修　　士　　論　　文

Zero-shot Learning with Shared Embedding Spaces

(共有埋め込み空間を利用したゼロショット学習)

指導教員　　鶴岡 慶雅 教授

東京大学大学院情報理工学系研究科
電子情報学専攻

氏 名　　48-186469 鄧　一凡

提 出 日　　2020 年 8 月 13 日

Abstract

Recently, with the development of deep learning technology, deep learning has achieved great progress, and deep neural network models have even outperformed humans in many tasks, especially in the field of natural language processing (NLP). However, the performance of deep models is still far from human performance in zero-shot learning (ZSL), where the models are asked to deal with targets belonging to unseen classes. ZSL is becoming increasingly important, as it can resolve the lack of labeled data which is a common problem in real-world applications. One approach to ZSL is to acquire shared embedding spaces of the seen and unseen classes. With such shared embedding spaces, the deep models can transfer the knowledge from seen classes to unseen classes, allowing the models to perform the ZSL tasks. This thesis takes bilingual lexicon induction (BLI) and zero-shot relation extraction (RE) in knowledge base question answering (KBQA) as two examples of research subjects in ZSL and investigates various methods utilizing the shared embedding spaces. BLI, generating word translations, is a basic application of bilingual word embedding (BWE) mapping. Research on the BLI can help to improve the understanding of embedding mapping which can be applied to zero-shot RE in KBQA. KBQA is a special type of question answering (QA) tasks, where the questions are based on knowledge bases. KBQA is closely related to real-world applications, as it provides a feasible and practical way to deal with vast real-world knowledge. RE is a key component of KBQA, identifying the relation in the question in order to understand the meaning of the question. Zero-shot RE is the RE where the test relations do not appear in the training data. The research in this thesis pushes the understanding of embedding spaces in zero-shot learning forward and improves the performance of BLI and zero-shot RE.

Acknowledgements

I could not enjoy the meaningful and joyful years in doing my master research without the help of many kind seniors and my lovely friends.

First and foremost, I want to thank my advisor, Professor Yoshimasa Tsuruoka. He has encouraged me when I apply for the master's project in Tsuruoka Laboratory, which bred the very beginning of my master's career. He is a successful teacher, providing the most interesting and challenging orientation tutorials. He is also an insightful scholar that he can always point out the covert but fatal problems in my work. He has been caring and supportive about my academic activities and my life and has provided many valuable chances such as a part-time job at the National Institute of Advanced Industrial Science and Technology (AIST). He has been giving suggestions but never giving compulsive orders, which I think has been the reason for the friendly and vigorous atmosphere of our laboratory. I really feel lucky that he is my advisor.

I would like to thank our laboratory members. When I came to the laboratory, I could hardly understand Japanese, and I had little practical knowledge of machine learning. My Chinese fellow, Li Tong, has been my "Japanese interpreter" and "technical encyclopedia" in the first semester. Ri Ryoukan, my deskmate, has always been caring about the research of laboratory members including me. We have discussed over many fields, and some discussions were very inspiring. Matiss Ritkers, a researcher, has been warm-hearted and always ready for help. And other members have been sharing their ideas, comments, suggestions, both in academics and in life experience, which are invaluable to me.

I want to thank AIST members, especially my direct leader, Hiroshi Noji. He is firm and persevering, leading me to a series of interesting topics, including the KBQA. I have encountered failure in some difficult topics, but he has never been frustrated and has been always encouraging me to look forward. Other AIST members are also kind and helpful. I really enjoy the open and friendly culture of AIST.

Besides, I want to thank Lyu Zhaoyang, my former roommate and best friend. We worked together on the BWE mapping, and we have discussed many latest topics in the machine learning field some of which are very inspiring. I also would like to thank other friends who concern my research and gives useful advice .

Thanks to all the members I have mentioned or not mentioned. Your help has made my experience as a master student fulfilling.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Backgound

The development of deep neural network models, especially in the field of natural language processing (NLP), is pushing the performance on many tasks forward in recent years. Some deep models have even outperformed humans in several tasks. However, the behavior of deep models is still far from human in some tasks, including zero-shot learning (ZSL).

ZSL is a setting for classification tasks when there are no training examples for certain classes, and those classes are called zero-shot classes. ZSL is a difficult task even from the view of humans. However, ZSL would be very useful in real-world applications. Because in real-world applications, the labels are usually insufficient for large data because of the difficulty and the labor-consuming of labeling, and the classification system might contain dynamic classes because of the ever-changing essence of the real-world knowledge. An example of ZSL is the animal recognition task where the models should tell the animal species given the pictures of the animals. However, in the dataset for this task, the labeled data only cover a few species of animals. Here, to recognize the animals whose species are not labeled in advance is a typical ZSL task.

In fact, the models are not ignorant of the zero-shot classes. There might be some explicit information about the zero-shot classes such as textual explanations. There might also be inexplicit information about the zero-shot classes, among which the shared embedding space is a popular information source.

The term, *embeddings*, denotes certain vector forms for representating some input items. There are word embeddings for words, relation embeddings for knowledge base relations, and so on. The embeddings are learned from data, usually pre-trained, forming an embedding space where similar items are represented as points in close proximity. The pre-training of the embedding spaces does not require labeled data in general; therefore, it is possible to get the embeddings in the embedding space for the items that belong to the unseen classes (in the case that the unseen classes come from the lack of labeling). With the shared embedding space of seen items and unseen items, much inexplicit information about the unseen items can be found, and help the classification of the unseen classes.

Bilingual word embedding (BWE) mapping is a research topic about building the relationship

between the word embedding spaces from two languages. Usually, the mapping is performed between a rich-resource language (e.g. English) and a low-resource language (e.g. Spanish). The mapping helps transfer the knowledge from the rich-resource language (source language) to the low-resource language (target language). The embedding mapping also constructs a uniform relationship for both seen and unseen classes. The relationship might be between words from one language to words from the other language, just like it is in the BWE mapping setting. The other end of the relationship, instead of words from the other languages, might also be items of other types, such as relations in a knowledge base. The uniform relationship can help to transfer the knowledge from the seen classes to the unseen classes. For example, in the bilingual lexicon inference (BLI) task, the translation words of unseen words can be inferred from the BWE mapping and the translation words of seen words. Therefore, embedding mapping is also utilized as a method for ZSL. The technique of embedding mapping in ZSL is called *adapter*.

The concept of adapters is first proposed by Wu et al. (2019) [1]. It is proposed to solve the zero-shot relation extraction (RE) problem in the knowledge base question answering (KBQA). KBQA is a special kind of question answering (QA) tasks, where the questions are based on some knowledge bases. The RE is a necessary component of KBQA. The Zero-shot RE problem is that there are unseen relations in the test questions. Since there are usually a large number of relations in a normal knowledge base, the existence of many unseen relations is a common phenomenon. Because of the existence of knowledge bases, a shared relation embedding space can be pre-trained from the knowledge base structures (or from distant supervision) for both seen and unseen relations. Therefore, the adapter is ideal to be applied to this problem.

1.2 Research Problem

In this thesis, we want to improve ZSL problems with shared embedding spaces. We take two ZSL tasks as the research subjects, the BLI and the RE in KBQA. In both of the tasks, the main approach is by embedding mapping which forms a uniform relationship for both seen and unseen classes.

The embedding mapping is first studied and improved in the BWE mapping setting for simplicity and explainability. After we get more acquainted about the BWE mapping, the knowledge is applied for the embedding mapping in zero-shot RE, and to improve zero-shot RE performance.

1.3 Thesis Outline

This thesis is organized as follows.

In Chapter 2, the background knowledge of this thesis, including the ZSL, the embeddings, the BWE mapping, and the KBQA, are introduced.

In Chapter 3, several important related studies that are the basis of this thesis are introduced. There are two models for BWE mapping and three models for KBQA and RE in KBQA.

In Chapter 4, we investigate the BWE mapping and improve it on the BLI.

In Chapter 5, we investigate zero-shot RE, and propose methods for improving zero-shot RE.

Chapter 6 is the conclusion of the main contributions in this thesis and states possible future work.

Chapter 2

Background

2.1 ZSL

As the recent developments of powerful deep learning techniques and deep models, supervised classification tasks have achieved significant success. In the field of NLP, recent deep models such as Transformer [2] have been prevailing in applications. The deep models beat recurrent neural networks (RNNs), including gated recurrent unit (GRU), long short-term memory (LSTM), bi-directional LSTM (BiLSTM), and etc., in many tasks, and even outperform humans in some tasks. The good performance of deep models is based on large amounts of supervision data. However, there are some tasks when the supervising data or labeled data are not sufficient or even do not exist, and deep learning would not behave well without few-shot learning or ZSL.

Few-shot learning and ZSL are methods dealing with the tasks where there are only a few or no training cases in the test classes. In the real world, it is common that the labeled data are not sufficient so that there are possibly only a few cases or no cases for a certain class. In some situations, the classes themselves are dynamic, updated from time to time, so that it is not reasonable to require labeled data for new classes. For example, knowledge bases such as Wikidata [3] are continuously updating the database, trying to align the stored knowledge to the ever-changing real-world knowledge. Few-shot learning and ZSL are becoming increasingly important as supervised learning has become increasingly powerful and more and more real-world applications are being considered.

This thesis focuses on ZSL where there are no training cases for some test classes. The classes for which there exist training cases are called seen classes, those for which there are no training cases are called unseen classes. The key in ZSL is to transfer the knowledge learned from the seen classes to the unseen classes. Therefore, ZSL is a variant of transfer learning [4].

2.2 Embeddings

2.2.1 Word Embeddings

Embedding is a term denoting the learned vector representation of the items in machine learning. It is first applied to replace the one-hot representation of words in NLP.

The one-hot representation is also a vector where there is only one dimension with the value of one while the values of other dimensions are all zeros, and different dimensions are set to be one for different words. There are many defects with the one-hot representation. First, there is no notion of similarities between different words. Any word is as far as any other words in the one-hot vector space. Second, the vector size is the same as the vocabulary size. The vocabulary size is usually large in normal NLP tasks, ranging from several tens of thousands to hundreds of thousands. It is not reasonable for a recent computing device to do complex and rapid calculations in a deep model with such long vectors of tens of thousands dimensions. Third, it is difficult to add a new word to the vocabulary. If we add a new dimension to the one-hot vectors and assign the new dimension to the new word, the whole model structure and previously optimized parameters should be modified. If we preserve the previous one-hot vector space, the new vector representation of the new word should be placed in a reasonable point so that the new vector is closer to the vectors of words which are closer (semantically or syntactically) to the new word.

With the developments of deep models, researchers propose to use the intermediate states of the deep models to represent words [5], which is found to be effective. Those intermediate states, which are later called word embeddings, have overcome all the defects of the one-hot word representations. They are compressed in a low dimension space where similar words are close in that space (in the case that the model is optimized toward a reasonable objective). If there is a new word to the vocabulary, its embedding can be randomly initialized in the embedding space, without the need to change the model structure or parameters.

In 2013, Mikolov et al. [6] design word embedding pre-training algorithms, such that pre-trained word embeddings can be applied in various algorithms. Based on the distributional hypothesis in linguistics that *"linguistic items with similar distributions have similar meanings"* (Harris, 1954), Mikolov et al. propose two embedding algorithms: CBOW algorithm and continuous skip-gram algorithm. The tasks in those algorithms are predicting the target word by its context or predicting the context by a word. The simple idea shows its power in the syntactical and semantical encoding ability. Two pieces of evidence are found to support the encoding ability. First, in pre-trained word embedding space trained with either CBOW or skip-gram, similar words share similar embeddings. For example embeddings of words *France* and *Italy* are close to each other. Second, the way that two words are similar is similar to the way of other similar word pairs. For example, $Embedding(\text{“}big\text{''}) - Embedding(\text{“}biggest\text{''}) \approx Embedding(\text{“}small\text{''}) - Embedding(\text{“}smallest\text{''})$. The high quality of the embeddings results in the wide applications of the pre-trained word embeddings.

After CBOW and skip-gram, various embedding algorithms have been proposed. The most popular embedding algorithms include GloVe [7] and fastText [8]. In fastText, subword information is taken into consideration for training the word embeddings.

Recently, contextualized word embeddings, such as ELMo [9] and BERT [10], have been proposed and show great power when applied in downstream tasks. In contextualized word embeddings, the embedding of a word is not fixed but depends on the context that word lies in. The contextualized

word embeddings alleviate the polysemy embedding problem. For example, the embedding of the word *"bank"* can be closer to "landscape" when the context contains words like "river" or "stone"; and it can be closer to "finance" when the context contains words like "money" or "ATM". Besides, deep contextualized word embeddings trained from large amounts of corpora can capture more semantical information and real-world knowledge.

2.2.2 BERT

As introduced in 2.2.1, BERT, Bidirectional Encoder Representations from Transformers, is pre-trained contextualized word embeddings. BERT is also well-known for its superiority as a general pre-trained model. It has outperformed many state-of-the-art models at the time it was proposed. BERT is an active research field in NLP and also in some cross fields. There are studies to improve the model architectures and pre-training tasks, such as the work of RoBERTa [11], ALBERT [12], XLNet [13] ,and T5 [14]. There are studies to distill the BERT models, such as the work of TinyBERT [15] and DistilBERT [16]. And there are studies to transfer BERT into other domains, such as Multilingual BERT for multilingual studies, SciBERT [17] for scientific texts, KnowBERT [18] for knowledge bases, and so on. BERT is also an important component of the research in this thesis. Therefore, here comes the introduction of BERT.

The body of BERT, as its name implies, is the transformer model [2]. The transformer is constructed based on the attention mechanism instead of recurrent neural networks (RNNs). The attention mechanism enables the model to deal with long sequences deep and fast. The transformer applies self-attention and represents the input position by adding positional embeddings to the word embeddings. Besides positional embeddings, BERT also adds segment embeddings in order to split the input sequence into more than one part. The transformer also applies multi-head attention to make the model deeper. There are two versions of BERT which apply a smaller and a larger transformer models. The one which applies the smaller transformer model is called $\text{BERT}_{\text{BASE}}$; the model is a 12-layer transformer with 768-dimension hidden states and 12 attention heads, with 110M parameters in total. The one which applies the larger transformer is called $\text{BERT}_{\text{LARGE}}$; the model is a 24-layer transformer with 1024-dimension hidden states and 16 attention heads, with 340M parameters in total. One key to the success of BERT is the large and deep transformer structure.

There are two pre-training tasks for BERT. The first task is the masked language model (MLM). In MLM, some tokens are masked out (replaced with a [MASK] token, or with a random token) from the input sequence, and the task is to predict the masked tokens. The second task is the next sentence prediction (NSP). In NSP, the input is two sentences, and the task is to predict whether the two sentences are a consequent sentence pair form a corpus. The two tasks do not require any human labels (the masking process is automatically performed), thus can be trained from large corpora. BERT is trained from BooksCorpus [19] and English Wikipedia, about 3,300M words in total. The large pre-training data is another key to the success of BERT.

There are two ways to use BERT in downstream tasks. The first way is to take BERT as contextualized word embeddings. The embeddings are usually got from the hidden states of the last layer, but the hidden states from all other layers or their concatenations can also be used as the word embeddings. The second way is to take BERT as a pre-trained model and fine-tune the BERT with the downstream tasks.

2.2.3 Relation Embeddings

In knowledge bases, there are many relations as the connection between different nodes (representing entities). For example, there is a relation *P69: "educated at"* in Wikidata [3], indicating an entity (e.g. *Q42: Douglas Adams*) been educated at another entity (e.g. *Q691283: St. John's College*). *P69* is the index of the relation and *"educated at"* is the name as well as a short description of the relation. When using the knowledge base as part of the input to neural networks, it is also preferable to encode the relations (as well as the entities) as vectors. The vector representations for relations are called relation embeddings.

Instead of the words in the relation name, the rich structural information of the knowledge base is more commonly applied to train the relation embeddings. The knowledge base can be seen as a graph, where the vertices denote the entities and the edges denote the relations. The knowledge in the knowledge base is represented by (head entity, relation, tail entity) triplets. For example, the knowledge that *"Douglas Adams is educated at St. John's College"* is represented as (Q42, P69, Q691283) in Wikidata. Therefore, the structural information is rich in knowledge base.

A popular way to apply this structural information is by translate models. The idea of translate models is training the embeddings of relations and entities in the form of a translation setting: predicting the tail entity by the (head entity, relation) tuple. The first proposed translate model is TransE [20]. The training target is to minimize the distance between the sum of the head entity embedding and relation embedding with the tail entity embedding: $d(\mathbf{h} + \mathbf{l}, \mathbf{t})$ where d denotes the distance metric, and $\mathbf{h}$, $\mathbf{l}$, $\mathbf{t}$ denotes the embeddings of head entity, relation, and tail embedding, respectively. Besides TransE, there are many other translate models such as TransH [21], TransR [22], and TransD [23].

Han et al. (2018) [24] have proposed a JointNRE model to jointly train word embeddings together with relation embeddings. In order to train word embeddings with relation embeddings, Han et al. utilize distant supervision text corpus for relations. The training target is to maximize the possibility of relation - distant supervision text pair. As a result, the word embeddings and relation embeddings can be trained into the same embedding space and thus can be easier to be utilized by downstream task models.

2.3 BWE Mapping

2.3.1 Background

After the development of pre-trained word embeddings, researchers begin to discover the similarity between word embedding spaces across different languages. As introduced above, the structure of the pre-trained word embedding space reflects the syntax and semantics of the language. As a result of the similarities of syntax and semantics among languages, there exist certain similarities between word embedding spaces of different languages. A certain relationship can be built among languages with the similarities, and help knowledge transfer from languages to languages. A common way to utilize the structural similarity among languages is to map the word embedding space of one language to that of another language (usually by a mapping matrix), as shown in Fig.2.1. This method is called word embedding mapping. The mapping between two languages, BWE mapping, is the basic setting, and is the focus of this thesis.

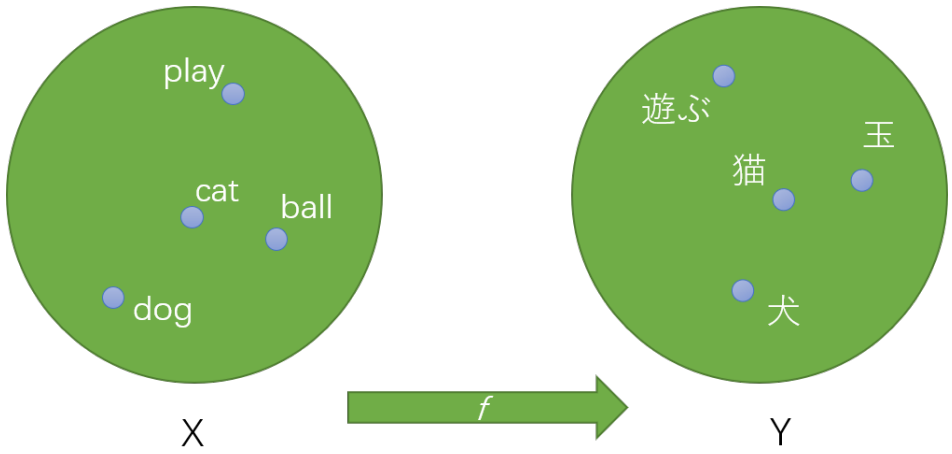


Figure 2.1: An overview of BWE mapping (taking English (en) - Japanese (jp) language pair as an example). A mapping function f (usually a linear transform with by a matrix) maps the source embedding space $\mathbf{X}$ of English to the target embedding space $\mathbf{X}$ of Japanese, such that $f(\mathbf{X}) \approx \mathbf{Y}$ (the embeddings of known translation word pairs are in respective rows of the two matrices).

A basic application of BWE is zero-shot word translation [25], aka bilingual lexicon induction (BLI). In BLI, the training data are usually known translation word pairs, and in test time, it is required to get the translations of words that do not appear in the training data. BLI is usually applied as the evaluation task to test mapping performance. Besides BLI, BWE mapping can also be applied to model transfer (transfer a trained model from the source language to the target language), and sentence translation [26].

2.3.2 Mapping Paradigm

As proposed by Mikolov et al. (2013) [25], the mapping is usually performed by multiplying a mapping matrix W to the source embedding $\boldsymbol{x}_i$ to map to the target embeddings $\boldsymbol{y}_i$. When there is a supervising lexicon (supervised setting), the training target can be minimizing the distance between the mapped embedding $W\boldsymbol{x}_i$ and the target embedding $\boldsymbol{y}_i$,

$$\min_W \sum_{i=1}^{n} d(W\boldsymbol{x}_i, \boldsymbol{y}_i) \tag{2.1}$$

where $d(,)$ denotes the distance metric between two vectors, n denotes the size of the supervising lexicon. The distance metric can be Euclidean distance [25], cosine distance [27], or others such as Relaxed Cross-Domain Similarity Local Scaling (RCSLS) [28]. When there is no supervising lexicon, it is called unsupervised BWE mapping. Adversarial learning of the mapping matrix [25, 29, 30] is a common way to deal with this setting.

Besides applying a matrix to do the mapping, researchers have tried other mapping functions such as multi-layer perceptron (MLP) or neural networks. Nakashole et al. (2018) [31] and Moshtaghi et al. (2019) [32] have proposed non-linear but locally linear mapping functions and have shown improvements in distant language pairs (such as English (en) - Chinese (zh) pair). Besides the two well-designed locally linear models, the more complex the mapping function is, the mapping performance is usually worse. That is because the complex function contains a large parameter space, which is not good for generalization.

Taking generalization into consideration, orthogonality matrices are proposed to reduce the parameter space. Xing et al. (2015) first normalize the word embedding space into hyper-spheres and then apply an orthogonal matrix to map (or "rotate") the embedding spaces. Orthogonal mapping has shown its power in generalization and is generally applied.

2.3.3 Orthogonal Procrustes Problem

The mapping problem can be reduced to a linear algebra problem in the case of supervised orthogonal mapping, the orthogonal Procrustes problem. The problem is formulated as

$$R = \arg\min_{\Omega} \|\Omega A - B\|_2 \tag{2.2}$$

$$\text{s.t.} \quad \Omega^{\mathrm{T}}\Omega = \mathrm{I}. \tag{2.3}$$

The orthogonal Procrustes problem is applied in the supervised setting of BWE mapping by replacing A with the source language embeddings in the supervising lexicon, and B with the target language embeddings, and seeing R as the mapping matrix. The solution to orthogonal Procrustes problem is

$$BA^T = U\Sigma V^T, \tag{2.4}$$

$$R = UV^T, \tag{2.5}$$

where the first equation denotes singular value decomposition.

Even though the orthogonal Procrustes problem gives the optimized mapping matrix under the orthogonality constraint, it is not the only solution to the supervised orthogonal mapping. For example, instead of applying the orthogonal Procrustes problem, Xing et al. (2015) propose replacing all the singular values of the mapping matrix with one after every update of the matrix. In the method of Xing et al. (2015), additional constraints can be added to the optimizing process of the mapping matrix.

2.3.4 Evaluation

As mentioned above, the common evaluation task for BWE mapping is a ZSL task, BLI. The task is fulfilled by extracting the target word from the nearest neighbors of the mapped embedding of the source word. In extracting the nearest neighbors, the distance metric can also be Euclidean distance, cosine distance, or a more complicated distance metric - CSLS [33]. The BLI performance is usually best when the distance metric for nearest neighbor extraction is aligned with the distance metric for training target (CSLS is aligned with RCSLS).

CSLS is proposed to alleviate the hubness problem [34] that in high-dimensional spaces, a point tends to be the nearest neighbors of many points at the same time. The idea of CSLS is to punish the distance of the points that are the popular nearest neighbors. Whether a point is a popular nearest neighbor is represented with the mean similarity with its neighbors. The formula for CSLS is:

$$r_T(W\boldsymbol{x}_i) = \frac{1}{K} \sum_{\boldsymbol{y}_i \in \mathcal{N}_T(W\boldsymbol{x}_i)} \cos(W\boldsymbol{x}_i, \boldsymbol{y}_i) \tag{2.6}$$

$$\text{CSLS}(W\boldsymbol{x}_i, \boldsymbol{y}_i) = 2\cos(W\boldsymbol{x}_i, \boldsymbol{y}_i) - r_T(W\boldsymbol{x}_i) - r_S(\boldsymbol{y}_i) \tag{2.7}$$

where $\mathcal{N}_T(Wx_s)$ denotes the top K neighbors of $W\boldsymbol{x}_i$ in the target embedding space, r_S denotes the mean similarity of $\boldsymbol{y}_i$ to its neighbors. CSLS is usually the best nearest neighbor extraction distance metric (resulting in higher BLI scores), and commonly applied for BLI.

2.3.5 Considerations

As the prevailing of contextualized word embeddings, there is also a research topic on contextualized BWE mapping. However, it remains a question that how to do the mapping for contextualized word embeddings, and this topic is not in the concerns of this thesis.

The idea of embedding mapping originates in the bilingual setting, helps the ZSL task, BLI, by helping to transfer the knowledge from seen words to unseen words. But the embedding mapping can also be applied in other ZSL tasks, such as zero-shot RE. The specific technique that applies embedding mapping for zero-shot RE will be introduced in 3.2.2.

2.4 KBQA

KBQA is a variant of QA tasks. Like general QA, the input is a question, and the output is an answer. The main difference between KBQA and QA is the existence of a knowledge base. A knowledge base is a database storing real-world knowledge. As introduced in 2.2.3, the knowledge is stored as (head entity, relation, tail entity) triplets, and the knowledge base can be seen as a graph. In KBQA, there is a given knowledge base, and the questions are targeting the knowledge in that knowledge base (otherwise the system could not answer). Usually, the question gives the hints about the head entity and the relation, and the KBQA system is required to get the tail entity as the final answer. For example, the question *"Where was Douglas Adams educated?"* indicates it want to get the tail entity as the answer in the (*Douglas Adams, educated at, St. John's College*) knowledge triplet. It would be quick and easy for database systems to search the tail entity given the (head entity, relation) tuple. So the main task is to extract the head entity and the relation in the question.

The main task contains two parts: to extract the entity and to extract the relation. These two tasks look similar, but due to the distinct features of entities and relations, there are different methods to handle the two tasks. There are three main differences between entities and relations. First, the surface form (how an entity is written in natural languages) of an entity is usually in a continuous span, while the surface form of a relation is usually in several discontinuous spans. For example, in the question *"Where was Douglas Adams educated?"*, the entity surface form is "Douglas Adams", a

continuous span; the relation surface form is "Where was ... educated", two spans split by the entity span. Second, the surface forms of one entity are very close, but there are always largely different variants of the surface forms of a relation. For example, the surface forms of the entity "Douglas Adams" could be "Douglas Adams", "Mr. Adams", or "Douglas Noel Adams". But the surface forms of the relation "educated at" can be hardly enumerated, which includes "Where was ... educated", "Which college did ... study in", "Where did ... get his degree". Third, the number of entities is usually much larger than the number of relations in knowledge bases. Take Freebase [35] as an example. Freebase is one of the most common knowledge bases, and there are two popular subsets of it: FB2M and FB5M. In FB2M, there are 2,150,604 entities and only 6,701 relations; in FB5M, there are 4,904,397 entities and only 7,523 relations [36].

Because the number of entities is very large, it is difficult to predict the entity in an end-to-end approach; and because the entity span is continuous and the surface forms are kind of fixed, it is easy to restrict the candidate entities with the raw text information. Therefore, there are usually two steps to extract the entity: restrict the candidates, and predict the entity among those restricted candidate space.

The surface forms of the relations are too difficult to directly infer the relation information from. However, it is common to apply the predicted entity information to restrict the candidate space to only the relations connected to the predicted entities. This can reduce the search space from thousands of relations into less than 10 relations in average.

After predicting the entities and relations, the next step is to form the (head entity, relation) tuple for this question. However, because none of the prediction results is perfect, the top predicted entity and the top predicted relation might be wrong, and the predicted (head entity, relation) tuple constructed from the top one predicted entity and the top one predicted relation might not be in the knowledge base. Therefore, there is a step to select the final (head entity, relation) tuple based on the results of entity extraction and RE. There are various methods to get the final (head entity, relation) tuple. Some methods like BuboQA [37] rank the tuples based on the prediction scores of the entities and the relations. Some methods like EARL [38] rank the tuples based on the knowledge base structure, and some like KEQA [39] rank the tuples based on the entity and relation embedding space.

In order to better explain KBQA, an example of a baseline model, BuboQA, for KBQA is introduced.

2.4.1 BuboQA

BuboQA is a strong baseline for KBQA proposed by Mohammed et al. (2018) in the paper *Strong Baselines for Simple Question Answering over Knowledge Graphs with and without Neural Networks*. BuboQA contains a full pipeline for solving KBQA and consists of various components for parts of the pipeline. The pipeline contains three big parts: entity extraction, RE, and evidence integration. The entity extraction consists of two steps: entity detection to get the entity spans, and entity linking to predict the entity with the spans. The RE is to predict the relation in the query question. The evidence integration is for combining the scores of the predicted entity and relation candidates (to get the scores for (entity, relation) candidates).

Entity Extraction

As introduced above, the entity extraction is composed of two steps, the entity detection and the entity linking. They will be introduced separately.

The task of entity detection is to predict the entity span in the sentence. The authors deal with this task by predicting whether a token is an entity token or a non-entity token. The authors provide two approaches to do the prediction: by RNN and by conditional random field (CRF).

For RNN, the authors consider BiLSTM. Each token in the question is represented by the concatenation of the hidden states from the forward pass and the backward pass of the BiLSTM model. A classifier (composed of a linear layer, batch normalization, ReLU activation, dropout, and a mapping layer) is trained to predict whether each token is an entity token or a non-entity token taking the hidden state representation as the input. Besides RNN, CRF is also a popular model for sequence inputs and outputs. The CRF model in BuboQA is adopted from the approach of Finkel et al. (2005) [40], which combines features such as word positions, part-of-speech (POS) tags, character n-grams, etc. into the input. Experiments done by the authors show that BiLSTM is better than CRF on entity detection.

The task of entity linking is to link the entity spans (or entity tokens) into a node (representing an entity) of the knowledge graph. Considering that each entity is provided with a canonical label, the entity linking can be seen as an approximate string matching problem (match the predicted tokens to a canonical label). For each entity, the authors build an inverted index over n-grams (n $\in \{1, 2, 3\}$) in advance. In linking time, all corresponding n-grams of the predicted spans are generated and looked up in the inverted index for all matches. The matching starts from n=3, and lower n-grams are not considered if an exact match is found. After getting all candidate entities (which have n-gram matches with the predicted spans), they are ranked by the Levenshtein Distance between the predicted spans and the canonical labels (the Levenshtein Distance is an edit distance, denoting the least number of edits to change a word to the other).

RE

The task of RE is to predict the relation in the query question. This task is transferred to a classification problem, classifying a given question into a certain relation. The authors provide three approaches: RNN, convolutional neural network (CNN), and logistic regression (LR).

For RNN, the authors test two model structures: BiLSTM and bi-directional GRU (BiGRU). A classifier is applied over the RNN model. The classifier structure is the same as that for the entity detection RNN, except that the input to the classifier is only the hidden states (the concatenation of the forward pass and backward pass) of the last token.

For CNN, the authors applied the model proposed by Kim et al. (2014) [41]. The CNN takes the question word embeddings as the input, and feed the input to a convolutional layer with multiple convolutional core widths, a max-pooling layer, and a full connected with dropout and softmax output.

For LR, the authors consider two feature sets: tf-idf on unigrams and bigrams, and word embeddings + relation words. Tf-idf is the abbreviation of *term frequency-inverse document frequency* which is a common technique in NLP. Tf-idf gives weights to words in text sequences extracted from a corpus. Tf means term frequency, for a word (or a term) t_i in a sequence (or a document) d_j, $\mathrm{tf}_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$, where $n_{i,j}$ denotes how many times word t_j appears in the document d_j. Idf means inverse document

frequency, for word t_i, $\text{idf}_i = \lg \frac{|D|}{|\{j:t_i \in d_j\}|}$, where $|D|$ denotes the number of documents in the corpus. Tf-idf is the product of the tf and the idf, $\text{tf} - \text{idf}_{i,j} = \text{tf}_{i,j} \times \text{idf}_i$. For the second set of features, the embeddings of tokens from the question are averaged, and then concatenated with the summed relation word one-hot vector. 300 most frequent tokens from the names of the relations are selected to form the one-hot vector space, and the one-hot vector of the corresponding tokens are summed.

Evidence Integration

The task of evidence integration is to combine the scores of the predicted entity candidates and the predicted relation candidates to get final scores of (entity, relation) candidate pairs.

The (entity, relation) pair score is calculated by the product of the scores of the corresponding entity and relation. Ideally, every candidate entity shall form a pair with every candidate relation. But because not any relation can form an actual pair (that exists in the knowledge graph) with a certain entity, the pairs are pruned to only actual ones. The authors notice scoring ties that because some entities share an identical label (such as all people named "Adam Smith"), they would form (entity, relation) pairs with the same score. The authors break the scoring ties by favoring more popular entities that having more relations in the knowledge base or further having a mapping to Wikipedia.

Experiments

The authors evaluate the model on SimpleQuestions dataset [42]. The SimpleQuestions is a KBQA dataset based on Freebase [35]. The questions in the dataset are simple questions that there is only one relation in one question. The results of the BuboQA with different components on SimpleQuestions are shown in Table.2.1.

Table 2.1: The BuboQA accuracy on SimpleQuestions. (data from BuboQA's paper.)

Entity	Relation	Acc.
BiLSTM	BiGRU	74.9
BiLSTM	CNN	74.7
BiLSTM	LR (tf-idf)	68.3
BiLSTM	LR (GloVe+rel)	70.9
CRF	BiGRU	73.7
CRF	CNN	73.6
CRF	LR (tf-idf)	67.3
CRF	LR (GloVe+rel)	69.9

The BuboQA is not the states-of-the-art KBQA model but a strong baseline for KBQA. And it provides a feasible pipeline for further study.

KBQA is still far from being solved. ZSL with unseen relations is one of the difficult cases of KBQA.

Chapter 3

Literature Review

3.1 BWE Mapping

3.1.1 Multilingual Unsupervised or Supervised word Embeddings (MUSE)

MUSE [33] is a comprehensive work done by Conneau et al. (2017). In MUSE, benchmark datasets for supervised and unsupervised settings, as well as respective solution methods, are proposed.

MUSE Datasets

The evaluation task is BLI. As introduced in 2.3.1, BLI is the task of zero-shot word translation. Besides source and target word embedding spaces, the MUSE datasets also provide word translation pairs (as training data or test data).

Most large bilingual lexicons are created by machine translation (such as Google Translate). To alleviate the problem of word polysemy, the authors apply an internal translation tool to generate a high-quality bilingual lexicon of 100k pairs of words. They rank the word pairs by the word frequencies. They take the top 5,000 word pairs as the training data for the supervised MUSE dataset (the MUSE(S) dataset) and take the top 5001-6500 word pairs as the test data for both the MUSE(S) dataset and the unsupervised MUSE dataset (MUSE(U) dataset). The word translation candidate space is 200k words of the target language.

The authors provide 300 dimension embeddings trained by fastText on Wikipedia. They have done experiments comparing the influence of embedding algorithms and training datasets. The experiments show that using embeddings trained on Wikipedia shows significant improvements on the BLI than using embeddings trained on another dataset, the WaCky dataset [43]. (The authors hypothesize that the improvements are brought by the similar co-occurrence statistics of the Wikipedia corpora.) The authors also found that fastText embeddings outperform continuous bag-of-words (CBOW), due to more syntactic information about the words is encoded in the fastText embeddings.

Both the MUSE(S) and the MUSE(U) datasets consist of 12 language pairs composed by taking English and one of other 6 languages as source-target and target-source language pairs. Those languages are Spanish, French, German, Russian, Chinese, and Esperanto (denoted as ‘es’, ‘fr’, ‘de’, ‘ru’,

'zh', and 'eo' respectively, and English is denoted as 'en'). Actually, the MUSE(U) dataset is just the pruned version of the MUSE(S) dataset (removing the supervision lexicon). Therefore, the MUSE(S) dataset is usually just called as the MUSE dataset.

Mapping Algorithms

The authors propose a refinement procedure. The refinement is implemented after learning the mapping matrix. With the learned mapping matrix, a pseudo bilingual lexicon can be generated. The procedure of using the pseudo lexicon as a new training lexicon to fine-tune the mapping matrix is called refinement. In order to raise the quality of the pseudo lexicon, the authors only take the mutual nearest neighbors of a source language word and a target language word in the mapped embedding space as an entry of the generated bilingual lexicon.

For the supervised setting, the authors propose to apply orthogonal Procrustes and refinement iteratively. This method is called iterative Procrustes, or MUSE(S).

As for the unsupervised setting, the initial mapping matrix for refinement cannot be learned by the initial training lexicon (because there are no training lexicons). The authors propose to apply adversarial training to learn the initial mapping matrix. A discriminator is trained to tell the mapped source embeddings from the target embeddings. The discriminator is a two-layer perceptron with Leaky-ReLU activation functions, trained with the loss $\mathcal{L}_D$:

$$\mathcal{L}_D(\theta_D|W) = -\frac{1}{n}\sum_{i=1}^{n}\log P_{\theta_D}(\text{source}=1|W\boldsymbol{x}_i) - \frac{1}{m}\sum_{i=1}^{m}\log P_{\theta_D}(\text{source}=0|\boldsymbol{y}_i), \tag{3.1}$$

where θ_D denotes the parameters of the discriminator, W denotes the mapping matrix (of size 300×300), $\log P_{\theta_D}(\text{source}=1|z)$ denotes the probability that the embedding z comes from the source language, $\log P_{\theta_D}(\text{source}=0|z)$ denotes the probability that z comes from the target language, n and m denote the number of the source embeddings and target embeddings respectively, and $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ denote the source embedding and target embedding respectively. Adversarially, the mapping matrix is trained to fool the discriminator with the loss $\mathcal{L}_W$:

$$\mathcal{L}_D(W|\theta_D) = -\frac{1}{n}\sum_{i=1}^{n}\log P_{\theta_D}(\text{source}=0|W\boldsymbol{x}_i) - \frac{1}{m}\sum_{i=1}^{m}\log P_{\theta_D}(\text{source}=1|\boldsymbol{y}_i). \tag{3.2}$$

The parameters of the discriminator and the mapping matrix are updated with stochastic gradient descent. Please refer to the paper *Word Translation Without Parallel Data* [33] for further training details such as the learning rate. After trained the mapping matrix W adversarially, iterative Procrustes follows by taking the trained W as the initial mapping matrix. In order to keep the mapping matrix W consistent before and after the iterative Procrustes, the authors propose to update the mapping matrix W through the following rule during the training:

$$W \leftarrow (1+\beta)W - \beta(WW^T)W, \tag{3.3}$$

where the β is a hyper-parameter and performs well when set to be 0.01. This update is for keeping the W close to the manifold of orthogonal matrices during the training. Overall, the method for the unsupervised setting is called MUSE(U).

Nearest Neighbor Extraction

Nearest neighbors of mapped embeddings should be extracted from the target embedding space for generating the translation of the source words. Due to the hubness problem in the high dimensional spaces, the authors propose a distance metric, Cross-Domain Similarity Local Scaling (CSLS), as introduced in 2.3.4. They compare CSLS with two other metrics by experiments, NN (nearest neighbors by cosine distance) and inverted softmax (ISF) [26]. The idea of ISF is to extract the target word which maximizes the probability that the candidate target word would translate back to the source word. The inverse probability is represented as a softmax $P_{j \to i}$:

$$P_{j \to i} = \frac{e^{\beta \boldsymbol{y}_i^T W \boldsymbol{x}_j}}{\sum_n e^{\beta \boldsymbol{y}_i^T W \boldsymbol{x}_j}}, \tag{3.4}$$

where β denotes "inverse temperature", trained by the maximizing the log probability over the training dictionary,

$$\max_{\beta} \sum_{\text{pairs } ij} \ln(P)_{j \to i}. \tag{3.5}$$

Only a part of the source words are sampled in order to save computational time, and a normalization vector α is applied:

$$P_{j \to i} = \frac{e^{\beta \boldsymbol{y}_i^T W \boldsymbol{x}_j}}{\alpha_j \sum_n e^{\beta \boldsymbol{y}_i^T W \boldsymbol{x}_j}}. \tag{3.6}$$

With experiments, the authors show the superiority of CSLS which outperforms NN and ISF at most cases.

The MUSE models are strong on etymologically similar language pairs, e.g. MUSE models have achieved over 80% BLI P@1 on en-es, es-en, en-fr, and fr-en pairs. However, they are weak on etymologically distant language pairs, e.g. the BLI P@1 on en-zh is 42.7% and 32.5% by MUSE(S) and MUSE(U). Søgaard et al. (2018) [44] report that the embedding spaces are non-isomorphic especially across distant languages, resulting in the worse performance of orthogonal mappings. The problem means a large space left for improvements to MUSE models.

3.1.2 Bilingual Lexicon Induction with Semi-Supervision (BLISS)

BLISS is a strong model in BWE mapping proposed by Patra et al. (2019) [45]. The idea of BLISS contains two keys: semi-supervision and weak orthogonality constraint.

Semi-supervision

Referring to the experiment results of MUSE, MUSE(U) is better than MUSE(S) for etymologically similar language pairs, and MUSE(S) is better than MUSE(U) for etymologically distant language pairs. This fact implies the advantages and drawbacks of the supervised methods and the unsupervised

methods on different occasions. The unsupervised methods are good at aligning clusters of words, but not good at fine-grained alignment inside the clusters. The authors have done experiments on a toy dataset to demonstrate this feature of the unsupervised methods.

The authors propose to alleviate the in-cluster alignment problem of the unsupervised methods by providing supervision of some anchor point pairs inside the clusters, which is the semi-supervision method. In this method, the mapping matrix W is trained adversarially with a discriminator D, and the mapping matrix W is also trained from supervision. The adversarial loss for the discriminator is the same as $\mathcal{L}_D(\theta_D|W)$ in the Eq.3.1, denoted as $\mathcal{L}_{D|W}$ in following texts. The adversarial loss for the mapping matrix W is the same as $\mathcal{L}_D(W|\theta_D)$ in the Eq.3.2, denoted as $\mathcal{L}_{W|D}$ in following texts. And the supervised loss for the mapping matrix W is:

$$\mathcal{L}_{W|S} = -\frac{1}{|S|} \sum_{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in S} d_s(W\boldsymbol{x}_i, j_i), \tag{3.7}$$

where S denotes the supervision set (composed of $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ embedding pairs), and $d_s(,)$ denotes the similarity metric between two embeddings.

Weak Orthogonality Constraint

Orthogonal mapping is not achieving BLI P@1 higher than 95% on any language pair from the MUSE dataset, and performs much worse on etymologically distant language pairs than on etymologically similar language pairs.

As reported by Søgaard et al. (2018) [44], the embedding spaces are non-isomorphic across languages, especially across etymologically distant languages. To evaluate the effect of the non-isomorphism of the embedding spaces to the mapping performance, Søgaard et al. propose a graph similarity metric to quantify the extent to which the embeddings spaces of a language pair are non-isomorphic. They transform the embedding spaces of a language pair into adjacency matrices A_1 and A_2. Then they calculated the Laplacians of the two nearest neighbor graphs:

$$\begin{aligned} L_1 &= D_1 - A_1, \\ L_2 &= D_2 - A_2, \end{aligned} \tag{3.8}$$

where the D_1 and D_2 are respective degree matrices of A_1 and A_2 (a degree matrix D of a matrix or of a graph G is a diagonal matrix defined as: $D_{ij} := \begin{cases} \deg(\boldsymbol{v}_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$, where $\boldsymbol{v}_i$ denotes the vertices of the graph, and deg() denotes the number of edges with an end at a vertex). Eigenvalues of the Laplacians are calculated, and top (largest) k eigenvalues are collected for next step of calculation. The k is set such that the sum of the top k eigenvalues is smaller than 90% of the sum of all eigenvalues, and the smaller k of the two Laplacians is selected that

$$\min_j \{\frac{\sum_{i=i}^{k} \lambda_{j_i}}{\sum_{i=i}^{n} \lambda_{j_i}} > 0.9\}, \tag{3.9}$$

where λ_{j_i} denotes the i-th eigenvalue of L_j, and n denotes the dimension of the Laplacians (also the number of eigenvalues of a Laplacian). The graph distance Δ is defined as:

$$\Delta = \sum_{i=1}^{k} (\lambda_{1_i} - \lambda_{2_i})^2. \tag{3.10}$$

The larger the Δ is, the more distant, i.e. more non-isomorphic, two graphs, or embedding spaces, are. Søgaard et al. Experiments done by Søgaard et al. show strong correlation ($\rho = 0.89$) between BLI P@1 score (stand for mapping performance) and graph similarity under orthogonal mapping.

The authors (Patra et al.) propose applying Gromov-Hausdorff (GH) distance to quantify how well orthogonal mapping is at mapping different language pairs. The Hausdorff distance measures the worst case of a distance $d(,)$ between two spaces, defined as:

$$\mathcal{H}(\mathcal{X}, \mathcal{Y}) = \max\{\sup_{x \in \mathcal{X}} \inf_{y \in \mathcal{Y}} d(\boldsymbol{x}, \boldsymbol{y}), \sup_{y \in \mathcal{Y}} \inf_{x \in \mathcal{X}} d(\boldsymbol{x}, \boldsymbol{y})\}, \tag{3.11}$$

where $\mathcal{X}$ and $\mathcal{Y}$ denote two metric spaces. And Gromov-Hausdorff distance gives the minimum over all isometric transforms between $\mathcal{X}$ and $\mathcal{Y}$ that:

$$\mathcal{GH}(\mathcal{X}, \mathcal{Y}) = \inf_{f,g} \mathcal{H}(f(\mathcal{X}), g(\mathcal{Y})), \tag{3.12}$$

where f and g belong to isometric transform set. (Note that orthogonal transforms are equivalent to isometric transforms in the case that the embedding spaces are mean centered.) The authors find that GH distance are larger between etymologically distant language pairs.

All these observations expose the drawbacks of the orthogonal mapping. Therefore, the authors train the mapping matrix with a weak orthogonality constraint to achieve approximately orthogonal mapping. The weak orthogonality constraint is performed by applying a orthogonality loss $L_{W|O}$ to the mapping matrix W:

$$\mathcal{L}_{W|O} = -\frac{1}{|X|} \sum_{\boldsymbol{x}_i \in X} d_a(\boldsymbol{x}_i, W^T W \boldsymbol{x}_i), \tag{3.13}$$

where $d_a(,)$ is a distance metric between embeddings. (Note that if W is orthogonal, an embedding $\boldsymbol{x}_i$ shall be reconstructed by $W^T W \boldsymbol{x}_i$.)

The BLISS Model

As introduced above, the BLISS model contains two part, a discriminator D and a mapping matrix W. The training loss for the discriminator is $\mathcal{L}_d(D|W)$, introduced in Eq.3.1. The training loss for the mapping matrix contains three part, $\mathcal{L}_{W\|D}$ (Eq.3.2), $\mathcal{L}_{W\|S}$ (Eq.3.7), and $\mathcal{L}_{W|O}$ (Eq.3.13). The final loss for the mapping matrix is:

$$\mathcal{L} = \mathcal{L}_{W\|D} + \mathcal{L}_{W\|S} + \mathcal{L}_{W|O}. \tag{3.14}$$

The $d_s(,)$ in the Eq.3.7 is the CSLS similarity metric. The mapping loss with CSLS similarity metric is called relaxed CSLS (RCSLS) loss [28] for it is applied as a weak (relaxed) constraint. The authors

also test the BLISS with cosine distance, which is not as good as the BLISS with the RCSLS loss, and the BLISS with the RCSLS loss is referred as the BLISS(R) in the original paper. And the $d_a(,)$ in the Eq.3.13 is the cosine distance.

BLISS benefits from both the supervised training and unsupervised training, resulting in state-of-the-art mapping performance (at the time the paper is published) on 15 of 18 language pairs on the MUSE dataset. The weak orthogonality constraint helps to improve the mapping performance on distant (in GH distance) language pairs.

BLISS also applies refinement after getting the mapping matrix with the semi-supervised training. The authors introduce a hubness filtering mechanism to filter out words that are hubs of the target language (words with neighbors more than a threshold are filtered out). This filtering results in a small improvement in the mapping performance.

The experiment results of the BLISS model comparing with MUSE models on the MUSE datase is shown in Table.3.1.

Table 3.1: The BLISS BLI performance on the MUSE dataset comparing with MUSE models. (data from BLISS's paper.)

Method	en-es	es-en	en-fr	fr-en	en-de	de-en	en-ru	ru-en	en-zh	zh-en
MUSE(S)	81.4	82.9	81.1	82.4	73.5	72.4	51.7	63.7	42.7	36.7
MUSE(U)	81.7	83.3	82.3	82.1	74.0	72.2	44.0	59.1	32.5	31.4
BLISS	**84.3**	**86.2**	**83.9**	**84.7**	**79.1**	**76.6**	**51.7**	**67.7**	**48.7**	**47.3**

3.2 KBQA

3.2.1 Hierarchical Residual BiLSTM (HR-BiLSTM)

Yu et al. (2017) propose an improved KBQA pipeline and a new RE model in their paper "Improved Neural Relation Detection for Knowledge Base Question Answering" [46]. The HR-BiLSTM is the RE model proposed by them.

The HR-BiLSTM Model

HR-BiLSTM takes the RE task as a ranking problem rather than a classification problem. The inputs to the HR-BiLSTM are the combination of a question and a relation, and the task of HR-BiLSTM is to give higher scores to a positive (question, relation) set, and lower scores to negative ones.

As shown in Fig.3.1, the structure of HR-BiLSTM consists of two separate BiLSTMs, a residual connection, max-pooling layers, and a cosine similarity calculation.

As introduced above, The input is a (question, relation) pair. The question input is question tokens. The relation input contains two parts: the relation, and the relation words. The relation inputs are the embeddings of the relations (the embeddings can be pre-trained, but the authors just apply randomly initialized embeddings). Taking two forms of relation representations helps the model

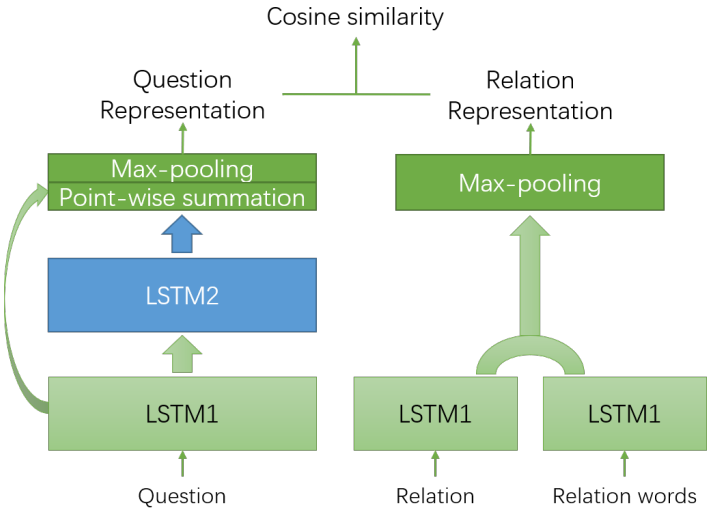


Figure 3.1: An overview of HR-BiLSTM.

to recognize the relations from different granularity. After getting the relation level hidden states and relation word level hidden states from the BiLSTM1, a max-pooling layer is applied on these hidden states to get a final representation of the relation h^r (a vector whose length is the same as the hidden states).

There are two BiLSTMs for retrieving the question representation. The hierarchical structure of the two BiLSTMs is for getting different abstractions of the question representations, such as lower abstractions on word information, and higher abstractions on relation and entity information. The hidden state lengths of the BiLSTM1 and BiLSTM2 are the same, such that the hidden states from the two BiLSTMs can be summed up point-wisely (forming a residual connection). The sum is sent to a max-pooling layer for the final representation of the question h^q, which is a vector with the same length of the BiLSTM hidden states, and of the relation representation.

The output of the HR-BiLSTM is a cosine similarity between the question representation h^q and the relation representation h^r. The cosine similarity can be seen as a score of a relation given the question: $s_{\text{rel}}(r; q)$.

In the training of the HR-BiLSTM, given a gold (question, relation) pair (the gold relation is denoted as r^+), several negative relations r^-s are randomly sampled from all other relations. A hinge loss is proposed for HR-BiLSTM:

$$l_{rel} = \max\{0, \gamma - s_{\text{rel}}(r^+; q) + s_{\text{rel}}(r^-; q)\}, \tag{3.15}$$

where γ is a margin hyper-parameter for the loss. It is calculated between the $s_{\text{rel}}(r^+; q)$ and each other $s_{\text{rel}}(r^-; q)$.

Improved RE Pipeline

Since the RE of HR-BiLSTM is based on ranking scores of relations, it would be time costing to calculate the scores of all relations in the evaluation phase (there would be thousands of relations in a knowledge base). The authors propose a two-step RE pipeline to constrain the search space of candidate relations with entity linking results.

The pipeline goes as the following process. First, K entity candidates are generated with entity extraction, and the relation search space can be constrained to only the ones that have connections to at least one of the candidate entities in the knowledge base. Then, the entities are re-ranked by a new score $s_{\text{rerank}}(e; q)$ which is calculated based on the entity linking score $s_{\text{linker}}(e; q)$ and the RE score $s_{\text{rel}}(r; q)$:

$$s_{\text{rerank}}(e; q) = \alpha \cdot s_{\text{linker}}(e; q) + (1 - \alpha) \cdot \max_{r \in R_q^l \cap R_e} s_{\text{rel}}(r; q), \tag{3.16}$$

where R_q^l denotes the set of best l relations based on the RE score, R_e denotes the set of relations that connect to the entity e, and α is a hyper-parameter. $K' < K$ top entities are selected based on the re-ranking score $s_{\text{rerank}}(e; q)$. Finally, the RE is implemented again constrained in the search space of relations connected to the K' entities with a *reformated question text*. The *reformated question text* is generated by replacing the tokens of the topic entity with a special token ¡e¿.

After the two-step RE pipeline, the final scores of the (entity, relation) pairs are calculated:

$$s(\hat{\boldsymbol{e}}, \hat{r}; q) = \max_{e \in EL'_{k'}(q), r \in R_e} (\beta \cdot s_{\text{rerank}}(e; q) + (1 - \beta) \cdot s_{\text{rel}}(r; e, q)), \tag{3.17}$$

where $EL'_{k'}(q)$ denotes the re-ranked K' entities, and β is a hyper-parameter.

HR-BiLSTM provides a strong model for RE. It has achieved 93.3% accuracy on SimpleQuestions RE (based on gold entities). The overall KBQA accuracy on SimpleQuestions is 77.0% with the proposed pipeline.

3.2.2 KBQA-Adapter

KBQA-Adapter is a modified HR-BiLSTM model proposed by Wu et al. (2019) [1] in the paper *Learning Representation Mapping for Relation Detection in Knowledge Base Question Answering*. This model is proposed to deal with zero-shot RE problem. Together with this model, a modified SimpleQuestions dataset focusing on zero-shot RE, SimpleQuestion-Balance (SQB), is also proposed.

The Adapters

Zero-shot relations (or unseen relations) do not appear in the training data, therefore difficult to train models that are good at recognizing those relations. There are pre-trained relation embeddings which might help the models recognize unseen relations. But the relation embeddings are updated through the training of RE model. The embeddings of the seen relations are trained toward the best positions, but those of unseen relations are not. Even though the unseen relations might be used as negative examples in training time, the embeddings of the unseen relations are trained to be pushed

away from some wrong points, but usually are not pushed toward the correct direction. After the training, the structure of the original relation embedding space is broke, therefore it becomes hard for the RE model to recognize the unseen relations.

The authors first try freezing the relation embeddings in training time, and find clear improvement on unseen accuracy. Therefore, they are inspired to apply some technique, to keep the structure of the relation embedding space while updating the whole space. That technique is the adapters. The idea of the adapters is to map the original relation embedding space to the updated embedding space, which is inspired from the BWE mapping. As there are different models for BWE mapping, the authors propose four different adapters. They are the Basic Adapter, the Adversarial Adapter, and these two adapters with reconstruction loss, as shown in Fig.3.2. The adapters are applied to HR-BiLSTM on the top of the relation input, as shown in Fig.3.3.

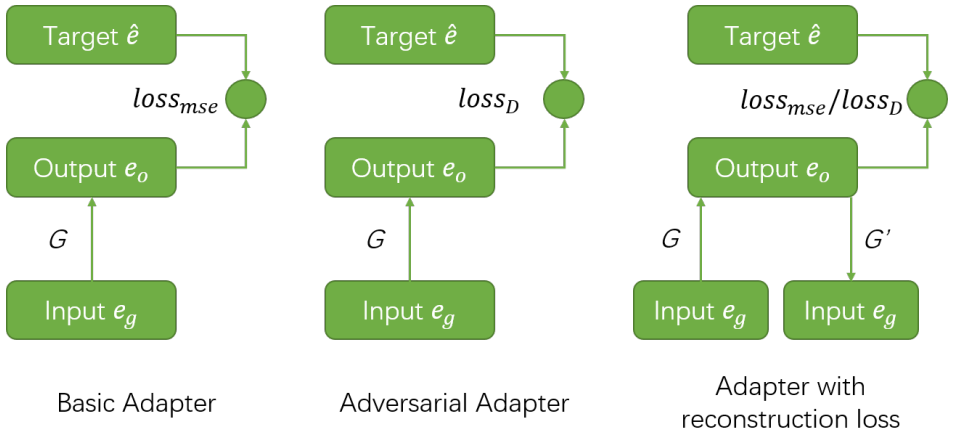


Figure 3.2: An overview of different adapters, where G denotes a linear mapping function, G' denotes a reverse linear mapping function, and D denotes a discriminator.

The **Basic Adapter** is composed of a linear mapping function G (or generator), which is optimized though the loss $\mathcal{L}_{adapter}$:

$$\mathcal{L}_{adapter} = \sum_{r \in S} loss_{\text{mse}}(\hat{\boldsymbol{e}}, G(\boldsymbol{e_g})), \tag{3.18}$$

where S denotes the seen relation set, $\boldsymbol{e_g}$ denotes the original relation embedding (or the general embedding), $\hat{\boldsymbol{e}}$ denotes the updated embedding, and and the $loss_{\text{mse}}(,)$ denotes mean square error (MSE) loss that

$$loss_{\text{mse}}(\hat{\boldsymbol{e}}, G(\boldsymbol{e_g})) = \|\hat{\boldsymbol{e}} - G(\boldsymbol{e_g})\|_2^2. \tag{3.19}$$

The the generator G is updated together with the training of the RE model.

The **Adversarial Adapter** is composed of a generator G and a discriminator D. The discriminator D is a feed forward neural network without sigmoid function in the last layer, designed to tell

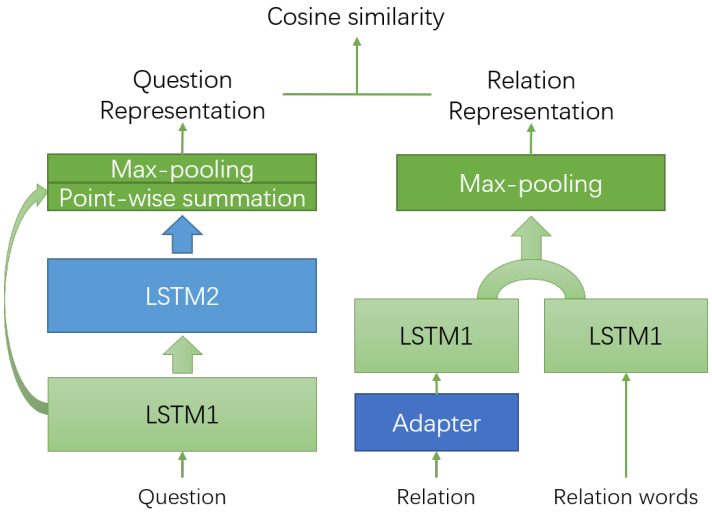


Figure 3.3: An overview of the KBQA-Adapter. The adapter model is applied above the relation input, transfering the general relation embeddings into adapted embeddings as the new input to upper models.

generated embedding $\boldsymbol{e_o}$s from the updated embeddings $\hat{\boldsymbol{e}}$. The G and the D are trained adversarially, like what has been introduced in 3.1.1. The loss for the discriminator $loss_D$ and the loss for the generator $loss_G$ are:

$$loss_D = \mathbb{E}_{r \in S}[D(G(\boldsymbol{e_g}))] - \mathbb{E}_{r \in S}[D(\hat{\boldsymbol{e}})], \tag{3.20}$$

$$loss_G = -\mathbb{E}_{r \in s}[D(G(\boldsymbol{e_g}))]. \tag{3.21}$$

The G and the D are trained with WassersteinGAN [47]. The training of the G and the D are performed alternatively.

The **reconstruction loss** is proposed to prevent the generator from generating only seen embeddings. The idea is to train a reverse generator G' to map the mapped embeddings back to the original embeddings. The reconstruction loss is:

$$loss_R = \sum_{r \in S \cup U} \|G'(G(\boldsymbol{e_g})) - \boldsymbol{e_g}\|_2^2, \tag{3.22}$$

where U denotes the unseen relation set. This loss can be added to both the Basic Adapter or the Adversarial Adapter.

The Relation Embeddings

In the KBQA-Adapter, the authors apply the JointNRE embeddings proposed by Han et al. (2018) [24]. As introduced in 2.2.3, the JointNRE trains relation embeddings jointly with word embeddings.

The relation embeddings and word embeddings are in the same embedding space, thus can be easily handled by the same BiLSTM.

The authors find that the training text corpus in JointNRE may not cover all relations. They propose to adapt the relation embeddings in the training of the relation embeddings. They pre-train the relation embeddings with the TransE algorithm [20]. Then they apply an adapter (to relation embeddings) in the fine-tuning of the embeddings with JointNRE, and get JointNRE* embeddings. Applying the JointNRE* embeddings show a little improvement on the original JointNRE embeddings.

SQB

SimpleQuestions, introduced in 2.4.1, is a good large-scale dataset for KBQA. However, most of the relations in SimpleQuestions development set (dev set) and test set have been seen in the training set. Only 0.65% examples in the dev set and 0.74% examples in the test set are with unseen relations. The authors propose a modified dataset called SimpleQuestion-Balance (SQB) where the seen and unseen relations are balanced.

The SQB dataset is re-organized from SimpleQuestions. The SimpleQuestions examples are randomly shuffled and split into 5 sets, train, dev-seen, dev-unseen, test-seen, and test-unseen, while the overlapping of relations, and the percentage of seen/unseen samples are checked. The statistics information of the SimpleQuestions and SQB datasets are shown in Table.3.2. The authors do the

Table 3.2: The statistics information of the SimpleQuestions and SQB datasets.

Datasets	SimpleQuestion	SQB
Train	75,910	75,819
Dev-seen	10,774	5,383
Dev-unseen	71	5,758
Test-seen	21,526	10,766
Test-unseen	161	10,717

re-organizion 10 times and create 10 folds of the SQB dataset, which can be used for cross validation.

The authors do 10-fold experiments on SQB with different adapters, the results are shown in Table.3.3. The results show the distinct improvement on unseen test accuracy on HR-BiLSTM brought by the adapters. The HR-BiLSTM + Adversarial Adapter with reconstruction loss is chosen as the final model for the KBQA-Adapter for the best unseen accuracy and all accuracy. In later part of this thesis, any mentioned KBQA-Adapter without explanations is referring the HR-BiLSTM + Adversarial Adapter with reconstruction loss model.

3.2.3 Knowledge Graph Embedding Based Question Answering (KEQA)

KEQA is a pipeline for KBQA proposed by Huang et al. (2019) in the paper *Knowledge Graph Embedding Based Question Answering* [39]. KEQA pays much attention to the embedding space. The pipeline consists of entity extraction, RE, and a joint search. The entity extraction is composed of

Table 3.3: The RE accuracy on the SQB dataset. The RE is based on gold entity linking results. The relation embeddings are the JointNRE embeddings. The no fine-tune row denotes the experiment about not fine-tuning the relation embedding space. (Data from KBQA-Adapter's paper.)

Model	Average Accuracy on SQB		
	Test-seen	Test-unseen	ALL
HR-BiLSTM	**93.5±0.6**	33.0±5.7	63.3±3.6
+ no fine-tune	93.4±0.7	57.8±9.8	75.6±5.0
+ Basic-Adapter	92.8±0.7	76.0±7.5	84.5±3.5
+ reconstruction	93.0±0.5	76.1±7.0	84.6±3.3
+ Adversarial-Adapter	92.6±0.9	77.1±7.1	**84.9±3.2**
+ reconstruction [Final]	92.4±0.8	**77.3±7.6**	**84.9±3.5**

two steps, the entity detection and the entity prediction. The entity prediction model and the relation prediction model share the same model structure.

The authors have tested the KEQA on SimpleQuestions, and modified SimpleQuestions to test the zero-shot ability of KEQA.

Entity Detection

Similar with the BuboQA 2.4.1, the entity detection is implemented by predicting whether each token is an entity token or a non-entity token.

In KEQA entity detection model, the input question tokens are sent to a BiLSTM. The hidden states from the forward pass and the backward pass of the BiLSTM are concatenated, and sent to a full connected layer followed by a softmax function. The model is called head entity detection model (HED). The predicted entity tokens form a span or many spans (if the entities are not successive). Each successive span is considered as a independent entity name.

After getting the predicted entity spans, the entity prediction search space is constrained to entities that with the same names as the predicted spans or with the names that contain the predicted spans.

Entity and Relation Prediction

Because the models for entity prediction and relation prediction are the same in structure, they are introduced together. The model is composed of a BiLSTM and attention mechanism. The inputs are the word embedding $\boldsymbol{x}$s of the input question, and the output is the predicted entity embedding $\hat{\boldsymbol{e}}_h$ or the predicted relation embedding $\hat{\boldsymbol{p}}_l$. The model structure is shown in Fig.3.4.

After the BiLSTM, the input $\boldsymbol{x}_j$ is transformed to BiLSTM hidden states $\vec{\boldsymbol{h}}_j$ and $\overleftarrow{\boldsymbol{h}}_j$ (from forward pass and backward pass respectively), where j denotes the respective items for the j-th input token. The $\vec{\boldsymbol{h}}_j$ and $\overleftarrow{\boldsymbol{h}}_j$ are concatenated to be $h_j = [\vec{\boldsymbol{h}}_j, \overleftarrow{\boldsymbol{h}}_j]$. Then the attention weight α_j of the j-th

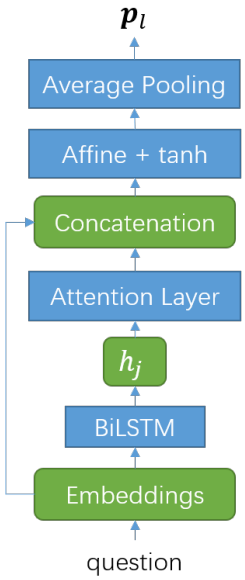


Figure 3.4: An overview of KEQA.

token is calculated as:

$$\alpha_j = \frac{\exp(q_j)}{\sum_{i=1}^{L} \exp(q_i)}, \tag{3.23}$$

$$q_j = \tanh(\boldsymbol{w}^T[\boldsymbol{x}_j; \boldsymbol{h}_j] + b_q), \tag{3.24}$$

where L denotes the lengths of the input sequence, and $\boldsymbol{w}$ and b_q are the attention parameters. The attention weight is applied to form another hidden state $\boldsymbol{s_j} = [\boldsymbol{x}_j; \alpha_j \boldsymbol{h}_j]$. The $\boldsymbol{s}_j$ is transformed to $\boldsymbol{r}_j \in \mathbb{R}^{d \times 1}$ where d is the dimension of the entity and relation embeddings. All the $\boldsymbol{r}_j$s are averaged to get the predicted entity (or relation) embeddings:

$$\hat{\boldsymbol{e}}_h \text{ (or } \hat{\boldsymbol{p}}_l) = \frac{1}{L} \sum_{j=1}^{L} \boldsymbol{r}_j^T. \tag{3.25}$$

The training loss for this model is the MSE between the predicted embedding and the embedding of the true entity (or the true relation).

Joint Search

The joint search, unlike the evidence integration in the BuboQA (see 2.4.1) which targets in a final (entity (or head entity), relation) pair, aims at geting a fact, the (head entity, relation, tail entity) (or (h, l, t)) triplet.

The main inputs to the joint search are the predicted (head) entity embedding $\hat{\boldsymbol{e}}_h$, the predicted relation embedding $\hat{\boldsymbol{p}}_l$, and the inferred tail entity embedding $\hat{\boldsymbol{e}}_t$. Here the inferred tail entity embedding $\hat{\boldsymbol{e}}_t$ is calculated by $\hat{\boldsymbol{e}}_t \approx f(\hat{\boldsymbol{e}}_h, \hat{\boldsymbol{p}}_l)$, where the $f(,)$ is the function in the main formula of the relation embeddings. The formula, for example, is $\hat{\boldsymbol{e}}_t \approx \hat{\boldsymbol{e}}_h + \hat{\boldsymbol{p}}_l$ in the TransE, and is $\hat{\boldsymbol{e}}_t M_l \approx \hat{\boldsymbol{e}}_h M_l + \hat{\boldsymbol{p}}_l$ in the TransR [22] where M_l is a parameter, the transform matrix of the relation embeddings.

A key concept in the joint search is the joint distance metric for selecing the best fact triplet that:

$$\begin{aligned} \underset{(h,l,t)\in\mathcal{C}}{\text{minimize}} \; & \|\boldsymbol{p}_l - \hat{\boldsymbol{p}}_l\|_2 + \beta_1\|\boldsymbol{e}_h - \hat{\boldsymbol{e}}_h\|_2 + \beta_2\|f(\hat{\boldsymbol{e}}_h, \hat{\boldsymbol{p}}_l) - \hat{\boldsymbol{e}}_t\|_2 \\ & - \beta_3 sim[n(h), HED_{\text{entity}}] - \beta_4[n(l), HED_{\text{non}}], \end{aligned} \tag{3.26}$$

where $\mathcal{C}$ denotes the candidate fact set, $\boldsymbol{e}_h$, $\boldsymbol{p}_l$, and $\boldsymbol{e}_t$ denotes the head entity embedding, the relation embedding, and the tail entity embedding of the fact triplet (h, l, t), $n()$ denotes the name of the entity or the relation, HED_{entity} or HED_{non} denotes the predicted entity tokens or the non-entity tokens by the HED model, $sim[,]$ denotes the similarity of two strings, and β_1, β_2, β_3, and β_4 are the hyper-parameters.

The candidate facts $\mathcal{C}$ are all the facts with the possible head entities decided by the head entity detection process. With the Eq.3.26, the top predicted fact (h^*, l^*, t^*) is extracted, where the t^* is the predicted answer to the KBQA question.

SimpleQ_Missing

The authors propose to modify the SimpleQuestions dataset to test the zero-shot ability of the KEQA model. The new dataset is called the SimpleQ_Missing. It is created by shuffling the questions in the original SimpleQuestions dataset. The authors first randomly split the relations into three groups, and then assign the questions to three groups according to the relations to form the training set, the dev set, and the test set. As a result, there are no intersecting relations across any two sets of the three sets. The authors get 75,474 questions in the training set, 11,017 questions in the dev set, and 21,951 questions in the test set.

The authors do experiments on the SimpleQuestions dataset and the SimpleQ_Missing dataset, based on KEQA with different relation embeddings. The results can be seen in Table.3.4. Here the KEQA_noEmbed means applying random embeddings as the relation (and entity) embeddings, KEQA_TransE applies the TransE embeddings, KEQA_TransH applies the TransH [21] embeddings, and KEQA_TransR applies the TransR embeddings. The table shows that the KEQA performs best with the TransE embeddings, and shows the zero-shot ability of the KEQA models.

Table 3.4: The RE accuracy of KEQA on the SimpleQuestions and the SimpleQ_Missing. (Data from the KEQA's paper.)

Model	SimpleQuestions	SimpleQ_Missing
KEQA_noEmbed	73.1	38.6
KEQA_TransE	**75.4** (+3.1%)	**41.8** (+8.3%)
KEQA_TransH	74.9 (+2.5%)	41.1 (+6.5%)
KEQA_TransR	75.3 (+3.0%)	41.7 (+8.0%)

Chapter 4

Investigation on the Approximate Orthogonality in BWE Mapping for BLI

The embedding mapping is an important element in the ZSL with embedding spaces. This chapter introduces the efforts to improve the BWE mapping on the ZSL task, BLI (see 2.3.1). The efforts are on the approximate orthogonality in the mapping.

4.1 The Influence of Approximate Orthogonality

4.1.1 The Idea of the Investigation

As introduced in 2.3.2, applying orthogonality constraint to the mapping matrix W is common in BWE mapping. There are strong constraints such as that in the orthogonal Procrustes. Xing et al. (2015) [27] also propose a strong orthogonal constraint that replacing all the singular values of the mapping matrix W with 1 after every update of the matrix.

However, Patra et al. 2019 [45] argue that strictly orthogonal mapping is not optimal for BWE alignment, since Søgaard et al. [44] have shown that the word embedding spaces are not isomorphic. Therefore there are also weak constraints such as that applied in the MUSE(U) (3.1.1), and that applied in the BLISS (3.1.2). The weak constraints result in approximately orthogonal mapping matrices. However, even though the weak orthogonality constraints are applied in the MUSE(U) and the BLISS, those models also apply iterative Procrustes (3.1.1) as the last step, forcing the final mapping matrix to be orthogonal. Since the approximate orthogonality helps BWE mapping, experiments of the approximate orthogonality should be done. In this part of the thesis, the influence of the approximate orthogonality of the mapping matrices is investigated. The investigation is implemented by watching the mapping performance on the BLI task of the mapping matrices that are approximately orthogonal to different extents.

4.1.2 The Investigation Model

In order to get the mapping matrices with different degrees of orthogonality approximation, an investigation model is proposed where the training of the mapping matrices follows the proposed loss:

$$\mathcal{L}_{\text{map}} = \|WX - Y\|_2, \tag{4.1}$$

$$\mathcal{L}_{\text{orth}} = \|WW^T - I\|_2, \tag{4.2}$$

$$\mathcal{L}_{\text{total}} = \frac{\alpha}{\alpha+1}\mathcal{L}_{\text{map}} + \frac{1}{\alpha+1}\mathcal{L}_{\text{orth}}, \tag{4.3}$$

where the X and the Y denotes the training word embedding spaces from the source language and the target language respectively (the embeddings of known translation word pairs are in the same rows of the two matrices, the same as those in the Fig.2.1), I denotes the identity matrix whose size is the same as the embedding length, and the α here is a hyper-parameter about the weights of the losses. The mapping loss $\mathcal{L}_{\text{map}}$ guides the mapping matrix W to being a good mapping matrix for the training data. The orthogonality constraint loss $\mathcal{L}_{\text{map}}$ put a weak orthogonality constraint on the mapping matrix, as the definition of an orthogonal matrix Q is that $QQ^T = I$. And weight hyper-parameter α controls the hardness of the orthogonality constraint.

This investigation model is simple, with the same mapping target with the orthogonal Procrustes. The only different thing in the concept is that the orthogonal Procrustes constrains the mapping matrix under absolute orthogonality, while this investigation model constrains the mapping matrix under approximate orthogonality. Therefore, the comparison of the two methods can be seen as the comparison of the absolute orthogonality and approximate orthogonality in the BWE mapping.

4.1.3 The Experiment Settings

The experiments are done on the MUSE dataset [33] (data acquired from the GitHub repository of the MUSE[1]). The MUSE dataset provides the supervising bilingual lexicon, the test bilingual lexicon, and the word embeddings (whose lengths are 300), for various language pairs. Same as the experiments of the BLISS, ten language pairs composed by pairing en with other five languages (es, fr, de, ru, and zh) are selected. The evaluation metric is the BLI P@1 score based on CSLS (3.1.1) nearest neighbor extraction.

The training of the model is based on stochastic gradient descent (SGD). The learning rate is set to be 0.02. If the training loss is not going down for a certain tolerance number (=500) of iterations, the learning rate is reduced by 2%. The training ends when the learning rate is lower than 5e-7, or when the training has passed the maximum iteration times (=50,000).

The hyper-parameter α is tuned from 0.05 and 20.

[1] https://github.com/facebookresearch/MUSE

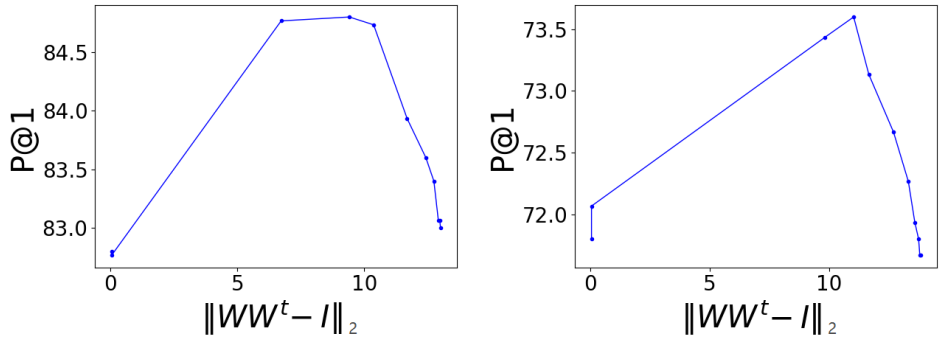
4.1.4 The BLI Curve with Respect to the Orthogonality Approximation Degree (OAD)

In this thesis, the OAD of a matrix is defined as:

$$\text{OAD}(Q) = \|QQ^T - I\|_2. \tag{4.4}$$

If the degree is 0, then the matrix is absolutely orthogonal, and the approximation of the orthogonality of the matrix is larger if the OAD is larger. An intuitive phenomenon is discovered that if the weight hyper-parameter α changes, the OAD changes monotonically. If the α is larger, then the training focus of the mapping matrix W is paid more on the mapping performance, and thus the optimized matrix W^* is more approximately orthogonal, that the OAD is larger. On the contrary, if the α is smaller, the OAD is smaller.

And if we look at the BLI P@1 scores (on the test set) based on different α, we will notice that there is always a best OAD resulting in the highest BLI P@1 for a language pair. And the BLI P@1 is lower as OAD deviate from the best point. Fig.4.1 shows two clear curves from es-en language pair and de-en language pair describing this phenomenon. From the curves we can see the BLI P@1 scores reach the highest point when the OAD is around 10 for es-en, and when the OAD is around 11 for de-en.



(a) es-en (b) de-en

Figure 4.1: The BLI P@1 for es-en (4.1a) and de-en (4.1b) when the OAD (denotes by $\|WW^t - I\|_2$) varies.

4.1.5 The Experiment Results

Since the the mapping performance varies with the OAD (of the optimized) as introduced in 4.1.4, and the OAD is controlled by the weight hyper-parameter α, the α is tuned on the training set between 0.05 and 20 before the test.

In the experiments, the proposed investigation model is compared with the orthogonal Procrustes, MUSE(S), and MUSE(U) models. The first two models learn the mapping matrix strictly follows

the orthogonality constraint. The third model, MUSE(U), learns the mapping matrix under a soft orthogonality constraint, and get the absolutely orthogonal final mapping matrix through iterative Procrustes refinement. Therefore, these three models are all absolutely orthogonal models. Considering these models are simple, they are taken as the representations of the absolutely orthogonal models. They are compared with the proposed simple approximately orthogonal model to show the influence of the approximate orthogonality.

The results are shown in Table.4.1. The results show that the proposed investigation model always outperforms the orthogonal Procrustes. The improvement of the investigation model over orthogonal Procrustes is especially obvious on distant language pairs such as en-ru, en-zh, and zh-en. This fact confirms the theory that orthogonal mapping is not optimal on non-isomorphic embedding spaces. The OAD is also large on the en-zh and zh-en language pairs which are the most etymologically distant language pairs, supporting that more approximation in the orthogonality is needed for more non-isomorphic embedding spaces (however, the OAD is not the third and fourth largest on the en-ru and ru-en pairs, whose reason remains a question). Besides, the investigation model is also better than the MUSE(S) and the MUSE(U) models on most of the language pairs, even thought there are no extra techniques such as refinements on this investigation model. This implies the powerfulness of the investigation model, thus the powerfulness of the approximate orthogonality in BWE mapping.

Table 4.1: The BLI P@1 on MUSE(S) dataset from orthogonal mapping and approximate orthogonal mapping, the improvement of our investigation model over the orthogonal Procrustes, and the OAD of the optimized matrix from our investigation model.

Method	en-es	es-en	en-fr	fr-en	en-de	de-en	en-ru	ru-en	en-zh	zh-en
Orthogonal Procrustes	81.40	82.87	81.07	82.40	73.47	72.40	51.67	63.67	32.47	25.13
MUSE(S)	81.87	83.47	**82.13**	82.40	74.27	72.73	51.67	63.67	32.47	25.13
MUSE(U)	81.20	83.33	81.53	82.53	**74.87**	**73.47**	35.40	59.80	0.00	0.00
Investigation model	**82.13**	**84.33**	81.77	**83.50**	73.60	**73.47**	**53.93**	**64.23**	**34.00**	**37.67**
Improvement	0.73	1.46	0.70	1.10	0.13	1.07	2.26	0.56	1.53	12.54
OAD	9.76	11.32	9.59	8.13	11.24	9.26	10.55	8.76	16.2	20.26

4.2 The Approximate Orthogonality Refinement

4.2.1 The Idea of Approximate Orthogonality Refinement

In the previous section, the powerfulness of the approximate orthogonality and the investigation model is introduced. Notice the investigation model only outperforms the three relatively simple models, and the state-of-the-art model on the MUSE dataset (at the time January 2020) is still the BLISS [45]. The BLISS model applies the weak orthogonality constraint in training the mapping matrix, but the approximately orthogonal learning is followed by the iterative Procrustes which would finally produce an absolutely orthogonal mapping matrix. Since the approximately orthogonal mapping outperforms the orthogonal mapping, the approximate orthogonality refinement is proposed.

4.2.2 The Proposed Refinement

The approximate orthogonality refinement follows the standard process of the refinement. The standard refinement process supervises the new rounds of learning with the lexicons generated by the optimized mapping matrix from the previous round of learning. The new rounds of learning are implemented by the orthogonal Procrustes in iterative Procrustes refinement. Here, the new rounds of learning are implemented by the investigation model (4.1.2).

4.2.3 The Experiment Settings

In order to test the performance of the approximate orthogonality refinement, experiments are implemented. In the experiments, the BLISS model with iterative Procrustes refinements and the raw BLISS model (the model without iterative Procrustes) with approximate orthogonality refinement are compared by their performances on the MUSE dataset (on the same language pairs as in 4.1.3).

Because the BLISS model applies the RCSLS loss as the mapping target, the mapping loss $\mathcal{L}$ (Eq.4.1) of the investigation model is replaced as the same mapping loss with the RCSLS as that (Eq.3.7) in the BLISS model. The initial training loss is set as 0.0001. Because the calculation of the RCSLS loss is time costing, the maximum iteration number is reduced to 5000, and the tolerance number is also reduced to 100. In pre-experiments, it is found that the investigation model works best with the RCSLS mapping loss when the weight hyper-parameter α is 20 (over other values from 0.05 to 20), therefore the α is fixed as 20 in these experiments.

The approximate orthogonality refinement is implemented only once over BLISS, unlike the iterative Procrustes.

4.2.4 The Experiment Results

As introduced above, the comparison is designated to be between the BLISS model with iterative Procrustes refinement and the BLISS model with the approximate orthogonality refinement. However, in the reproduction of the BLISS model using the released codes[2], it is found that the iterative Procrustes does not improve the mapping performance. And the model is found to automatically take the mapping matrix before the refinement as the final one if the refinement does not give a higher score. Therefore, the comparison is proposed to be among the raw BLISS model, the BLISS with iterative Procrustes refinement, and the BLISS with the approximate orthogonality refinement.

The results are shown in Table.4.2. In our reproduction, the iterative Procrustes refinement never outperforms the original BLISS. And the results show that the approximate orthogonality refinement is always better than the iterative Procrustes refinement, and is usually better than the raw BLISS.

Based on the experiment results, the use of the approximate orthogonality refinement is suggested.

4.3 Conclustions

In this chapter, the approximate orthogonality (of the mapping matrix) in the BWE mapping is researched.

[2]https://github.com/joelmoniz/BLISS

Table 4.2: BLI P@1 by BLISS and the refinements on the MUSE dataset.

Method	en-es	es-en	en-fr	fr-en	en-de	de-en	en-ru	ru-en	en-zh	zh-en
BLISS	83.60	86.47	84.20	**84.73**	**78.33**	**76.33**	57.07	**67.07**	**48.33**	**47.93**
Iterative Procrustes Refinement	82.87	84.33	82.93	83.60	76.93	72.87	54.47	65.13	36.40	26.27
Approximate Orthogonality Refinement	**83.80**	**86.73**	**84.33**	84.60	77.27	75.60	**57.27**	66.0	36.40	45.73

The influence of the approximate orthogonality is first studied. The investigation implies different optimal OADs for different language pairs (or embedding space pairs). Then, the powerfulness of the approximate orthogonality is shown by experiments over the proposed investigation model.

Based on the fact that approximate orthogonality is powerful, the approximate orthogonality refinement is proposed. The approximate orthogonality refinement is shown to be successful by experiments on adding the refinement to the BLISS model.

As the conclusion of this chapter, the approximate orthogonality is shown to be important and powerful in the BWE mapping on the BLI task.

Chapter 5

Zero-shot RE in KBQA

5.1 Investigation of KBQA-Adapter

5.1.1 Investigation Contents

As introduced in 3.2.2, the KBQA-Adapter [1] improves zero-shot RE performance largely over the HR-BiLSTM model, bringing the unseen accuracy (the zero-shot accuracy) from 33.0% to 77.3%. This is a great work, but there still remains a gap between the seen accuracy and the unseen accuracy. In this section, the actual performance of the KBQA-Adapter model is investigated for the hints or inspirations about improving the unseen performance to be discovered.

The investigation is implemented by collecting statistical information about the test results produced by the KBQA-Adapter on the SQB dataset. The statistics are based on three levels of manmade groups. The top level divides the test results into two parts, the correct examples and the fault examples. The second level divides the results by whether the target relations and the predicted relations are seen or unseen. The third level is the specific statistical terms. There are many terms are included in the third level, for the possibility that the statistics would help discover useful information.

Before the terms of the third level, the concept of the group and the subgroup of a relation is going to be introduced. In the Freebase [35] knowledge base which is the base knowledge base in the SimpleQuestions [42] and the SQB datasets, the relation names contain different parts. For example, the name of the relation *people.person.gender* contains three parts divided by the dot: *people*, *person*, and *gender*. The relation names contain at least three parts in the Freebase. In this thesis, the first part of a relation name is seen as the group as the relation, and the combination of the first two parts is seen as the subgroup of this relation. In the example of the relation *people.person.gender*, the group is *people*, and the subgroup is *people.person*.

Then the terms of the third level groups are listed here:

- Whether the group of the target relation is seen or unseen (a seen group contains at least one seen relation in this group);
- Whether the subgroup of the target relation is seen or unseen (a seen subgroup contains at least one seen relation in this subgroup);

- Whether the group of the predicted relation is seen or unseen;
- Whether the subgroup of the predicted relation is seen or unseen;
- Whether the predicted relation and the target relation are in the same group;
- Whether the predicted relation and the target relation are in the same subgroup;
- Whether the most similar relation with the predicted relation (in the relation embedding space) is the target relation;
- Whether the most similar relation with the predicted relation is seen or unseen;
- The cosine similarity score between the most similar relation and the predicted relation;
- Whether the second most similar relation in the relation representation with the question representation (the relation representation and the question representation are got at the highest layer of the HR-BiLSTM or of the KBQA-Adapter, and this second most similar relation is thus referred as the high similar relation in the following contexts) is the target relation;
- Whether the high similar relation is the target relation;
- Whether the high similar relation is the same relation as the most similar relation;
- The relative similarity score (the similarity score denotes the cosine similarity between the relation representation and the question representation) of the high similar relation comparing to the predicted relation;
- The cosine similarity score between the high similar relation and the predicted relation.

These three levels of groups divide the test results into more than 250 items, the actual statistical results can be seen in Appendix.A.

5.1.2 Investigation Results

Three important discoveries are found based on the statistical results.

1. The KBQA-Adapter works better on subgroup seen relations than on subgroup unseen relations.

   The average accuracy on the subgroup seen relations is much higher than that on the subgroup unseen relations. The average accuracy for the subgroup seen relations is 93.58% while the average accuracy for the subgroup unseen relations is only 65.18%. Considering the seen accuracy is much higher than the unseen accuracy, the accuracy of subgroup seen but themselves unseen relations is counted, 85.12%, also much higher than the subgroup unseen accuracy.

   (Because the number of group unseen relations is less than ten, their statistical results are not considered.)

2. Wrongly predicted relations tend to be in the same group or subgroup as the target relation.

   65.45% wrong predictions are in the same group as the target relation, and 33.53% wrong predictions are in the same subgroup as the target. Considering there are 90 groups and 1966 subgroups in total, those rates show a clear tendency of the prediction.

3. Wrongly predicted relations tend to be seen relations.

   84.57% wrong predictions are seen relations. When the target relation is unseen, still 82.72% wrong predictions are seen relations.

Based on these three discoveries, it is considered that the groups and subgroups of the relations are an important aspect of the relation information to the KBQA-Adapter, and that unseen relations should be better adapted.

For applying more group and subgroup information, we design to sample more negative examples with the relations in the same group and subgroup in the training time. The experiments are not successful (resulting in lower RE seen and unseen accuracy) and the experiment details are not included in this thesis.

For improving the adaptation, the experiment details are introduced in the next section.

5.2 Improving Adapters

As the investigation results imply, there is room for improving the adapters. This section introduces the efforts to construct a new adapter.

As introduced in 3.1.1 and in 3.1.2, there are two important factors in a mapping algorithm. First, there should be a mapping loss. There are different forms of mapping losses. In the MUSE(S), the mapping loss is the Euclidean distance between the mapped embeddings and the target embeddings. In the MUSE(U), the mapping loss is the adversarial loss. In the BLISS, the mapping loss is the combination of the RCSLS loss and the adversarial loss. Second, there might be an orthogonality constraint. There are different constraints applied in the MUSE(S), the MUSE(U), and the BLISS models. These two factors are also concluded in the adapters. In the Basic Adapter, the mapping loss is the MSE loss $\mathcal{L}_{\text{mse}}$, which is essentially the square of Euclidean distance. In the Adversarial Adapter, the mapping loss is the adversarial loss $\mathcal{L}_D$. And the reconstruction loss can be seen as a certain constraint.

Because of the success of the BLISS model which combines the mapping loss and the adversarial loss together, a **Mixed Adapter** is proposed. In this adapter, the $\mathcal{L}_{\text{mse}}$ and the $\mathcal{L}_D$ are combined in the training, together with the reconstruction loss $loss_R$. Because the cosine distance as a mapping loss usually works better than the Euclidean distance, the mapping loss in the adapters as cosine distance is also tested. The RCSLS loss is also a powerful mapping loss, but it is not tested in this thesis for the time complexity. In order to align the distance metrics in the mapping loss and the reconstruction loss, the distance metric in the reconstruction loss, which is originally the square of Euclidean distance, is also tested as the cosine distance aligning the mapping loss. At last, as the conclusion in Chapter 4 claims, that the approximate orthogonality is important and powerful to the BWE mapping, the reconstruction constraint is changed to the approximate orthogonality constraint

with the loss $loss_{\text{orth}}$ and tested:

$$loss_{\text{orth}} = \sum_{r \in S \cup U} \|G(\boldsymbol{e_g})^T G(\boldsymbol{e_g}) - \boldsymbol{e_g}^T \boldsymbol{e_g}\|_2^2, \tag{5.1}$$

where S denotes the seen relations, U denotes the unseen relations, G denotes the generator of the adapter which is a mapping matrix, and $\boldsymbol{e_g}$ denotes the original embedding (or called general embedding) of the relation r.

The results are shown in Table.5.1. For Mixed Adapter, changing the distance metrics in the mapping loss and the reconstruction loss to the cosine distance improves the unseen accuracy from 76.2% to 77.5%. Replacing the reconstruction loss with the orthogonality constraint loss further raises the unseen accuracy to 78.1%. The Mixed Adapter with approximate orthogonality constraint has already outperformed the originally best adapter, the Adversarial Adapter with original reconstruction loss, in unseen and all accuracies.

Table 5.1: The RE accuracy on the SQB dataset of modified adapters. The general experiment settings are the same as those in KBQA-Adapter's paper. The → symbol denotes replacing one component with a new one. For example, "→ cos mapping loss" denotes replacing the original MSE mapping loss with the cosine mapping loss, and "→ orth constraint" denotes replacing the original reconstruction loss with the orthogonality constraint loss.

Model	Average Accuracy on SQB		
	Test-seen	Test-unseen	ALL
HR-BiLSTM	**93.3±0.6**	31.9±3.5	62.8±1.9
+ Basic-Adapter	92.8±0.6	75.9±6.9	84.4±3.2
+ reconstruction	92.8±0.6	75.9±6.2	84.4±2.8
→ cos mapping loss	90.3±0.6	**80.6±5.4**	85.5±2.5
→ cos reconstruction	91.9±0.9	79.8±5.4	**85.8±2.5**
→ orth constraint	89.8±1.1	78.8±6.7	84.3±3.4
→ cos mapping loss	91.4±0.7	80.1±6.1	**85.8±2.8**
+ Adversarial-Adapter	92.5±0.6	77.4±6.1	85.0±2.9
+ reconstruction	92.5±0.5	77.3±7.3	85.0±3.5
+ cos reconstruction	92.0±0.7	78.0±5.3	85.1±2.5
+ orth constraint	92.3±0.6	77.5±7.3	84.9±3.4
+ Mixed-Adapter	92.9±0.6	76.2±5.6	84.6±1.6
→ cos mapping loss	92.3±0.7	77.3±6.8	84.9±3.1
→ cos reconstruction	92.1±0.4	77.5±6.4	84.8±3.1
→ orth constraint	92.4±0.6	78.1±5.8	85.3±2.8

However, the best improvement is not brought by the Mixed Adapter. When the mapping loss is transformed to the cosine mapping loss in the Basic Adapter, the all accuracy is raised to 85.8%. Applying the same modification to the Basic Adapter with reconstruction loss, the unseen accuracy is raised to 80.6%; and the all accuracy also reaches 85.8% when the original reconstruction is further changed to the cosine reconstruction.

The Mixed Adapter does not outperform the Basic Adapter and the Adversarial Adapter, unlike what has happened in the BWE mapping. This fact is probably because of the differences between the BWE spaces and the relation embedding spaces. Due to those differences, different mapping algorithms should be applied.

The approximate orthogonality constraint outperforms the original reconstruction loss in all three adapters in unseen accuracy, but does not always outperform the cosine reconstruction loss. This fact shows the power of the approximate orthogonality constraint in the adapters. But the reason why sometimes different constraints on the mapping matrix perform better is still unknown.

5.3 The Effects of the Word Embedding Space

Besides the relation embedding space, the word embedding space is also an important shared embedding space for seen and unseen relations, since the relation names are with both seen and unseen relations. This section introduces two aspects of efforts to improve the quality of the word embedding space.

5.3.1 The Word Adapter

As introduced in 3.2.2, the relation embedding space is the JointNRE which is trained jointly with the word embedding space. Therefore, the original word embedding space and the relation embedding space are aligned. After the training with RE, the inner structure of the relation space is broken, and the KBQA-Adapter is proposed to keep the original structure of the relation embedding space. In this thesis, the word adapter is also proposed to adapter the word embedding space. In the experiment, the adversarial word adapter with reconstruction is tested. This word adapter shares the same model structure as the Adversarial Adapter with reconstruction loss for the relation embedding space.

However, adding the word adapter does not lead to good results on the SQB dataset. The test seen accuracy, unseen accuracy, and all accuracy are 84.3%±2.2%, 73.4%±6.0%, and 78.8%±3.0%, respectively. All the accuracies drop from the performance of the KBQA-Adapter without word adapter.

The reason for the drop in the performance might be that almost every word in the test time has been seen in the training time. Therefore, the word adapter does not help the adaptation of the word embedding space. On the contrary, the word adapter largely restricts the expressing ability of the word embedding space. The alignment of the word embedding space and the relation embedding space seems not to be important for the HR-BiLSTM model. And the restriction of the expressing ability might be the answer to the gap between the seen accuracy and the unseen accuracy of the KBQA-Adapter on the SQB dataset. It would be difficult to avoid the lack of expressing ability while adapting the unseen relations only by the adapter. Further information, possibly from the distant supervision or pre-training, would be the key to eliminate the gap of the RE model performance on seen and unseen relations.

5.3.2 Inserting Paraphrase Information to the Word Embedding Space

A problem with the RE task is the recognition of the different expressions (or surface forms) of a relation. The problem fits both seen RE and unseen RE. Although there are various surface forms of a relation, these various surface forms can be all seen as paraphrases, and as paraphrases to the relation name. Therefore, the problem especially fits the unseen RE, because in unseen RE the main reliable information is the relation embedding and the relation name. Considering there is a famous public paraphrase database called PPDB [48], if the paraphrase information from the database can be utilized, the surface form recognition would be aided.

There is a work about inserting paraphrase information into word embedding spaces proposed by Mrkšić et al. (2016) in their paper *Counter-fitting Word Vectors to Linguistic Constraints* [49]. The idea of the inserting paraphrase information is to push the embeddings of antonymous words away from each other, and to pull the embeddings of synonymous words close to each other, while trying to keep the structure of the word embedding space. The pushing is implemented through an *Antonym Repel* (AR) hinge loss:

$$\mathrm{AR}(V') = \sum_{(u,w)\in A} \tau(\delta - d(\boldsymbol{v}'_u, \boldsymbol{v}'_w)), \tag{5.2}$$

where V' denotes the embedding vector space, A denotes the paraphrase set, $\boldsymbol{v}'_u$ and $\boldsymbol{v}'_w$ are the embeddings of the paraphrase pair (u, w), $d(,)$ denotes a cosine similarity distance $d(\boldsymbol{v}_i, \boldsymbol{v}_j) = 1 - \cos(\boldsymbol{v}_i, \boldsymbol{v}_j)$, $\tau(x) = \max(0, x)$, and δ is the margin hyper-parameter. The pulling is implemented through a *Synonym Attract* (SA) loss:

$$\mathrm{SA}(V') = \sum_{(u,w)\in A} \tau(d(\boldsymbol{v}'_u, \boldsymbol{v}'_w) - \gamma), \tag{5.3}$$

where γ is another margin hyper-parameter. And the embedding vector space preservation is implemented through a VSP loss:

$$\mathrm{VSP}(V, V') = \sum_{i=1}^{N} \sum_{j\in N(i)} \tau(d(\boldsymbol{v}'_i, \boldsymbol{v}'_j) - d(\boldsymbol{v}_i, \boldsymbol{v}_j)), \tag{5.4}$$

where V denotes the original embedding vector space, and $N(i)$ denotes the ids of neighbors of the i-th word's vector in V. The three losses are summed up together with three weights k_1, k_2, and k_3:

$$C(V, V') = k_1\mathrm{AR}(V') + k_2\mathrm{SA}(V') + k_3\mathrm{VSP}(V, V'). \tag{5.5}$$

This technique is called counter-fitting. For detailed experiment settings, please refer to the paper. By counter-fitting, the authors report the change in the embedding space, that semantically closer words become closer in the embedding space. For example, the nearest neighbors of the word "east" are west, north, south, southeast, and northeast, in the original GloVe embedding space [7]; and the nearest neighbors are eastward, eastern, and easterly in the counter-fitted GloVe embedding space.

The paraphrase information from the PPDB is inserted to the JointNRE word embedding space by the introduced counter-fitting, and experiments are done with the KBQA-Adapter on the counter-fitted

JointNRE. The results are shown in Table.5.2. From the table, we can see that by counter-fitting, the unseen accuracy has been improved over the KBQA-Adapter on the original word embedding space. This experiment shows that the high quality of embedding spaces is an important factor in zero-shot learning.

Table 5.2: The RE accuracy on the SQB dataset of different word embedding spaces.

Model	Average Accuracy on SQB		
	Test-seen	Test-unseen	ALL
HR-BiLSTM	**93.3±0.6**	31.9±3.5	62.8±1.9
KBQA-Adapter	92.5±0.5	77.3±7.3	**85.0±3.5**
+ word adapter	84.3±2.2	73.4±6.0	78.8±3.0
+ counter-fitting	92.4±0.7	**77.5±6.7**	**85.0±3.0**

5.4 Investigation on the KEQA

5.4.1 The Superiority of KEQA

KEQA, as introduced in 3.2.3, is a KBQA pipeline. It is a strong KBQA pipeline, and claimed to be capable for ZSL. In the previous part of this chapter, the focus is on the RE of the KBQA pipeline. However, in one of my other work that is not included in this thesis, the KBQA pipeline is also investigated. In that investigation, it is found the KEQA pipeline works better than the HR-BiLSTM pipeline (the HR-BiLSTM is based on outer entity extraction models, and the entity extraction of BuboQA (2.4.1) is applied in our experiments) on the overall KBQA accuracy.

Besides the overall KBQA accuracy, the relation prediction model of the KEQA is better than the RE of HR-BiLSTM from some aspects. The prediction model is not giving a predicted relation in the KEQA pipeline but giving a predicted relation embedding. The predicted embedding is used for predicting the relation together with other information in the joint search part of the KEQA pipeline. However, in order to compare the RE ability, the predicted embedding is used solely for predicting the relation (by selecting the relation with the closest embedding to the predicted embedding) in the experiments introduced in the following part of this section, and we name the modified model KEQA-RE. Of course, cutting down other parts of the KEQA would harm the power of the relation prediction, but the overall power of the KEQA-RE is not the interest of this thesis. This thesis focuses on the zero-shot learning in RE, and does not hope other parts of the KBQA such entity extraction influence the RE.

The KEQA-RE is found to be more robust than HR-BiLSTM. The previous experiments of the HR-BiLSTM model and the KBQA-Adapter model are all based on the correct entities. When the entities are not provided, which means the relation candidates are not constrained to only a small fraction, the RE performance of the HR-BiLSTM drops rapidly. The RE accuracy on the SimpleQuestions dataset by HR-BiLSTM drops from 93.3% to 77.0% when the entities are not provided. In the meanwhile, the RE accuracy of the KEQA-RE model is 80.8% when the entities are not provided.

The superiority of the KEQA model motivates the interest in checking zero-shot RE performance of KEQA-RE. In the preliminary experiments, the KEQA-RE is found weak at zero-shot RE, therefore the modifications for improving the KEQA-RE are going to be introduced.

5.4.2 Improving KEQA-RE

This part is going to introduce several modifications on the KEQA-RE model.

Normalizing the relation embedding space. The relation embedding space is not normalized in the origin settings of KEQA-RE. In this modification, the relation embeddings are normalized such that they all have the L2-norm of one:

$$\boldsymbol{p}_{\text{normalized}} = \frac{\boldsymbol{p}_{\text{origin}}}{\|\boldsymbol{p}_{\text{origin}}\|_2}, \forall r \in S \cup U, \tag{5.6}$$

where $\boldsymbol{p}_{\text{origin}}$ denotes the original embedding of the relation r, and $\boldsymbol{p}_{\text{normalized}}$ denotes the normalized embedding.

Using the cosine distance as the loss function. The original loss function of the KEQA-RE is the MSE between the predicted relation embedding and the target relation embedding. Now, it is changed to the cosine distance between the predicted relation embedding and the target relation embedding. In evaluation phase, the nearest neighbor extraction is also modified to be based on the cosine distance (originally on the Euclidean distance). This idea is inspired by the superiority of the cosine distance metric over the Euclidean distance metric, as a mapping target or as an extraction metric, for the embedding mapping tasks. To speed up the calculation of the cosine distance, this modification is implemented together with normalizing the relation embedding space.

Using the CSLS metric for nearest neighbor extraction in evaluation. This modification is also inspired by the studies on the embedding mapping. To align it with the loss function, this modification is implemented together with the previous cosine modification (RCSLS is not applied as a loss function for its time complexity).

5.4.3 Experiment Settings

In order to test the zero-shot ability of the KEQA-RE model, the SQB dataset is utilized.

For comparison with the KBQA-Adapter, the relation embedding is the JointNRE* (the word embedding is the JointNRE embedding). Actually, the KBQA-Adapter uses the JointNRE relation embeddings in the previous experiments. But the JointNRE embedding works badly on the KEQA-RE model, while the performance of the JointNRE* on the KEQA-RE model is on the same level as the TransE embedding.

The relation candidates are all constrained to the relations that connect to the gold entity of each question for the models.

There is a synthesis process in the original implementation of the KEQA model from the released codes[3], where a question is synthesized from a fact in the knowledge base for each relation. The synthesis is done by some heuristics. For fairness, the synthesis process is closed in the experiments by default.

[3]https://github.com/xhuang31/KEQA_WSDM19

Other hyper-parameters such as max training epochs and the learning rate are the same as the default settings in the released codes. If you are interested in the experiment settings, you can check the released codes for details.

5.4.4 Experiment Results

The experiment results are shown in Table.5.3. The three modifications have all improved the KEQA-RE model both on seen accuracy and on unseen accuracy. The KEQA-RE model outperforms the original HR-BiLSTM model by a large margin on the unseen accuracy and all accuracy. However, the KEQA-RE is much worse than the KBQA-Adapter, even with synthesized inputs.

Table 5.3: The RE accuracy on the SQB dataset by modified KEQA-RE and HR-BiLSTM based models.

Model	Average Accuracy on SQB		
	Test-seen	Test-unseen	ALL
KEQA-RE	90.3±0.6	36.8±7.6	63.7±3.5
+ normalizing	90.8±0.6	40.0±8.0	65.6±3.5
+ cosine	90.9±0.7	40.2±8.5	65.7±3.8
+ CSLS	91.0±0.7	41.9±8.0	66.6±3.4
+ synthesis	91.2±0.6	46.8±7.4	69.2±3.3
HR-BiLSTM	**93.3±0.6**	31.9±3.5	62.8±1.9
KBQA-Adapter	92.5±0.5	**77.3±7.3**	**85.0±3.5**

Even though the KEQA-RE model outperforms the HR-BiLSTM model on the unseen accuracy and all accuracy, it fails to keep the superiority as soon as the HR-BiLSTM freeze the relation embeddings in the training time. Because the relation embedding space is not the input to the KEQA-RE, the same technique such as freezing embedding space or adding an adapter can not be applied to the KEQA model.

As a conclusion for this section, taking embedding space as input is very important and powerful.

5.5 Conclusions

In this chapter, our study on the embedding spaces in zero-shot RE is introduced.

We introduce our experiments about improving adapters, which confirms the power of the orthogonality constraint in the embedding mapping and of the cosine distance as the mapping metric.

We also introduce our experiments about improving the word embedding space, from which the importance of the word embedding quality is confirmed.

And we introduce experiments about the KEQA-RE model, which implies the importance of the embedding space as one part of the input.

Chapter 6

Conclusions

6.1 Thesis Conclusions

In this thesis, we introduce our work on ZSL in the RE of the KBQA by shared embedding spaces between seen relations and unseen relations.

The background information about the ZSL, embeddings, embedding mapping, KBQA, and RE are introduced. We propose to study two ZSL problems, the BLI and zero-shot RE in KBQA, and to push the understanding and machine performance forward in these two ZSL problems. To achieve this goal, we survey many related studies and introduce the most relevant ones in this thesis.

Embedding mapping is an important technique in ZSL with shared embedding spaces. It helps to transfer knowledge from seen items to unseen items. To further improve the embedding mapping, we investigate the origin of the embedding mapping, the BWE mapping. We notice and prove the power of approximate orthogonality in the BWE mapping, and improve the BLI task by exploiting the approximate orthogonality.

For improving zero-shot RE in KBQA, we first investigate the performance of the KBQA-Adapter [1] which is strong on zero-shot RE. Based on our investigation results, we propose to improve the adapters in the KBQA-Adapter. The improvements are based on previous studies in BWE mapping and our investigations on the BWE mapping. We second investigate the effect of the word embedding space which is also a shared embedding space for the seen and unseen relations. We find that inserting paraphrase information to the word embedding space can improve the RE performance.

As a conclusion, we improve two ZSL problems with shared embedding spaces. The BLI is improved by approximate orthogonality. Zero-shot RE in KBQA is improved by better embedding mapping and higher quality of the word embedding space.

6.2 Future Work

Besides the work in this thesis, there are still many things to improve in this field of research.

As we have seen in the results of BWE mapping, the existent methods are still weak for etymologically distant language pairs on the BLI task. The space for improving the BWE mapping, whether

on etymologically distant language pairs or other language pairs, is still large. There are now many studies on consistently improving the BWE mapping, not only in mapping algorithms but also on the mapping of different embedding formats (such as the contextualized word embeddings). Improvements from the BWE mapping would also benefit the research of zero-shot RE as practiced in this thesis.

The cosine reconstruction loss and the approximate orthogonality constraint are proved to improve the performance of the adapters. But the reason why these two constraints on the mapping matrix are good at different situations is not explored. The theory of the embedding mapping for adapters is insufficient. More theory work for adapters shall be done as part of the future work.

We have shown that the quality of shared embedding spaces is important for ZSL, not only by experiments of adapters but also by the experiment of inserting paraphrase information. There are also other ways to improve the embedding space quality, such as using larger contextualized word embeddings like BERT. Consistently increasing embedding spcae qualities would be an intereasting and helpful topic for ZSL.

The group and subgroup information of the relations are investigated in 5.1.1 to be important for the RE. But how to utilize that information is not successfully discovered in this thesis. Wise ways, to exploit not only the group and subgroup information but also other undiscovered useful information of the relations, should be explored to aid both seen RE and zero-shot RE.

The KEQA-RE is a good RE model, but it cannot be improved on zero-shot RE by adapters for not taking the relation embedding space as input. Better RE model structures can be proposed based on the inspiration of HR-BiLSTM and KEQA-RE models. We have tested the BERT model for zero-shot RE, but the results are not better, which implies that simply inreasing the model size is not helpful.

In this thesis, we focus on the zero-shot RE but not the whole KBQA pipeline. In fact, increasing gold entity based RE accuracy cannot result in distinct improvement on the KBQA accuracy, based on our observations on the BuboQA pipeline. The current pipelines (both the BuboQA pipeline and KEQA pipeline) rely much on the entity extraction results. It is not very related to the ZSL, but maybe better KBQA pipeline can be proposed.

The ZSL tasks of BLI and RE in KBQA are introduced where the shared embedding spaces for the seen and unseen classes are rich. There are other tasks where there exist few shared embedding spaces. For example, there are general RE tasks where the relation embedding space can not be trained for there are no knowledge bases to train it. Improving the ZSL in those tasks is also important and remains unsolved.

Bibliography

[1] Peng Wu, Shujian Huang, Rongxiang Weng, Zaixiang Zheng, Jianbing Zhang, Xiaohui Yan, and Jiajun Chen. Learning representation mapping for relation detection in knowledge base question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6130–6139, Florence, Italy, July 2019. Association for Computational Linguistics.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

[3] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, Vol. 57, No. 10, pp. 78–85, 2014.

[4] Wei Wang, Vincent W Zheng, Han Yu, and Chunyan Miao. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, Vol. 10, No. 2, pp. 1–37, 2019.

[5] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, 2008.

[6] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, 2013.

[7] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

[8] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, Vol. 5, pp. 135–146, 2017.

[9] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proc. of NAACL*, 2018.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, Vol. abs/1810.04805, , 2018.

[11] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[12] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[13] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pp. 5754–5764, 2019.

[14] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[15] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

[16] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[17] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: A pretrained language model for scientific text. 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. 3606–3611, 2019.

[18] Matthew E. Peters, Mark Neumann, Robert L Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. Knowledge enhanced contextual word representations. In *EMNLP*, 2019.

[19] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pp. 19–27, 2015.

[20] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pp. 2787–2795. Curran Associates, Inc., 2013.

[21] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes, 2014.

[22] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion, 2015.

[23] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 687–696, Beijing, China, July 2015. Association for Computational Linguistics.

[24] Xu Han, Zhiyuan Liu, and Maosong Sun. Neural knowledge acquisition via mutual attention between knowledge graph and text, 2018.

[25] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting Similarities among Languages for Machine Translation. *CoRR*, Vol. abs/1309.4, , 2013.

[26] Samuel L Smith, David H P Turban, Steven Hamblin, and Nils Y Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *5th International Conference on Learning Representations (ICLR 2017)*, 2017.

[27] Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation. In *Proceedings of the 2015 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1006–1011, Denver, Colorado, 2015. Association for Computational Linguistics.

[28] Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. Loss in Translation: Learning Bilingual Word Mapping with a Retrieval Criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2979–2984, Brussels, Belgium, 2018. Association for Computational Linguistics.

[29] Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1959–1970, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[30] Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. Earth mover's distance minimization for unsupervised bilingual lexicon induction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1934–1945, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[31] Ndapa Nakashole. {NORMA}: Neighborhood Sensitive Maps for Multilingual Word Embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 512–522, Brussels, Belgium, 2018. Association for Computational Linguistics.

[32] Masud Moshtaghi. Supervised and Nonlinear Alignment of Two Embedding Spaces for Dictionary Induction in Low Resourced Languages. 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. 823–832, Hong Kong, China, 2019. Association for Computational Linguistics.

[33] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *Proceedings of ICLR*, 2018.

[34] Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. Improving zero-shot learning by mitigating the hubness problem, 2014.

[35] Kurt Bollacker, Patrick Tufts, Tomi Pierce, and Robert Cook. A platform for scalable, collaborative, structured information integration. Intl. Workshop on Information Integration on the Web (IIWeb' 07), pp. 22–27, 2007.

[36] Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schütze. Simple question answering by attentive convolutional neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 1746–1756, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[37] Salman Mohammed, Peng Shi, and Jimmy Lin. Strong baselines for simple question answering over knowledge graphs with and without neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 291–296, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[38] Mohnish Dubey, Debayan Banerjee, Debanjan Chaudhuri, and Jens Lehmann. EARL: joint entity and relation linking for question answering over knowledge graphs. In *International Semantic Web Conference*, pp. 108–126. Springer, 2018.

[39] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. Knowledge Graph Embedding Based Question Answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, pp. 105–113, New York, NY, USA, 2019. ACM.

[40] Jenny Rose Finkel, Trond Grenager, and Christopher D Manning. Incorporating non-local information into information extraction systems by gibbs sampling. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL' 05), pp. 363–370, 2005.

[41] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[42] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*, 2015.

[43] Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, Vol. 43, No. 3, pp. 209–226, 2009.

[44] Anders Søgaard, Sebastian Ruder, and Ivan Vulić. On the Limitations of Unsupervised Bilingual Dictionary Induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 778–788, Melbourne, Australia, 2018. Association for Computational Linguistics.

[45] Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R Gormley, and Graham Neubig. Bilingual Lexicon Induction with Semi-supervision in Non-Isometric Embedding Spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 184–193, Florence, Italy, 2019. Association for Computational Linguistics.

[46] Mo Yu, Wenpeng Yin, Kazi Saidul Hasan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. Improved Neural Relation Detection for Knowledge Base Question Answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 571–581, Vancouver, Canada, 2017. Association for Computational Linguistics.

[47] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[48] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 758–764, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[49] Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. Counter-fitting Word Vectors to Linguistic Constraints. In *Proceedings of the 2016 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–148, San Diego, California, 2016. Association for Computational Linguistics.

Published Paper

The related publication with this thesis is listed as follows.

- 鄧一凡, 呂照陽, 鶴岡慶雅. On Approximately Orthogonal Matrices in Bilingual Word Embedding Mapping, 言語処理学会第 26 回年次大会 (NLP2020), 2020 年 3 月.

Appendices

Appendix A

KBQA-Adapter investigation Results

Table A.1: The investigation results on the KBQA-Adapter. Refer to 5.1.1 for detailed introduction about each check items.

group1	group2	group3	number1 / mean1	rate1 / std1	number2 / mean2	rate2 / std2	mean all	std all
correct / fault			18057	86.28	2872	13.72		
correct	target seen, unseen		9974	55.24	8083	44.76		
correct	target unseen	target group seen / unseen	8022	99.25	61	0.75		
correct	target unseen	target sub-group seen / unseen	6445	79.74	1638	20.26		
correct	target unseen	similar seen / unseen	3536	43.75	4547	56.25		
correct	target unseen	similar score	-0.301	0.112	-0.320	0.119	-0.312	0.116
correct	target unseen	similar high seen / unseen	6392	79.08	1691	20.92		
correct	target unseen	similar high score	-0.230	0.108	-0.278	0.285	-0.240	0.163
correct	target unseen	similar = / != similar high	2128	26.33	5955	73.67		
correct	target unseen	similarity score					0.36	0.51
correct	target seen	target group seen / unseen	9974	100.00	0	0.00		
correct	target seen	target sub-group seen / unseen	9974	100.00	0	0.00		
correct	target seen	similar seen / unseen	9234	92.58	740	7.42		
correct	target seen	similar score	-0.285	0.114	-0.334	0.122	-0.288	0.116
correct	target seen	similar high seen / unseen	9011	90.34	963	9.66		
correct	target seen	similar high score	-0.267	0.188	-0.208	0.105	-0.261	0.183
correct	target seen	similar = / != similar high	4927	49.40	5047	50.60		

Table A.1: The investigation results on the KBQA-Adapter. *(continued)*

group1	group2	group3	number1 / mean1	rate1 / std1	number2 / mean2	rate2 / std2	mean all	std all
correct	target seen	similarity score					0.55	0.48
fault	target seen, predict seen / unseen		778	27.09	92	3.20		
fault	target unseen, predict seen / unseen		1656	57.66	346	12.05		
fault	target seen, predict seen	group same / different	492	63.24	286	36.76		
fault	target seen, predict seen	subgroup same / different	368	47.30	410	52.70		
fault	target seen, predict seen	similar = / != target	409	52.57	369	47.43		
fault	target seen, predict seen	similar seen / unseen	725	93.19	53	6.81		
fault	target seen, predict seen	similar score	-0.159	0.127	-0.262	0.128	-0.166	0.130
fault	target seen, predict seen	similar high = / != target	622	79.95	156	20.05		
fault	target seen, predict seen	similar high seen / unseen	753	96.79	25	3.21		
fault	target seen, predict seen	similar high score	-0.076	0.063	-0.070	0.046	-0.076	0.063
fault	target seen, predict seen	similar = / != similar high	378	48.59	400	51.41		
fault	target seen, predict seen	similarity score					0.53	0.50
fault	target seen, predict unseen	group same / different	49	53.26	43	46.74		
fault	target seen, predict unseen	subgroup same / different	28	30.43	64	69.57		
fault	target seen, predict unseen	predict group seen / unseen	91	98.91	1	1.09		
fault	target seen, predict unseen	predict subgroup seen / unseen	61	66.30	31	33.70		
fault	target seen, predict unseen	similar = / != target	6	6.52	86	93.48		
fault	target seen, predict unseen	similar seen / unseen	19	20.65	73	79.35		
fault	target seen, predict unseen	similar score	-0.152	0.100	-0.228	0.123	-0.212	0.123
fault	target seen, predict unseen	similar high = / != target	61	66.30	31	33.70		
fault	target seen, predict unseen	similar high seen / unseen	83	90.22	9	9.78		

Table A.1: The investigation results on the KBQA-Adapter. *(continued)*

group1	group2	group3	number1 / mean1	rate1 / std1	number2 / mean2	rate2 / std2	mean all	std all
fault	target seen, predict unseen	similar high score	-0.056	0.046	-0.057	0.060	-0.056	0.047
fault	target seen, predict unseen	similar = / != similar high	12	13.04	80	86.96		
fault	target seen, predict unseen	similarity score					0.19	0.44
fault	target unseen, predict seen	group same / different	1081	65.28	575	34.72		
fault	target unseen, predict seen	subgroup same / different	483	29.17	1173	70.83		
fault	target unseen, predict seen	target group seen / unseen	1656	100.00	0	0.00		
fault	target unseen, predict seen	target subgroup seen / unseen	1021	61.65	635	38.35		
fault	target unseen, predict seen	similar = / != target	372	6.52	1284	93.48		
fault	target unseen, predict seen	similar seen / unseen	1190	71.86	466	28.14		
fault	target unseen, predict seen	similar score	-0.232	0.117	-0.119	0.090	-0.200	0.121
fault	target unseen, predict seen	similar high = / != target	951	57.43	705	42.57		
fault	target unseen, predict seen	similar high seen / unseen	489	29.53	1167	70.47		
fault	target unseen, predict seen	similar high score	-0.077	0.068	-0.097	0.074	-0.091	0.073
fault	target unseen, predict seen	similar = / != similar high	463	27.96	1193	72.04		
fault	target unseen, predict seen	similarity score					0.28	0.51
fault	target unseen, predict unseen	group same / different	229	66.18	117	33.82		
fault	target unseen, predict unseen	subgroup same / different	84	24.28	262	75.72		
fault	target unseen, predict unseen	target group seen / unseen	346	100.00	0	0.00		
fault	target unseen, predict unseen	target subgroup seen / unseen	106	30.64	240	69.36		
fault	target unseen, predict unseen	predict group seen / unseen	346	100.00	0	0.00		
fault	target unseen, predict unseen	predict subgroup seen / unseen	98	28.32	248	71.68		
fault	target unseen, predict unseen	similar = / != target	227	65.61	119	34.39		

Table A.1: The investigation results on the KBQA-Adapter. *(continued)*

group1	group2	group3	number1 / mean1	rate1 / std1	number2 / mean2	rate2 / std2	mean all	std all
fault	target unseen, predict unseen	similar seen / unseen	0	0.00	346	100.00		
fault	target unseen, predict unseen	similar score	nan	nan	-0.073	0.072	-0.073	0.072
fault	target unseen, predict unseen	similar high = / != target	177	51.16	169	48.84		
fault	target unseen, predict unseen	similar high seen / unseen	85	24.57	261	75.43		
fault	target unseen, predict unseen	similar high score	-0.042	0.036	-0.038	0.033	-0.039	0.034
fault	target unseen, predict unseen	similar = / != similar high	207	59.83	139	40.17		
fault	target unseen, predict unseen	similarity score					0.73	0.44