Learning Representation Mapping for Relation Detection in Knowledge Base Question Answering

Peng Wu[1,2], **Shujian Huang**[1,2] , **Rongxiang Weng**[1,2], **Zaixiang Zheng**[1,2],
Jianbing Zhang[1,2], **Xiaohui Yan**[3], **Jiajun Chen**[1,2]
[1]National Key Laboratory for Novel Software Technology, Nanjing, China
[2]Nanjing University, Nanjing, China
[3]Poisson Lab, Huawei Technologies, Beijing, China
{wup, wengrx, zhengzx}@nlp.nju.edu.cn
{huangsj, zjb, chenjj}@nju.edu.cn
yanxiaohui2@huawei.com

Abstract

Relation detection is a core step in many natural language process applications including knowledge base question answering. Previous efforts show that single-fact questions could be answered with high accuracy. However, one critical problem is that current approaches only get high accuracy for questions whose relations have been seen in the training data. But for unseen relations, the performance will drop rapidly. The main reason for this problem is that the representations for unseen relations are missing. In this paper, we propose a simple mapping method, named representation adapter, to learn the representation mapping for both seen and unseen relations based on previously learned relation embedding. We employ the adversarial objective and the reconstruction objective to improve the mapping performance. We re-organize the popular SimpleQuestion dataset to reveal and evaluate the problem of detecting unseen relations. Experiments show that our method can greatly improve the performance of unseen relations while the performance for those seen part is kept comparable to the state-of-the-art.[1]

1 Introduction

The task of Knowledge Base Question Answering (KBQA) has been well developed in recent years (Berant et al., 2013; Bordes et al., 2014; Yao and Van Durme, 2014). It answers questions using an open-domain knowledge base, such as Freebase (Bollacker et al., 2008), DBpedia (Lehmann et al., 2015) or NELL (Carlson et al., 2010). The knowledge base usually contains a large set of triples. Each triple is in the form of ⟨*subject, relation, object*⟩, indicating the *relation* between the *subject entity* and the *object entity*.

Typical KBQA systems (Yao and Van Durme, 2014; Yin et al., 2016; Dai et al., 2016; Yu et al., 2017; Hao et al., 2018) can be divided into two steps: the entity linking step first identifies the target entity of the question, which corresponds to the subject of the triple; the relation detection step then determines the relation that the question asks from a set of candidate relations. After the two steps, the answer could be obtained by extracting the corresponding triple from the knowledge base (as shown in Figure 1).

Our main focus in this paper is the relation detection step, which is more challenging because it needs to consider the meaning of the whole question sentence (e.g., the pattern "where was ... born"), as well as the meaning of the candidate relation (e.g., "place_of_birth"). For comparison, the entity linking step benefits more from the matching of surface forms between the words in the question and subject entity (e.g., "Mark Mifsud").

In recent deep learning based relation detection approaches, each word or relation is represented by a dense vector representation, called *embedding*, which is usually learned automatically while optimizing the relation detection objective. Then, the inference processes of these approaches are executed by neural network computations. Such approaches enjoy great success in common KBQA datasets, such as SimpleQuestion (Bordes et al., 2015), achieving over 90% accuracy in relation detection. In the words of Petrochuk and Zettlemoyer (2018), "SimpleQuestion is nearly solved." However, we notice that in the common split of

[1]Our code and data are available at `https://github.com/wudapeng268/KBQA-Adapter`.

Question:	where was **Mark Mifsud** born?
Candidate Relations:	**people.person.place_of_birth**
	people.person.nationality
	people.person.profession
	...
Triple:	<Mark Mifsud, people.person.place_of_birth, Malta>

Figure 1: A KBQA example. The bold words in the question are the target entity, identified in the entity linking step. The relation detection step selects the correct relation (marked with bold font) from a set of candidate relations. The answer of this question is the object entity of the triple extracted from the knowledge base.

the SimpleQuestion dataset, 99% of the relations in the test set also exist in the training data, which means their embeddings could be learned well during training. On the contrary, for those relations which are never seen in the training data (called *unseen relations*), their embeddings have never been trained since initialization. As a result, the corresponding detection performance could be arbitrary, which is a problem that has not been carefully studied.

We emphasize that the detection for these unseen relations is critical because it is infeasible to build training data for all the relations in a large-scale knowledge base. For example, SimpleQuestion is a large-scale human annotated dataset, which contains 108,442 natural language questions for 1,837 relations sampled from FB2M (Bordes et al., 2015). FB2M is a subset of FreeBase (Bollacker et al., 2008) which have 2 million entities, 6,700 relations. A large portion of these relations can not be covered by the human-annotated dataset such as SimpleQuestion. Therefore, for building up a practical KBQA system that could answer questions based on FB2M or other large-scale knowledge bases, dealing with the unseen relations is very important and challenging. This problem could be considered as a zero-shot learning problem (Palatucci et al., 2009) where the labels for test instances are unseen in the training dataset.

In this paper, we present a detailed study on this zero-shot relation detection problem. Our contributions could be summarized as follows:

1. Instead of learning the relation representation barely from the training data, we employ methods to learn the representations from the whole knowledge graph which has much wider coverage.

2. We propose a mapping mechanism, called *representation adapter*, or simply *adapter*, to incorporate the learned representations into the relation detection model. We start with the simple mean square error loss for the non-trivial training of the adapter and propose to incorporate adversarial and reconstruction objectives to improve the training process.

3. We re-organize the SimpleQuestion dataset as SimpleQuestion-Balance to evaluate the performance for seen and unseen relations, separately.

4. We present experiments showing that our proposed method brings a great improvement to the detection of unseen relations, while still keep comparable to the state-of-the-art method for the seen relations.

2 Representation Adapter

2.1 Motivation

Representation learning of human annotated data is limited by the size and coverage of the training data. In our case, because the unseen relations and their corresponding questions do not occur in the training data, their representations cannot be properly trained, leading to poor detection performance. A possible solution for this problem is to employ a large number of unannotated data, which may be much easier to obtain, to provide better coverage.

Usually, pre-trained representations are not directly applicable to specific tasks. One popular way to utilize these representations is using them as initialization. These initialized representations are then fine-tuned on the labeled training data, with a task specific objective. However, with the above mentioned coverage issues, the representations of unseen relations will not be updated properly during fine-tuning, leading to poor test performance.

To solve this problem, we keep the representation unchanged during training, and propose a *representation adapter* to bridge the gap between general purposed representations and task specific ones. We will then present the basic adapter framework, introduce the adversarial adapter and the reconstruction objective as enhancements.

Throughout this paper, we use the following notations: let r denote a single relation; S and U denote the set of seen and unseen relations, respectively; $e(r)$ or e denote the embedding of r; specifically, we use e_{g} to denote the general pre-trained

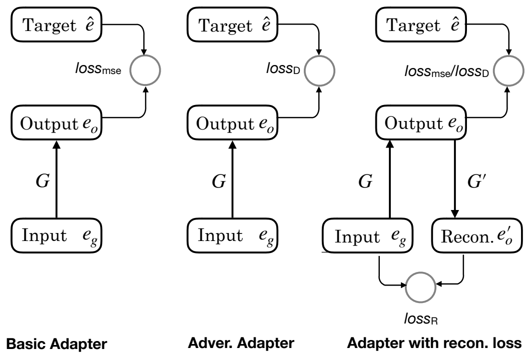

Figure 2: The structures of representation adapter. On the left is the basic adapter; on the middle is the adversarial adapter; on the right is the adapter with the reconstruction loss. Adver. and recon. are the abbreviation of adversarial and reconstruction, respectively.

embedding.

2.2 Basic Adapter

Pseudo Target Representations The basic idea is to use a neural network representation adapter to perform the mapping from the general purposed representation to the task specific one. The input of the adapter is the embedding learned from the knowledge base. However, the output of the adapter is undecided, because there is no oracle representation for the relation detection task. Therefore, we first train a traditional relation detection model similar to Yu et al. (2017). During training, the representations for relations in the training set (seen relations) will be updated for the relation detection task. We use these representations as *pseudo target representations*, denoted as $\hat{e}$, for training the adapter.

Linear Mapping Inspired by Mikolov et al. (2013), which shows the representation space of similar languages can be transferred by a linear mapping, we also employ a linear mapping function $G(\cdot)$ to map the general embedding e_g to the task specific (pseudo target) representation $\hat{e}$ (Figure 2, left).

The major difference between our adapter and an extra layer of neural network is that specific losses are designed to train the adapter, instead of implicitly learning the adapter as a part of the whole network. We train the adapter to optimize the following objective function on the seen relations:

$$\mathcal{L}_{\text{adapter}} = \sum_{r \in S} loss(\hat{e}, G(e_{\text{g}})). \tag{1}$$

Here the loss function could be any metric that evaluates the difference between the two representations. The most common and simple one is the mean square error loss (Equation (2)), which we employ in our basic adapter. We will discuss other possibilities in the following sub-sections.

$$loss_{\text{MSE}}(\hat{e}, G(e_{\text{g}})) = ||\hat{e} - G(e_{\text{g}})||_2^2 \tag{2}$$

2.3 Adversarial Adapter

The mean square error loss only measures the absolute distance between two embedding vectors. Inspired by the popular generative adversarial networks (GAN) (Goodfellow et al., 2014; Arjovsky et al., 2017) and some previous works in unsupervised machine translation (Conneau et al., 2018; Zhang et al., 2017a,b), we use a discriminator to provide an adversarial loss to guide the training (Figure 2, middle). It is a different way to minimize the difference between $G(e)$ and $\hat{e}$.

In detail, we train a discriminator, $D(\cdot)$, to discriminate the "real" representation, i.e., the fine-tuned relation embedding $\hat{e}$, from the "fake" representation, which is the output of the adapter. The adapter $G(\cdot)$ is acting as the generator in GAN, which tries to generate a representation that is similar to the "real" representation. We use WassersteinGAN (Arjovsky et al., 2017) to train our adapter. For any relations sampled from the training set, the objective function for the discriminator $loss_{\text{D}}$ and generator $loss_{\text{G}}$ are:

$$loss_{\text{D}} = \mathbb{E}_{r \in S}[D(G(e_{\text{g}}))] - \mathbb{E}_{r \in S}[D(\hat{e})] \tag{3}$$

$$loss_{\text{G}} = -\mathbb{E}_{r \in S}[D(G(e_{\text{g}}))] \tag{4}$$

Here for $D(\cdot)$, we use a feed forward neural network without the sigmoid function of the last layer (Arjovsky et al., 2017).

2.4 Reconstruction Loss

The adapter could only learn the mapping by using the representations of seen relations, which neglects the potential large set of unseen relations. Here we propose to use an additional reconstruction loss to augment the adapter (Figure 2, right). More specifically, we employ a reversed adapter $G'(\cdot)$, mapping the representation $G(e)$ back to e.

The advantage of introducing the reversed training is two-fold. On the one hand, the reversed adapter could be trained with the representation for all the relations, both seen and unseen ones. On the other hand, the reversed mapping could also serve as an extra constraint for regularizing the forward mapping.

For the reversed adapter $G'(\cdot)$, We simply use a similar linear mapping function as for $G(\cdot)$, and train it with the mean square error loss:

$$loss_{\mathrm{R}} = \sum_{r \in S \cup U} ||G'(G(e_{\mathrm{g}})) - e_{\mathrm{g}}||_2^2 \qquad (5)$$

Please note that, different from previous loss functions, this reconstruction loss is defined for both seen and unseen relations.

3 Relation Detection with the Adapter

We integrate our adapter into the state-of-the-art relation detection framework (Yu et al., 2017, Hierarchical Residual BiLSTM (HR-BiLSTM)).

Framework The framework uses a question network to encode the question sentence as a vector $\mathbf{q}_{\mathrm{f}}$ and a relation network to encode the relation as a vector $\mathbf{r}_{\mathrm{f}}$. Both of the two networks are based on the Bi-LSTM with max-pooling operation. Then, the cosine similarity is introduced to compute the distance between the $\mathbf{q}_{\mathrm{f}}$ and $\mathbf{r}_{\mathrm{f}}$, which determines the detection result. Our adapter is an additional module which is used in the relation network to enhance this framework (Figure 3).

Adapting the Relation Representation The relation network proposed in Yu et al. (2017) has two parts for relation representations: one is at word-level and the other is at relation-level. The two parts are fed into the relation network to generate the final relation representation.

Different from previous approaches, we employ the proposed adapter $G(\cdot)$ on the relation-level representations to solve unseen relation detection problem. There are several approaches to obtain the relation representations from the knowledge base into a universal space (Bordes et al., 2013; Wang et al., 2014; Han et al., 2018). In practice, we use the JointNRE embedding (Han et al., 2018), because its word and relation representations are in the same space.

Training Following Yu et al. (2017), the relation detection model is trained by the hinge loss (Bengio et al., 2003) which tries to separate the score of each negative relation from the positive relation

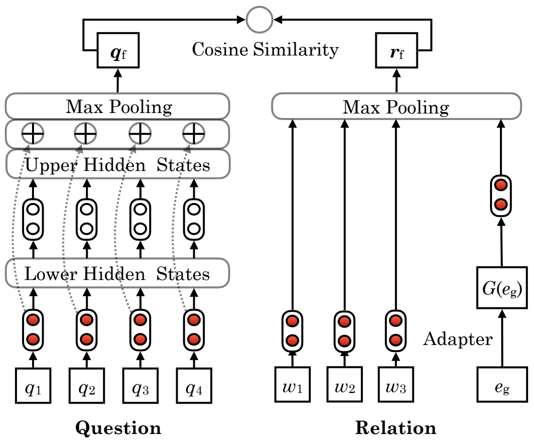

Figure 3: KBQA baseline with the adapter. Shared Bi-LSTM is marked with the same color. The adapter maps task independent representations for each relation to the task specific ones, which are fed into the relation network.

by a margin:

$$\mathcal{L}_{\mathrm{rd}} = \sum \max(0, \gamma - s(\mathbf{q}_{\mathrm{f}}, \mathbf{r}_{\mathrm{f}}^{+}) + s(\mathbf{q}_{\mathrm{f}}, \mathbf{r}_{\mathrm{f}}^{-})), \quad (6)$$

where γ is the margin; $\mathbf{r}_{\mathrm{f}}^{+}$ is the positive relation from the annotated training data; $\mathbf{r}_{\mathrm{f}}^{-}$ is the relation negative sampled from the rest relations; $s(\cdot, \cdot)$ is the cosine distance between $\mathbf{q}_{\mathrm{f}}$ and $\mathbf{r}_{\mathrm{f}}$.

The basic relation detection model is pre-trained to get the pseudo target representations. Then, the adapter is incorporated into the training process, and jointly optimized with the relation detection model. For the adversarial adapter, the generator and the discriminator are trained alternatively following the common practice.

4 SimpleQuestion-Balance (SQB)

As mentioned before, SimpleQuestion (SQ) is a large-scale KBQA dataset. Each sample in SQ includes a human annotated question and the corresponding knowledge triple. However, the distribution of the relations in the test set is unbalanced. Most of the relations in the test set have been seen in the training data. To better evaluate the performance of unseen relation detection, we re-organize the SQ dataset to balance the number of seen and unseen relations in development and test sets, and the new dataset is denoted as *SimpleQuestion-Balance (SQB)*.

The re-organization is performed by randomly shuffle and split into 5 sets, i.e. Train, Dev-seen, Den-unseen, Test-seen and Test-unseen, while

Datasets	SQ	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**

Table 1: The number of instances in each subset from SimpleQuestion (SQ) and SimpleQuestion-Balance (SQB) datasets. Dev-seen and Dev-unseen are seen and unseen part of development set; Test-seen and Test-unseen are seen and unseen part of test set, respectively.

checking the overlapping of relations and the percentage of seen/unseen samples in each set. We require the sizes of the training, development and test sets are similar to SQ.

The details of the resulting SQB and SQ are shown in Table 1. The SQ dataset only have 0.65% (71 / 10845) and 0.74% (161 / 21687) of the unseen samples in the dev set (Dev-unseen) and test set (Test-unseen), respectively.

5 Experiment

5.1 Settings

Implementation Details We use RMProp (Tieleman and Hinton, 2012) as the optimization strategy to train the proposed adapter. The learning rate is set as 10^{-4}. We set the batch size as 256. Following Arjovsky et al. (2017), we clip the parameters of discriminator into $[-c, c]$, where c is 0.1. Dropout rate is set as 0.2 to regularize the adapter.

The baseline relation detection model is almost same as Yu et al. (2017), except that the word embedding and relation embedding of our model are pre-trained by JointNRE (Han et al., 2018) on FB2M and Wikipedia , with the default settings reported in the Han et al. (2018). The embeddings are fine-tuned with the model.

More specifically, the dimension of relation representation is 300. The dimension for the hidden state of Bi-LSTM is set to 256. Parameters in the neural models are initialized using a uniform sampling. The number of negative sampled relations is 256. The γ in hinge loss (Equation (6)) is set to 0.1.

Evaluation To evaluate the performance of relation detection, we assume that the results of entity linking are correct. Two metrics are employed. *Micro average accuracy* (Tsoumakas et al., 2010) is the average accuracy of all samples, which is the metric used in previous work. *Macro average accuracy* (Sebastiani, 2002; Manning et al., 2008; Tsoumakas et al., 2010) is the average accuracy of the relations.

Please note that because different relations may correspond to the different number of samples in the test set, the micro average accuracy may be affected by the distribution of unseen relations in the test set. In this case, the macro average accuracy will serve as an alternative indicator.

We report the average and standard deviation (std) of 10-folds cross validation to avoid contingency.

5.2 Main Results

Main results for baseline and the proposed model with the different settings are listed in Table 2. The detailed comparison is as follows:

Baseline The baseline HR-BiLSTM (line 1) shows the best performance on Test-seen, but the performance is much worse on Test-unseen. For comparison, training the model without fine-tuning (line 2) achieves much better results on Test-unseen, demonstrating our motivation that the embeddings are the reason for the weak performance on unseen relations, and fine-tuning makes them worse.

Using Adapters Line 3 shows the results of adding an extra mapping layer of neural networks between the pretrained embedding and the relation detection networks, without any loss. Although ideally, it is possible to learn the mapping implicitly with the training, in practice, this does not lead to a better result (line 3 v.s. line 2).

While keeping similar performance on the Test-seen with the HR-BiLSTM, all the models using the representation adapter achieve great improvement on the Test-unseen set. With the simplest form of adapter (line 4), the accuracy on Test-unseen improves to 76.0% / 69.5%. It shows that our model can predict unseen relation with better accuracy.

Using adversarial adapter (line 6) can further improve the performance on the Test-unseen in both micro and macro average scores.

Using Reconstruction Loss Adding reconstruction loss to basic adapter can also improve the performance (line 5 v.s. line 4) slightly. The similar improvement is obtained for the adversarial

#	Model	Micro / Macro Average Accuracy on SQB (%)		
		Test-seen	Test-unseen	All
1	HR-BiLSTM	**93.5±0.6** / 84.7±1.4	33.0±5.7 / 49.3±1.7	63.3±3.6 / 71.2±1.3
2	+ no fine-tune	93.4±0.7 / 83.8±0.7	57.8±9.8 / 60.8±2.0	75.6±5.0 / 75.0±0.6
3	+ no fine-tune + mapping	93.3±0.7 / 84.0±1.6	52.0±7.2 / 60.6±2.1	72.7±3.8 / 75.1±1.3
4	+ Basic-Adapter	92.8±0.7 / 84.1±1.2	76.0±7.5[†] / 69.5±2.0[†]	84.5±3.5 / 78.5±1.3
5	+ reconstruction	93.0±0.5 / 84.4±0.8	76.1±7.0[†] / 70.7±1.8[†]	84.6±3.3 / 79.2±0.8
6	+ Adversarial-Adapter	92.6±0.9 / **86.4±1.4**	77.1±7.1[†] / **73.2±2.1**[†]	**84.9±3.2** / **81.4±1.4**
7	+ reconstruction [Final]	92.4±0.8 / 86.1±0.7	**77.3±7.6**[†] / 73.0±1.7[†]	**84.9±3.5** / 81.1±0.8

Table 2: The micro average accuracy and macro average accuracy of relation detection on the SQB dataset. "†" indicates statistically significant difference ($p < 0.01$) from the HR-BiLSTM.

adapter in micro average accuracy (line 7 v.s. line 6).

Finally, using all the techniques together (line 7) gets the score of 77.3% / 73.0% on Test-unseen, and 84.9% / 81.1% on the union of Test-seen and Test-unseen in micro/macro average accuracy, respectively. We mainly use this model as our final model for further comparison and analysis.

We notice that the results of our model on Test-seen are slightly lower than that of HR-BiLSTM. It is because we use the mapped representations for the seen relations instead of the directly fine-tuned representations. This dropping is negligible compared with the improvement in the unseen relations.

Integration to the KBQA To confirm the influence of unseen relation detection for the entire KBQA, we integrate our relation detection model into a prototype KBQA framework. During the entity linking step, we use FocusPrune (Dai et al., 2016) to get the mention of questions. Then, the candidate mentions are linked to the entities in the knowledge base. Because the FreeBase API was deprecated [2], we restrict the entity linking to an exact match for simplicity. The candidate relations are the set of relations linked with candidate subjects. We evaluate the KBQA results using the micro average accuracy introduced in Bordes et al. (2015), which considers the prediction as correct if both the subject and relation are correct.

As shown in Table 3, the proposed adapter method can improve KBQA from 48.5% to 63.7%. Comparing with the result of relation detection, we find that the boost of relation detection could indeed lead to the improvement of a KBQA system.

[2]https://developers.google.com/freebase/

Model	Accuracy (%)
HR-BiLSTM	48.5±3.3
+ no fine-tune	56.4±3.4
Final	**63.7±3.2**

Table 3: The micro average accuracy of the whole KBQA system with different relation detection models.

Model	Seen Rate ↓ (%)
HR-BiLSTM	47.2±2.0
+ no fine-tune	34.8±2.3
Final	**21.2±1.7**

Table 4: Seen relation prediction rate in the Test-unseen set. We calculate the macro average of this rate.

5.3 Analysis

Seen Relation Bias We use macro-average to calculate the percentage of instances whose relations are wrongly predicted to be a seen relation on Test-unseen. We call this indicator the *seen rate*, the lower the better. Because the seen relations are better learned after fine-tuning, while the representations for unseen relations are not updated well. So the relation detection model may have a strong trend to select those seen relations as the answer. The result in Table 4 shows that our adapter makes the trend of choosing seen relation weaker, which helps to promote a fair choice between seen and unseen relations.

Influence of Number of Relations for Training We discuss the influence of the number of relations in the training set for our adapter. Our adapter are trained mainly by the seen relations, because we can get pseudo target representation for these relations. In this experiment, we random sample 60,000 samples from the training set to perform the training, and plot the accuracy against the different number of relations for training. We report the macro average accuracy on Test-unseen.

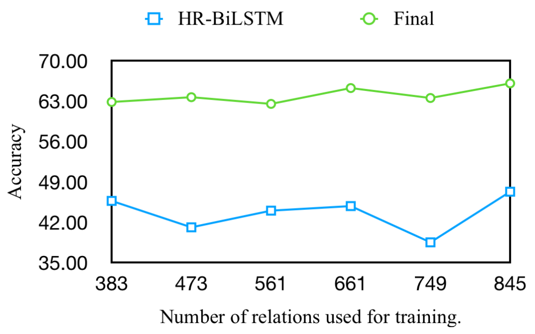

Figure 4: Macro average accuracy for different relation size in the training set.

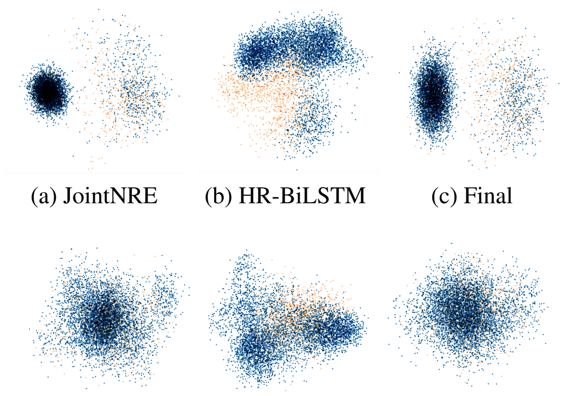

Figure 5: Relation Representation Visualization of different models. The yellow (light) point represent the seen relation, and the blue (dark) point represent the unseen relation.

As shown in Figure 4, with different number of relations, our model still perform better than HR-BiLSTM. Note that, our adapter can beat HR-BiLSTM with even a smaller number of seen relations. When there are more relations for training, the performance will be improved as expected.

Relation Representation Analysis We visualize the relation representation in JointNRE, HR-BiLSTM and the output representation of our final adapter by principal component analysis (PCA) with the help of TensorBoard. We use the yellow (light) point represents the seen relation, and the blue (dark) point represents the unseen relation.

As shown in Figure 5a), the JointNRE representation is pre-trained by the interaction between knowledge graph and text. Because without knowing the relation detection tasks, seen and unseen relations are randomly distributed. [3]

[3]We also notice that there is a big cluster of relations on the left hand side. This is presumably the set of less updated relations in the training of JointNRE, due to lack of correspondence with the text data. This cluster does not affect our main observation with adapter training.

Model	Accuracy
Final	77.3±7.6 / 73.0±1.7
Final*	**77.5±6.0** / 72.4±1.8

Table 5: Results on Test-unseen with and without the adapter in training JointNRE.

After training with HR-BiLSTM (Figure 5b), the seen and unseen relations are easily separated, because the training objective is to discriminate the seen relations from the other relations for the corresponding question. Although the embeddings of unseen relations are also updated due to negative sampling, they are never updated towards their correct position in the embedding space. As a result, the relation detection accuracy for the unseen relations is poor.

The training of our final model uses the adapter to fit the training data, instead of directly updating the embeddings. Despite the comparable performance on seen relations, the distribution of seen and unseen relations (Figure 5c) is much similar to the original JointNRE, which is the core reason for its ability to obtain better results on unseen relations.

Adapting JointNRE Interestingly, we notice that JointNRE is to train the embedding of relations with a corpus of text that may not cover all the relations, which is also a process that needs the adapter. As a simple solution, we use a similar adapter to adapt the representation from TransE [4] (Lin et al., 2015) to the training of JointNRE. With the resulting relation embedding, denoted as JointNRE*, we train the baseline and final relation detection models, denoted as HR-BiLSTM* and Final*, respectively.

We visualize the relation representation in these models again. Clearly, the distribution of seen and unseen relations in JointNRE* (Figure 5d) looks more reasonable than before. This distribution is interrupted by fine-tuning process of HR-BiLSTM* (Figure 5e), while is retained by our adapter model (Figure 5f).

Furthermore, as shown in Table 5, using JointNRE* can further improve the unseen relation detection performance (77.5% v.s. 77.3%). This provides further evidence of the importance of repre-

[4]https://github.com/thunlp/Fast-TransX

Question 1	who produced recording *Twenty One*
Candidate Relations	**music.recording.producer** music.recording.artist
HR-BiLSTM	music.recording.artist
Final	**music.recording.producer**
Question 2	what is *Tetsuo Ichikawa*'s profession
Candidate Relations	people.person.gender **people.person.profession**
HR-BiLSTM	**people.person.profession**
Final	**people.person.profession**
Question 3	which village is in *Arenac county* ?
Candidate Relations	location.us_county.hud_county_place **location.location.contains**
HR-BiLSTM	location.us_county.hud_county_place
Final	location.us_county.hud_county_place

Table 6: Case studies for relation detection using different models. For each question, the gold relation is marked with bold font; the gold target entity of the question is marked with italic font. The models and notations are the same as in Table 2.

sentations for unseen relations.

Case Study In the first case of Table 6, *Twenty One* is the subject of question. "music.recording.producer" is the gold relation, but it is an unseen relation. The baseline model predicts "music.recording.artist" because this relation is seen and perhaps relevant in the training set. A dig into the set of relations shown that there is a seen relation, "music.recording.engineer", which happens to be the closest relation in the mapped representation to the gold relation. It is possible that the knowledge graph embedding is able to capture the relatedness between the two relations.

In the second case, although the gold relation "people.person.profession" is unseen, both baseline and our model predict the correct answer because of strong lexical evidences: "profession".

In the last case, both the gold relation and predict error relation are unseen relation. "Hud_county_place" refers to the name of a town in a county, but "location.location.contains" has a broader meaning. When asked about "village", "location.location.contains" is more appropriate. This case shows that our model still can not process the minor semantic difference between word. We will leave it for future work.

6 Related Work

Relation Detection in KBQA Yu et al. (2017) first noticed the zero-shot problem in KBQA relation detection. They split relation into word sequences and use it as a part of the relation representation. In this paper, we push this line further and present the first in-depth discussion about this zero-shot problem. We propose the first relation-level solution and present a re-organized dataset for evaluation as well.

Embedding Mapping Our main idea of embedding mapping is inspired by previous work about learning the mapping of bilingual word embedding. Mikolov et al. (2013) observed the linear relation of bilingual word embedding, and used a small starting dictionary to learn this mapping. Zhang et al. (2017a) use Generative Adversarial Nets (Goodfellow et al., 2014) to learn the mapping of bilingual word embedding in an unsupervised manner. Different from this work which maps words in different languages, we perform mappings between representations generated from heterogeneous data, i.e., knowledge base and question-triple pairs.

Zero-Shot Learning Zero-shot learning has been studied in the area of natural language process. Hamaguchi et al. (2017) use a neighborhood knowledge graph as a bridge between out of knowledge base entities to train the knowledge graph. Levy et al. (2017) connect nature language question with relation query to tackle zero shot relation extraction problem. Elsahar et al. (2018) extend the copy actions (Luong et al., 2015) to solve the rare words problem in text generation. Some attempts have been made to build machine translation systems for language pairs without direct parallel data, where they relying on one or more other languages as the pivot (Firat et al., 2016; Ha et al., 2016; Chen et al., 2017). In this paper, we use knowledge graph embedding as a bridge between seen and unseen relations, which shares the same spirit with previous work. However, less study has been done in relation detection.

7 Conclusion

In this paper, we discuss unseen relation detection in KBQA, where the main problem lies in the learning of representations. We re-organize the SimpleQuestion dataset as SimpleQuestion-Balance to reveal and evaluate the problem, and propose an adapter which significantly improves the results.

We emphasize that for any other tasks which contain a large number of unseen samples, training, fine-tuning the model according to the performance on the seen samples alone is not fair. Similar problems may exist in other NLP tasks, which will be interesting to investigate in the future.

Acknowledgement

We would like to thank the anonymous reviewers for their insightful comments. Shujian Huang is the corresponding author. This work is supported by the National Science Foundation of China (No. 61772261), the Jiangsu Provincial Research Foundation for Basic Research (No. BK20170074). Part of this research is supported by the Huawei Innovation Research Program (HO2018085291).

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan. In *NIPS*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *JMLR*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *EMNLP*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD Conference*.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. In *EMNLP*.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *CoRR*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*.

Yun Chen, Yang Liu, Yong Cheng, and Victor OK Li. 2017. A teacher-student framework for zero-resource neural machine translation. In *ACL*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *ICLR*.

Zihang Dai, Lei Li, and Wei Xu. 2016. Cfo: Conditional focused neural question answering with large-scale knowledge bases. In *ACL*.

Hady Elsahar, Christophe Gravier, and Frederique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. In *NAACL*.

Orhan Firat, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T Yarman Vural, and Kyunghyun Cho. 2016. Zero-resource translation with multi-lingual neural machine translation. In *EMNLP*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*.

Thanh-Le Ha, Jan Niehues, and Alexander Waibel. 2016. Toward multilingual neural machine translation with universal encoder and decoder. *CoRR*.

Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. 2017. Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach. In *IJCAI*.

Xu Han, Zhiyuan Liu, and Maosong Sun. 2018. Neural knowledge acquisition via mutual attention between knowledge graph and text. In *AAAI*.

Yanchao Hao, Hao Liu, Shizhu He, Kang Liu, and Jun Zhao. 2018. Pattern-revising enhanced simple question answering over knowledge bases. In *COLING*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *CoNLL*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*.

Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation. In *ACL*.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *CoRR*.

Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. 2009. Zero-shot learning with semantic output codes. In *NIPS*.

Michael Petrochuk and Luke Zettlemoyer. 2018. Simplequestions nearly solved: A new upperbound and baseline approach. In *EMNLP*.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.*

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook, 2nd ed.* Springer.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*.

Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with freebase. In *ACL*.

Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schütze. 2016. Simple question answering by attentive convolutional neural network. In *COLING*.

Mo Yu, Wenpeng Yin, Kazi Saidul Hasan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2017. Improved neural relation detection for knowledge base question answering. In *ACL*.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017a. Adversarial training for unsupervised bilingual lexicon induction. In *ACL*.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017b. Earth mover's distance minimization for unsupervised bilingual lexicon induction. In *EMNLP*.