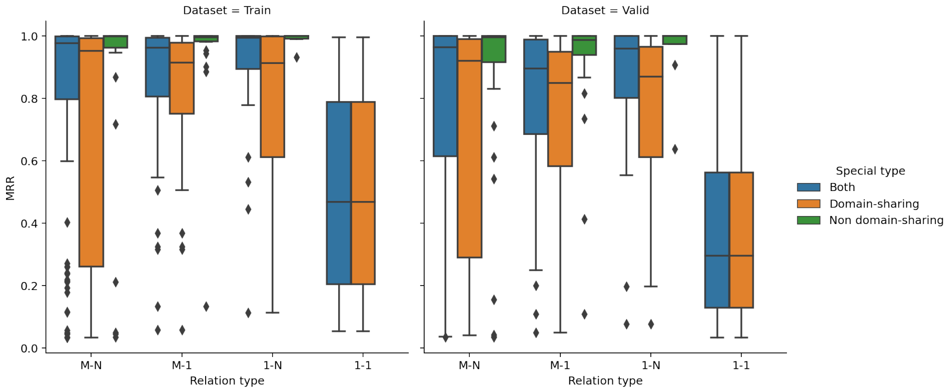

Figure 5.4: Distribution of filtered relation MRR on validation and training data over relations for for non domain-sharing and domain-sharing relations (standard overall model)

The improvement on relation prediction is illustrate nicely in Figure 5.4. As shown in that Figure, all median MRRs of standard overall model are much higher than all median MRRs of standard entity model, specially for M-N and 1-N relations on both dataset. However, the median MRRs of non domain-sharing relations still tend to be more higher than median MRRs of domain-sharing for all all relation types, and the MRR distribution of domain-sharing relations in all relation type are more dispersed compared to standard entity model. Those observations indicate that the standard overall model could achieve better MRR for most of non domain-sharing relations but only for some of domain-sharing relations. Figure 5.5 also nicely illustrates evidences to support this claims.

As shown in Figure 5.5, the first-rank relations for non domain-sharing relations are quite similar to the actual relation while the second and third rank relation are dissimilar to the true relation. Due to the characteristic of non domain-sharing relations, i.e, there does not exist another relation that has similar subject and object with this relation, the more similar to actual relation a relation is, the more

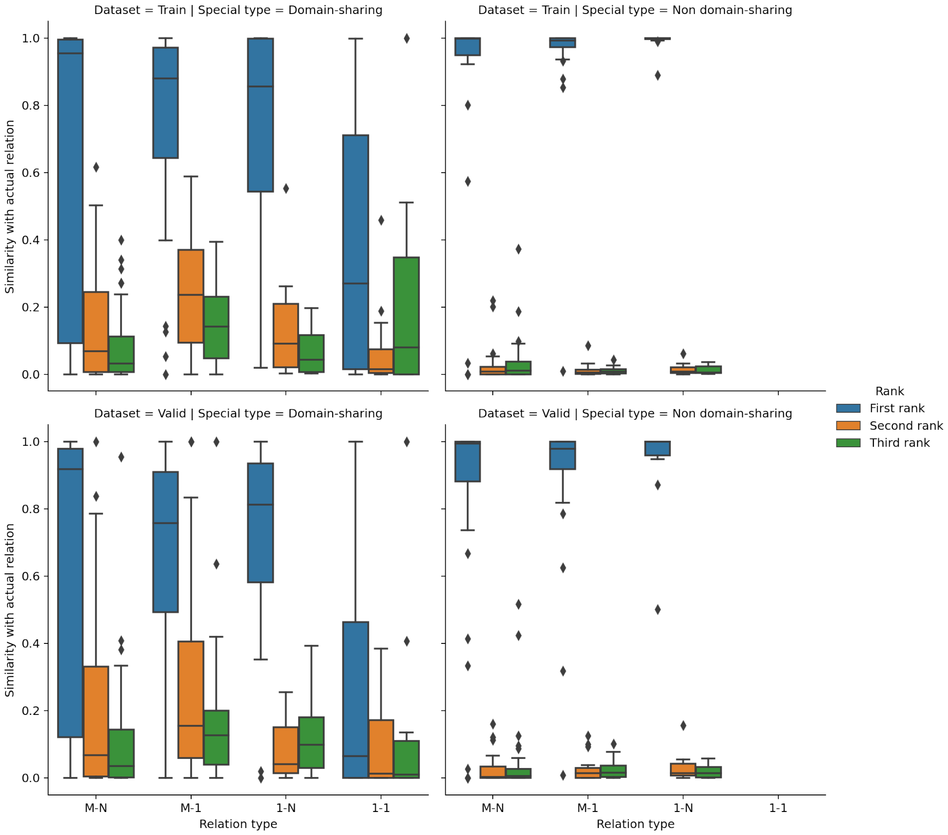

Figure 5.5: Distribution of average similarity between ranked relation and actual relation on validation and training data over relations for non domain-sharing and domain-sharing relations (standard overall model)

they are the same. Therefore, by selecting the best-performance model on overall MRR, we can find a model could perform better on predicting non domain-sharing relation than the baseline model.

However, Figure 5.5 also highlights the limitation of standard overall model. We can see that there is a wide dispersion on similarity for domain-sharing relations, especially for first rank. Entity overall model could rank the similar relation to actual relation higher the for some of domain-sharing relations. However, generally speaking, the model still struggles on predicting the domain-sharing relation. Table 5.3 illustrates some examples.

In sum, by selecting the best model on overall relation, we can find the model could achieve good MRRs for non domain-sharing relations without training from scratch or applying another training methods. However, it is not perfect, the model still struggles with some of domain-sharing relation, even though, we observed some improvement in predicting domain-sharing relation.

Subject	**Relation**	**Object**	**Rank**	**Type**
Demi Lovato	/celebrities/celebrity/celebrity_friends./celebrities/friendship/friend	Taylor Swift	14	M-N
	/education/educational_institution/campuses		1	1-1
	/education/educational_institution_campus/educational_institution		2	1-1
	/location/hud_county_place/place		3	1-1
My Name is Khan	/film/film/story_by	Karan Johar	3	M-1
	/film/film/written_by		1	M-1
	/film/film/produced_by		2	M-1
	/award/award_winning_work/awards_won./award/award_honor/award_winner		3	M-1
Europe	/base/locations/continents/countries_within	Poland	3	1-N
	/education/educational_institution/campuses		1	1-1
	/education/educational_institution_campus/educational_institution		2	1-1
	/location/hud_county_place/place		3	1-1
Austria	/base/aareas/schema/administrative_area/capital	Vienna	4	1-1
	/education/educational_institution_campus/educational_institution		1	1-1
	/education/educational_institution/campuses		2	1-1
	/location/hud_county_place/place		3	1-1

Table 5.3: Prediction examples from standard overall model. The first relations on the top of each box are the true relations while the remain ones are the predicted relation.

5.3 Hybrid training objective with overall MRR

As reported in Chapter 4, to overcome the trade-off from selecting best model on overall MRR, we could re-train model with hybrid standard training objective and select the best model based on overall MRR. The hybrid overall model could achieve entity MRR of 34.6% and relation MRR of 96.8%.

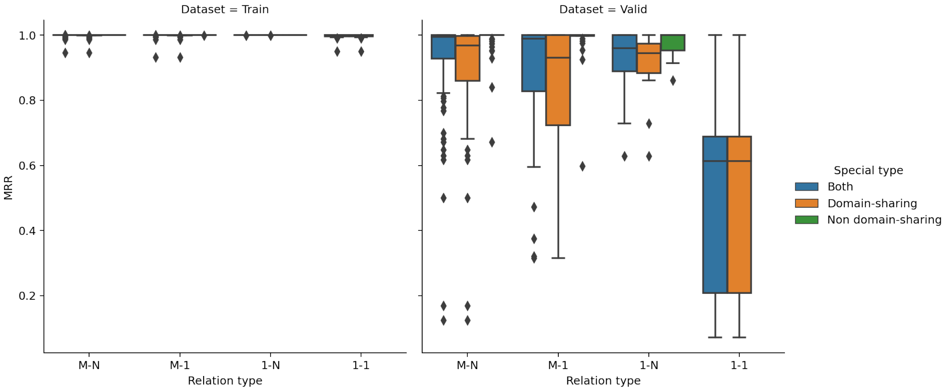

Figure 5.6: Distribution of filtered relation MRR on validation and training data over relations for for non domain-sharing and domain-sharing relations (hybrid overall model)

As illustrated in Figure 5.6, by training on hybrid training objective, undoubtedly, the best model could achieve significantly high MRRs on training dataset. Therefor, this in section, we mainly focus on the performance of hybrid overall model on validation dataset.

It can be seen in Figure 5.6, most distributions of MRRs are not as dispersed as they were shown by other models and the median MRRs of all relation type are significantly higher than median MRRs from other models. This indicates that the hybrid overall model achieve high MRR consistently for most relation types. Furthermore, from Figure 5.7, we can clearly see that the model could rank the relation having similar domain with actual relation higher on both non domain-sharing and domain-sharing relations. Besides, we can also observed that the median of second rank is more higher than the third rank in domain-sharing relation which indicates that model now have ability to capture the domain that shared between domain-sharing relations

However, surprisingly, even we observed that the model can achieved high MRRs for 1-1 relation on training dataset in Figure 5.6, the hybrid overall model still showed a wide dispersion for 1-1 relations and median of MRR for 1-1 relation is just over 60%. This could be a sign of overfitting in predicting 1-1 relations (Table 5.4 illustrates some examples).

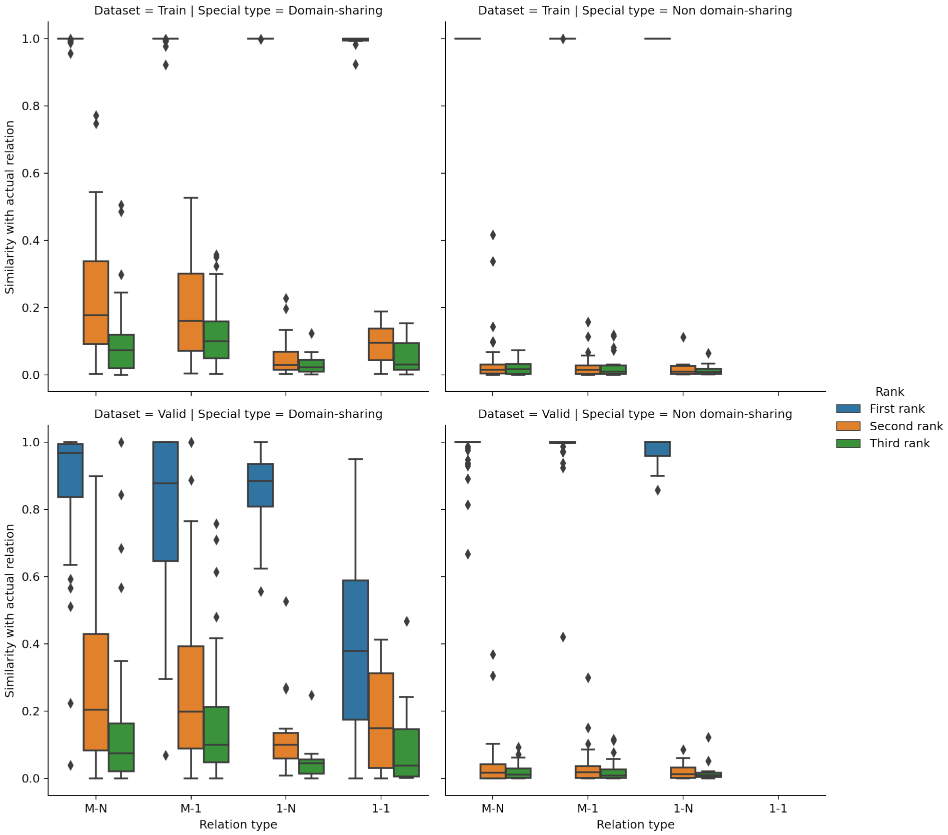

Figure 5.7: Distribution of average similarity between ranked relation and actual relation on validation and training data over relations for non domain-sharing and domain-sharing relations (hybrid overall model)

Subject	Relation	Object	Rank	Type
Singapore	/location/country/capital	Singapore	3	1-1
	/location/location/adjoin_s./location/adjoining_relationship/adjoins		1	M-N
	/military/military_combatant/military_conflicts./military/military_combatant_group/combatants		2	M-N
	/music/performance_role/regular_performances./music/group_membership/role		3	M-N
Syria	/sports/sports_team_location/teams		4	1-1
	/people/person/profession	Syria national	1	M-N
	/common/topic/webpage./common/webpage/category	football team	2	M-1
	/location/location/contains		3	M-N

Table 5.4: Prediction examples from hybrid overall model. The first relations on the top of each box are the true relations while the remain ones are the predicted relation.

Chapter 6

Conclusion

Knowledge graph completion is one of essential tasks since most available knowledge graphs are often missing many facts, and some of the edges they contain are incorrect (Angeli & Manning, 2013). There are two approaches to adding missing relation (Wang et al., 2017): entity prediction and relation prediction. However, the former have more attention than the latter (Chang et al., 2020). Therefore, the main objectives of this thesis is to study the impact of relation prediction in training objective and in evaluation protocol.

In this study, in order to incorporate the relation prediction into training objective, instead of only replacing the subject or objects in a triple to generate negative example (standard training objective), we also include the negative triples that generated by replacing the true relation with the relations in KG. We denoted that training as hybrid training objective.

Furthermore, to investigate the impact of relation on evaluation protocol, instead of selecting best-performance models on entity MRR, we investigate the effect of using overall MRR where the micro-average between MRRs of subject, object and relation predictions.

Within the limited of time, we can only study the impact of hybrid training on ComplEx mode, however, the finding led to a nice outcome, hybrid training does bring positive effect not only to entity prediction, but also to relation prediction.

Furthermore, by selecting models on overall MRR, we can find the model that can work well with both relation prediction and entity ranking. However, the performance of entity ranking is slightly decreased. To overcome that, the combination between overall MRR and hybrid training can brings huge impact, it's not only improve the relation prediction performance but also overcome the trade-off of overall MRR.

6.1 Future Work

Our work clearly has some limitations, we have not consider the HPC with including the best configuration form Chen et al. (2021)'s HPC space. We note that this HPC space could potentially yield better models; however, we have not fully explored that HPC space. Therefore, future work should concentrate on that HPC space.

Secondly, the thesis is only focus on ComplEx model and on only two dataset FB15K-237 and WN18RR, therefore it would be more interesting if we can explore other KGE models such as DistMult (Yang et al., 2014) or RotatE (Sun et al., 2019) due to their similarity to the ComplEx model.

Thirdly, in the Chapter 5, we have identify the limitation and the improvement that hybrid training objective. However, we still can not fully explain the reason why the hybrid model could bring nice benefit to not only relation prediction but also entity prediction.

Finally, we observed that reciprocal relation could bring a nice improvement for model's performance on entity ranking. However, when coming to relation ranking with reciprocal relation, an open question would be that which relation we should consider when evaluation an triple $(i, ?, j)$. One potential solution would be combine reciprocal relation and relation together to have one single relation when performing relation prediction.

6.2 Summary

In sum, even with the limitations, this study is able to show importance of relation prediction not only in training objective but also in model selection. By including the negative examples generated by replacing the true relation with another relation to the standard 1vsAll training objective, the model could achieve the MRR of 96.8% in relation prediction on validation dataset. This MRR is significantly higher than the MRR which the model trained on standard 1vsAll can achieve (MRR of 21.8%).

Furthermore, model selection could become handy to find the model trained on standard objective that can work well on both relation and entity prediction tasks without retraining from scratch.

Bibliography

Luis A Nunes Amaral. A truer measure of our ignorance. *Proceedings of the National Academy of Sciences*, 105(19):6795–6796, 2008.

Gabor Angeli and Christopher D Manning. Philosophers are mortal: Inferring the truth of unseen facts. In *Proceedings of the seventeenth conference on computational natural language learning*, pp. 133–142, 2013.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. TuckER: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5185–5194, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1522. URL https://aclanthology.org/D19-1522.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1533–1544, 2013.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013a. URL https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013b.

Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embeddings, 2014a.

Antoine Bordes, Jason Weston, and Nicolas Usunier. Open question answering with weakly supervised embedding models, 2014b.

Samuel Broscheit, Daniel Ruffinelli, Adrian Kochsiek, Patrick Betz, and Rainer Gemulla. LibKGE - A knowledge graph embedding library for reproducible research. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 165–174, 2020. URL https://www.aclweb.org/anthology/2020.emnlp-demos.22.

David Chang, Ivana Balazevic, Carl Allen, Daniel Chawla, Cynthia Brandt, and Richard Andrew Taylor. Benchmark and best practices for biomedical knowledge graph embeddings, 2020.

Yihong Chen, Pasquale Minervini, Sebastian Riedel, and Pontus Stenetorp. Relation prediction as an auxiliary training objective for improving multi-relational graph representations. In *3rd Conference on Automated Knowledge Base Construction*, 2021. URL https://openreview.net/forum?id=Qa3uS3H7-Le.

Zijun Cui, Pavan Kapanipathi, Kartik Talamadupula, Tian Gao, and Qiang Ji. Type-augmented relation prediction in knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7151–7159, 2021.

Bhavana Dalvi, Niket Tandon, and Peter Clark. Domain-targeted, high precision knowledge extraction. *Transactions of the Association for Computational Linguistics*, 5:233–246, 2017.

Danica Damljanovic and Kalina Bontcheva. Named entity disambiguation using linked data. 2012.

Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pp. 1811–1818, February 2018. URL https://arxiv.org/abs/1707.01476.

Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. Shared embedding based neural networks for knowledge graph completion. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 247–256, 2018.

Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1709–1719, 2019.

Larry Heck, Dilek Hakkani-Tür, and Gokhan Tur. Leveraging knowledge graphs for web-scale unsupervised semantic parsing. 2013.

Seyed Mehran Kazemi and David Poole. Simple embedding for link prediction in knowledge graphs. *Advances in neural information processing systems*, 31, 2018.

Stanley Kok and Pedro M. Domingos. Statistical predicate invention. In *Proceedings of the 24th International Conference on Machine Learning*, volume 227 of *ACM International Conference Proceeding Series*, pp. 433–440. ACM, 2007. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273551.

Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. Canonical tensor decomposition for knowledge base completion. In *International Conference on Machine Learning*, pp. 2863–2872. PMLR, 2018.

Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379*, 2015a.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015b.

Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6):1150–1170, 2011.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Icml*, 2011.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, Jan 2016a. ISSN 1558-2256. doi: 10.1109/jproc.2015.2483592. URL http://dx.doi.org/10.1109/JPROC.2015.2483592.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016b.

Daniel Ruffinelli, Samuel Broscheit, and Rainer Gemulla. You can teach an old dog new tricks! on training knowledge graph embeddings. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BkxSmlBFvr.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. End-to-end structure-aware convolutional networks for knowledge base completion, 2018.

Baoxu Shi and Tim Weninger. Proje: Embedding projection for knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Michael PH Stumpf, Thomas Thorne, Eric De Silva, Ronald Stewart, Hyeong Jun An, Michael Lappe, and Carsten Wiuf. Estimating the size of the human interactome. *Proceedings of the National Academy of Sciences*, 105(19):6959–6964, 2008.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space, 2019.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1499–1509, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1174. URL https://aclanthology.org/D15-1174.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International conference on machine learning*, pp. 2071–2080. PMLR, 2016.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), Mar. 2016a. URL https://ojs.aaai.org/index.php/AAAI/article/view/10329.

Ruobing Xie, Zhiyuan Liu, Maosong Sun, et al. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*, volume 2016, pp. 2965–2971, 2016b.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

Liang Yao, Chengsheng Mao, and Yuan Luo. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*, 2019.

Haiyuan Yu, Pascal Braun, Muhammed A Yıldırım, Irma Lemmens, Kavitha Venkatesan, Julie Sahalie, Tomoko Hirozane-Kishikawa, Fana Gebreab, Na Li, Nicolas Simonis, et al. High-quality binary protein interaction map of the yeast interactome network. *Science*, 322(5898):104–110, 2008.

Zhicheng Zheng, Xiance Si, Fangtao Li, Edward Y. Chang, and Xiaoyan Zhu. Entity disambiguation with freebase. In *The 2012 IEEE/WIC/ACM International Conference on Web Intelligence (WI'2012)*, 2012.

Appendix A

Program Code / Resources

The source code, are available at my personal repository in branch ”second_milesstone”
https://github.com/Kenkoko/kge/tree/second_milesstone
They as well as a PDF version of this thesis is available on Github:
https://github.com/Kenkoko/master_thesis_2022.
The repo stores all of configuration files to reproduce the master thesis results:
https://github.com/Kenkoko/config_files

A.1 New parameters

Parameter	**Potential Values**	**Default value**	**Purpose**
entity_ranking			
query_types			
sp_to_o	{True, False}	True	Report object prediction
po_to_s	{True, False}	True	Report subject prediction
so_to_p	{True, False}	False	Report relation prediction
1vsAll			
query_types			
sp_	{True, False}	True	Train on object prediction
po_	{True, False}	True	Train on subject prediction
so_	{True, False}	False	Train on relation prediction
query_weight			
sp_	Close interval, value(s)	1	Set weight for object pred
po_	Close interval, value(s)	1	Set weight for subject pred
so_	Close interval, value(s)	1	Set weight for relation pred
eval			
metrics_per			
relation_prediction	{True, False}	False	Report relation prediction

Table A.1: New parameters has been implemented for this study

Appendix B

Details of experimental setting

Hyperparameter	**Unrestricted search**	**Semi-restricted search**
Embedding size	{128, 256, 512}	{256, 512, 1024, 2048}
Training type	1vsAll	1vsAll
Reciprocal	{True, False}	False
Loss	CE	CE
Optimizer	{Adam, Adagrad}	Adagrad
Batch size	{128, 256, 512, 1024}	{128, 256, 512, 1024}
Learning rate	[3.0e-3, 1.0], log scale	[3.0e-3, 1.0]
LR scheduler patience	[0, 10]	[0, 10]
L_p regularization	{L1, L2, L3, None}	{L1, L2, L3, None}
Entity emb. weight	[10-20, 10-5]	[10e-20, 1.0]
Relation emb. weight	[10-20, 10-5]	[10e-20, 1.0]
Frequency weighting	{True, False}	True
Dropout		
Entity embedding	[-0.5, 0.5]	[-0.5, 0.5]
Relation embedding	[-0.5, 0.5]	[-0.5, 0.5]
Embedding initialization	{Normal, Unif, XvNorm, XvUnif}	Normal
Std. deviation (Normal)	[1.0e-5, 1:0]	0.001
Interval (Unif)	[-1.0, -1.0e-4]	
Gain (XvNorm)	1.0	
Gain (XvUnif)	1.0	
Relation prediction weight	[0.005, 4.0]	[0.1, 8.0]

Table B.1: Hyperparameter search space used in our study.

Appendix C

Further Experimental Results

				Entity prediction				Relation prediction			
Selected on	**Dataset**	**Entity**	**Relation**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
Entity MRR	FB15K-237	Yes	No	35.2	26.4	38.5	52.7	21.8	09.8	17.3	58.1
		Yes	Yes	35.0	26.0	38.4	52.9	95.4	93.6	96.9	98.3
		No	Yes	23.7	16.7	25.6	37.9	97.2	95.6	98.9	99.6
	WN18RR	Yes	No	46.9	43.8	48.0	52.7	76.0	66.1	80.5	99.9
		Yes	Yes	46.6	43.3	48.4	52.6	67.2	54.5	75.4	85.9
		No	Yes	43.6	41.2	44.5	48.0	63.1	49.5	71.5	82.3
Overall MRR	FB15K-237	Yes	No	33.5	24.8	36.5	51.4	89.4	85.7	91.7	96.7
		Yes	Yes	34.6	25.8	37.6	52.3	96.8	95.3	98.2	99.2
		No	Yes	24.0	17.1	25.8	38.0	97.4	95.8	98.8	99.5
	WN18RR	Yes	No	45.7	43.2	46.4	50.4	76.0	66.1	80.5	99.9
		Yes	Yes	45.0	42.0	46.1	50.9	67.2	54.5	75.4	85.9
		No	Yes	43.4	40.7	44.6	48.4	63.1	49.5	71.5	82.3
Relation MRR	FB15K-237	Yes	No	30.9	22.8	33.7	47.2	92.9	90.5	94.7	97.5
		Yes	Yes	32.0	23.6	34.9	48.9	97.3	95.8	98.8	99.5
		No	Yes	17.9	12.1	18.6	29.6	97.8	96.3	99.2	99.8
	WN18RR	Yes	No	45.6	43.2	46.4	50.4	89.4	82.3	96.3	1.000
		Yes	Yes	40.0	36.7	41.5	45.8	89.3	82.9	95.1	1.000
		No	Yes	02.1	01.0	02.0	03.9	90.8	84.4	97.0	1.000

Table C.1: Performance of models on unrestricted HPC space.

				Entity prediction				Relation prediction			
Selected on	**Dataset**	**Entity**	**Relation**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
Entity MRR	FB15K-237	Yes	No	36.6	27.4	40.0	55.1	94.7	92.1	96.8	98.9
		Yes	Yes	37.3	28.2	41.0	55.5	97.9	96.6	99.3	99.7
		No	Yes	25.8	18.6	28.0	40.3	96.2	94.4	97.9	98.9
	WN18RR	Yes	No	47.9	44.1	49.2	55.6	83.7	77.3	87.0	98.7
		Yes	Yes	48.1	44.3	49.4	55.4	86.0	80.3	89.2	99.0
		No	Yes	45.8	42.3	47.3	52.6	80.9	74.9	83.1	97.1
Overall MRR	FB15K-237	Yes	No	36.5	27.3	40.0	55.0	94.9	92.4	96.9	99.0
		Yes	Yes	37.1	27.9	41.0	55.5	98.0	96.7	99.3	99.7
		No	Yes	25.8	18.5	28.0	40.4	96.4	94.7	98.0	99.1
	WN18RR	Yes	No	48.0	44.2	48.9	55.7	83.9	77.2	88.2	99.1
		Yes	Yes	48.1	44.1	49.4	55.8	86.1	80.9	88.4	98.8
		No	Yes	45.2	41.4	46.9	52.1	87.8	83.0	90.6	97.6
Relation MRR	FB15K-237	Yes	No	35.5	26.2	38.8	54.4	95.0	92.5	97.2	99.0
		Yes	Yes	36.5	27.3	40.0	54.6	98.0	96.7	99.3	99.7
		No	Yes	17.3	10.9	18.2	30.2	97.8	96.3	99.2	99.8
	WN18RR	Yes	No	48.1	44.1	49.4	56.2	84.8	78.4	88.5	99.5
		Yes	Yes	36.3	34.9	36.8	38.7	90.2	83.9	95.8	1.000
		No	Yes	40.0	35.8	42.2	47.6	89.2	84.8	91.5	98.6

Table C.2: Performance of models on semi-restricted HPC space.

Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Master-/Bachelorarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Er- klärung rechtliche Folgen haben wird.

Mannheim, den 31.08.2014 Unterschrift