A2RN: Attending to Relation Structure for Knowledge Graph Inference

Hongqing Chen [1] and Yusha Xie [2]

***Abstract*— In the field of knowledge graph inference,embedding based models have achieved great success and quickly become the most mainstream inference method.But we noticed that previous embedding base models hardly introduced attention mechanism.We thus propose a novel attention-based method to make full use of the structural information contained in the neighbours of the relation in the triple leading to more accurate knowledge graph inference.As far as we know,no embedding models have ever used information contained in the neighbours of the relation to help inferring.We have tested our model on two standard data sets and compared it with several state-of-the-art models.The results showed that We have achieved significantly better performance than others on some data sets.**

I. INTRODUCTION

The knowledge graph is a structured way to organize information and the information are usually presented in the form of triples.A typical triple includes a head,a relation and a tail.Famous knowledge graphs include YAGO, DBpedia,Freebase and so on.Although these knowledge graphs are quite large and are still growing rapidly,the information stored in them are far from complete.So it's an important task to infer new triples based on the triples that already known.

Suppose a triple in the knowledge graph is represented in the form of (h, r, t),where h and r both entities and r is a relation.We call h the head,r the relation and t the tail.In the inference of the knowledge graph, the most common task can be described as given an entity and a relation,we need to infer the missing entity of the triple.Intuitively,we need to complete the tuple $(h, r, ?)$ or the tuple $(?, h, t)$.In fact, these two seemingly different problems can be transformed into one.

A lot of methods have been proposed to solve the problems stated above,among which knowledge graph embedding is an important method that cannot be neglected and is receiving increasing attention.TransE(Bordes et al. 2013) is the first work of the translating embedding model series,whose idea is quite simple but has proved to be very successful. It embeds the entities and relationships of a knowledge graph in low-dimensional vector spaces to simplify the manipulation while preserving the inherent structure of the knowledge graph. Following it,TransH(Wang et al. 2014),TransR(Lin et al. 2015) and TransD (Ji et al.2015) were proposed to make up for some defects of transE.

*Equal contribution on the main work. For report writing, Xie focus on Introduction, Related Work and Method; Chen focus on Experiment.

[1] Hongqing Chen is with Faculty of Computer Science, University of SJTU, 517021910856, Shine1999@sjtu.edu.cn

[2] Yusha Xie is with Faculty of Computer Science, University of SJTU, 517021910831, xys2017@sjtu.edu.cn

However,the embedding representations for an entity or a relation in these models depends only on the entities and relation involved in the triple.In other words,they didn't take the knowledge graph structure into consideration.

In our view,it's necessary to take advantage of the structure of the knowledge graph to optimize the embedding representations of entities and relation in the triple.In other words,it's beneficial to involve the triple's neighbors in the inference process.

We noticed that in recent years,some papers have already tried to introduce attention mechanism into knowledge graph inference.Both A2N(Bansal et al. 2019) and LENA(Kong et al. 2019) take advantage of the neighbours of the head to optimize the embedding representations of entities so as to improve accuracy of a specific query.

But still,they only consider the triples whose tail is the head of the triple to be inferred as the neighbours of it.We consider the triples whose relation is the same as the triple to be inferred's also as neighbours.

II. RELATED WORK

Knowledge graph embedding is an important method of knowledge graph inference.

TransE(Bordes et al. 2013) is the first to propose the representation of entities and relationships in the knowledge graph as low-dimensional vectors.Inspired by the fact that word vector space has translation invariant phenomenon,they believe a true triple will satisfy the formulation $\mathbf{h}+\mathbf{r}=\mathbf{t}$.So they use the scoring function $s(h, r, t) = ||\mathbf{h} + \mathbf{r} - \mathbf{t}||$ to measure the probability of a triple. In order to solve the limitations of the TransE model when dealing with one-to-many, many-to-one, and many-to-many complex relationships, the TransH(Wang et al. 2014) proposes that an entity has different representations in different relationships.Moreover,TransR(Lin et al. 2015) believes that an entity is a combination of multiple attributes, and different relationships should have different semantic spaces.

Unlike translation model series,Semantic matching models exploit similarity-based scoring functions. They measure plausibility of facts by matching latent semantics of entities and relations embodied in their vector space representations. RESCAL (Nickel, Tresp, and Kriegel 2011) obtains its latent semantics by using a vector to represent each entity and a matrix to represent each relationship which models the pairwise interaction between potential factors.Its scoring function is defined as $s(h, r, t) = \mathbf{h}^T\mathbf{M}_r\mathbf{t}$.Its Extensions include DistMult (Yang et al. 2014),HolE(Nickel et al. 2016) and ComplEx (Trouillon et al. 2016).

All the aforementioned didn't introduce attention mechanism into the embedding model.A2N(Bansal et al. 2019) and LENA(Kong et al. 2019) proposed to use information contained in the neighbourhood of the triple to optimize the vector representations of entities and relationships.

III. METHOD

As noted in the previous section, there are some KB embedding models whose "atomic" loss function is based on neighbors. The term of knowledge graph is synonymous with knowledge base with a minor difference. A knowledge graph can be viewed as a graph when considering its graph structure. Usually it is represented as a directed graph with nodes as entities and edges as relations. Therefore, when we talk about neighbors of a triple, we usually think of the neighbors of entities. In the context of knowledge graph completion, those graph-attention models mainly focus on the triples whose head is the same with the target triple. However, when we view knowledge as a triple, it is clear that if we focus on relations, we can have a different span of attention.

A. Problem Formulation

We consider a knowledge graph as a set of triples $G := \{(h, r, t)\}$. Each triple consists of a head entity h, a relation r and a tail entity t. For link inference, the goal is to predict the tail entity given a query of head entity and relation or to predict the head entity given a query of tail entity and relation.

B. Data Processing

Firstly, we combine these two tasks into one by expanding the relation set. For any triple (h, r, t) in G, it will have a reciprocal triple (h, r^-, t) with the reciprocal relation r^-. Then our task is only predicting the tail given a query of head and relation. The task of predicting the head can be converted to predicting the tail given a query of head and reciprocal relation.
Then, we build subsets based on relations. For any given triple (h, r, t), we define the span of attention as:

$$G'(h, r, t) := \{i \in G : r(i) = r, i \neq (h, r, t)\}$$

$G'(h, r, t)$ is the subset of G which contains all the triples whose relation is r except itself. Although in the graph, we can not see these triples besides the target triple, we still consider them neighbors and the subset can be regarded as the neighborhood of triple (h, r, t). As shown in Fig1,$G'(h, r, t)$ contains the edges $(h1, r, t1)$,$(h2, r, t2)$,$(h3, r, t3)$ and $(h4, r, t4)$.

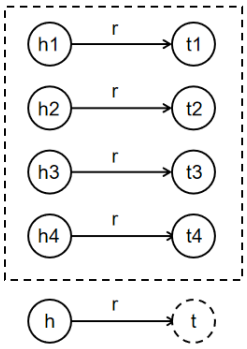

Fig. 1. Example of the subgraph $G'(h, r, t)$

C. Score Function

The scoring function is used to evaluate the possibility that a triple is valid. Many scoring functions have been proposed such as TransE,DistMult and ComplEx.We choose DistMult as our scoring function because of its simplicity and good performance.In fact,our model can be combined with any other scoring function.

Suppose we represent entities and relations in the form of k-dimensional vectors.In the scoring function,$v(h, r, t)$ is k-dimensional a vector which contains the information extracted from the neighbourhood of (h, r, t) and we'll explain it in detail later.$Diag(\mathbf{r})$ is a $k \times k$ diagonal matrix with $\mathbf{r}$ in its diagonal.

$$s(h, r, t) = (v(h, r, t))^\top Diag(\mathbf{r})\mathbf{t}$$

After getting the score $s(h, r, t)$ of each possible triple,we can get the probability $p(t|h, r)$ of it using the softmax function.N is the set of all entities.

$$p(t|h, r) = \frac{exp(s(h, r, t))}{\sum_{t' \in N} exp(s(h, r, t'))}$$

D. Attention Mechanism

After finding the span of attention $G'(h, r, t)$ of the triple (h, r, t),we'll explain how to extract information from $G'(h, r, t)$. Because the size of $G'(h, r, t)$ is usually quite large,we'll divide it into smaller subsets to reduce the computation.

Let L be prescribed positive integer.We define $H_L(h, r, t)$ as the set of all subsets of $G'(h, r, t)$ which contains L edges.The value L is then referred to as the window size. Let $\Gamma \in H_L(h, r, t)$ be an arbitrary window of $G(h, r, t)$ containing L edges.We'll use $h(l : \Gamma)$ to represent the head entity of the $l-$th edge in $Gamma$ and $t(l : \Gamma)$ to represent the tail entity of the edge.

Let vector $\alpha_\Gamma := [\alpha_\Gamma(0), \alpha_\Gamma(1), \cdots, \alpha_\Gamma(L)]^T$ represents the sets of attention weights.We have each element $\alpha_\Gamma(l) \geq 0$ and the sum $\sum_{l=0}^{L} \alpha_\Gamma(l) = 1$. The formulation of each element $\alpha_\Gamma(l)$ is:

$$\alpha_\Gamma(l) = \frac{exp\langle \gamma_h, \mathbf{h}(l : \Gamma)\rangle}{\sum_{j=0}^{L} exp\langle \gamma_h, \mathbf{h}(j : \Gamma)\rangle}$$

We note that the attention parameter γ_h only depends on the head entity h of the triple(h, r, t).Intuitively,we want to use head entities that resembles h in $G'(h, r, t)$ to optimize the embedding representation of it.

Using the attention weights defined above,the vector $v(\Gamma)$ of a window Γ is:

$$v(\Gamma) := \alpha_\Gamma(0)\mathbf{h} + \sum_{l=1}^{L} \alpha_\Gamma(l)\mathbf{h}(l : \Gamma)$$

To get $v(h, r, t)$,we'll apply a pooling operation across a fixed number of windows. Let $\tilde{H}_L(h, r, t)$ be a random subset of $H_L(h, r, t)$ containing H windows where H is a prescribed positive integer.So we can get $v(h, r, t)$:

$$v(h, r, t) := max\_pooling\{v(\Gamma) : \Gamma \in \tilde{H}_L(h, r, t)\}$$

Now we can see the information of the neighbourhood $G'(h, r, t)$ is contained in the vector $v(h, r, t)$ and we can use it to optimize the scoring function.

E. Network Structure

As shown in Fig 2,the network has three layers:two combination layers and one softmax layer.The first layer is used to extract information from the neighbourhood of r.The second layer is used to compute score of each candidate.The third layer uses softmax function to get the probability of each candidate.The blue nodes are entities and the green nodes are relations.Both entities and relations are $k-$dimensional vectors.The orange nodes are the vector which contains information of the neighbourhood.For clarity we only illustrate two candidates in Fig 2,however there can be an arbitrary number of candidate-entities.

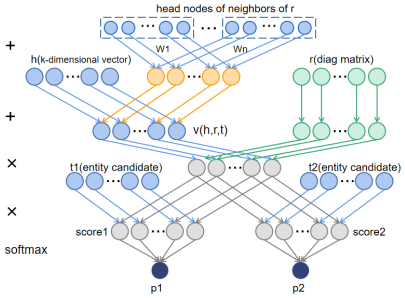

Fig. 2. network structure

F. Training

During training,first,we initialize the embeddings and parameters randomly.For a query $(h, r, ?)$,we find the neighbourhood of r and extract information from a part of its neighbourhood.To get useful information,we assign different weights to different neighbors and extract information from neighbors based on the weights.After getting $v(h, r, t)$ which contains the information of r's neighbours,we combine it with h to get the new head for prediction.Then we use the scoring function to score the candidate triples and use softmax function to get the probability of each of them.Finally,we update the embeddings and parameters to minimize the loss.

IV. EXPERIMENT

We evaluate the A2RN by doing link prediction on FB15K-237 and WIN18-RR dataset. The evaluation metrics we use are mean rank (MR), which refers to the average rank of all testing cases; top-10 hit(HIT) ,which means the percentage of the testing triples that have rank value no greater than 10 and reciprocal rank (MRR), the average of the multiplicative inverse of the rank value for all testing triples. Also, we compare our model with two state-of-art models Dismult and LENA.

A. Data Preparation

FB15K is a subset of FreeBase, a large-scale general-fact KB, and WN18 is a subset of WordNet, in which entities represent word senses and relations describe lexical relationships between two word senses. It has been noted that for FB15K and WN18 dataset, many testing triples are reciprocal to triples in the training set. So we use FB15K-237 and WN18-RR which are proposed by removing those reciprocal triples from FB15K and WN18. The statistics of the datasets are listed as in Table I. Then, in order to do the (?, r, t) prediction, we produce its reciprocal triple and add it to the training set for each triple in the origin training set.

TABLE I
THE STATISTICS OF DATASETS

Datasets	entities	relations	triples(train/test/valid)
FB15K-237	14,541	237	272,115/20,266/17,535
WIN18-RR	40,943	11	86,835/3,134/3,034

B. Baselines and Implementation Details

There are many score functions which can evaluate the possibility that a triple is valid. As we mentioned before, the attention models can be combined with any scoring function. Therefore, we employ Dismult as baseline and achieve the head attention model LENA and our relation attention model A2RN based on the Dismult score function. For each model, we set the embedding dimension as 200. We use mini-batch stochastic gradient descent (SGD) with learning rate of 0.01. The batch size is set as 100. And for the attention models, the window width is set as 4 and the window height is set as 30.

C. Results

We reimplement DisMult, and LENA model in our experiments. We compare A2RN with them on the dataset FB15K-237 and WINRR-18 as shown in Table II. We can see that our model gets the lowest MR and highest MRR and Hit on FB15K-237, which means it performs much better than the baseline models. On WINRR, DisMult performs the best, but we can see A2RN is close to DisMult. It may because that on FB15K-237, the number of relations is large and the neighbors have much information which is similar to the source head. However, on WINRR, the number of relations is only 11 and there is little similarity between

neighbors and the source head, so the performance of A2RN is similar to DisMult. However, it's not bad. There is a survey [11] that those seemingly more expressive models do not necessarily have better performance and the reason could be that expressive models often require a large number of parameters and tend to overfit on small- and medium-sized datasets. Therefore, A2RN has a good performance on both large-sized datasets and small-size datasets.

TABLE II
COMPARE WITH OTHER APPROACHS

Model	FB15K-237			WINRR		
	MR	MRR	Hit	MR	MRR	Hit
Dismult	588.7	13.3	24.9	6662.1	**26.5**	**40.8**
LENA	1883.2	11.7	21.1	**5251.8**	17.9	38.1
A2RN	**558.4**	**15.3**	**27.1**	6528.5	25.4	38.5

D. Further Analysis

1) Performance on Different relations: We compare the performance of different models on relations in WINRR.

TABLE III
PERFORMANCE ON DIFFERENT RELATIONS

realation	Metrics	Dismult	LENA	A2RN
0	MR	9825.3	7756.8	8191.5
	MRR	7.3	12.3	5.5
	HIT	11.1	19.9	8.8
1	MR	560.4	365.5	837.16
	MRR	62.2	33	60.3
	HIT	93.5	77.9	89.6
2	MR	2208.4	3586.8	1323.8
	MRR	26.2	31.9	31.4
	HIT	45.9	53.3	54.1
3	MR	5355.4	2520.5	3474.8
	MRR	43.5	24.3	33
	HIT	60.7	57.1	55.4
5	MR	3381.7	4715.4	2210.5
	MRR	17.9	39.9	19.9
	HIT	30.7	49.1	36
9	MR	364.1	664.8	1890.5
	MRR	81.2	32.9	77.5
	HIT	97.4	79.5	89.7
10	MR	1.3	14	1.3
	MRR	83.3	20.4	83.3
	HIT	100	66.7	100

ID	relations
0	_hypernym
1	_derivationally_related_form
2	_instance_hypernym
3	_also_see
4	_member_meronym
5	_synset_domain_topic_of
6	_has_part
7	_member_of_domain_usage
8	_member_of_domain_region
9	_verb_group
10	_similar_to

TABLE IV
ID MAP FOR RELATIONS

Some of the results which shows the different performance of these models are shown in Table III. The corresponding relation names of these IDs are shown in Table IV. We can see that LENA performs much better than A2RN and DisMult on hypernym but A2RN performs the best on _instance_hypernym. In fact, attending to relation neighbors of a triple means that we think triples of the same relation have much similarity. However, for _hypernym, the hypothesis is not suitable. For _instance_hypernym, the neighbors are all instance, which is already an importance common feature. Similarly, for _derivationally_related_for, the neighbors do not have much similarity. If we choose the neighbors of head, the performance would be good. From Table III, we can conclude that, if the relation is specific, A2RN would have the edge. And if the relation is simple and common, LENA would perform better.

2) Using Tail Information: For entity-based attention models, as the head and tail entities have the same domain, both the head and relation neighbors and be used. However, for relation-based attention models, we can only use the head neighbors. We then try to utilize the information of the tail of neighbors. We use the same attention to compute the and view it as a prediction of tail.

$$v(\Gamma) := \alpha_\Gamma(0)\mathbf{t} + \sum_{l=1}^{L} \alpha_\Gamma(l)\mathbf{t}(l:\Gamma)$$

$$pred(t) := max\_pooling\{v(\Gamma) : \Gamma \in \tilde{H}_L(h,r,t)\}$$

We define the weighted prediction score as the different of the predicted tail and the real tail. The score function then would be

$$s(h,r,t) = (v(h,r,t))^\top Diag(\mathbf{r})\mathbf{t} - ||pred(t) - \mathbf{t}||_F$$

The new model A2RN++ is evaluated on FB15K-237 and the result is shown in Table V. The performance is worse and a possible reason may be that the model is too complex to train. From this experiment, we can also see that the attention weight should be different for head, relation and tail and that is why LENA would learning different weight for head and relation. Although head and tail are both entity, we still can not use the attention of head to do the prediction of tail.

TABLE V
PERFORMANCE OF A2RN++ ON FB15K-237

Model	FB15K-237		
	MR	MRR	Hit
A2RN++	3508.4	13.6	21.9

V. CONCLUSION

We introduced attention mechanism into knowledge graph inference and proposed A2RN,an attention based model which makes full use of the neighbourhood of relation to get more accurate results.We show that by using the information of relational neighbors, we can achieve significantly better results on certain inference tasks and certain data sets.

ACKNOWLEDGMENT

Thanks for the guidance and help of teacher Luoyi Fu and teaching assistants of the class EE447 of Shanghai Jiaotong University.

REFERENCES

[1] Bansal, B.;Jaun, D.;Ravi, S.;and McCallum, A.2019.A2N: Attending to Neighbors for Knowledge Graph Inference.In *ACL*,4387–4392.

[2] Bordes, A.; Usunier, N.; García-Durán, A.; Weston, J.; and Yakhnenko, O. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NIPS*, 2787–2795.

[3] Ji, G.;He, S.;Xu, L.;Liu, K.; and Zhao, J.2015.Knowledge graph embedding via dynamic mapping matrix.In *ACL*,687–696.

[4] Kong, F.;Zhang, R.;Mao Y.; and Deng T.2019.LENA: Locality-Expanded Neural Embedding for Knowledge Base Completion.In *AAAI*.

[5] Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; and Zhu, X. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *AAAI*, 2181–2187.

[6] Nickel, M.; Rosasco, L.; and Tomaso, P. 2016. Holographic Embeddings of Knowledge Graphs. In *AAAI*.

[7] Nickel, M.; Tresp, V.; and Kriegel, H.-P. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *ICML*, 809–816.

[8] Shi, B., and Weninger, T. 2017. ProjE: Embedding projection for knowledge graph completion. In *AAAI*, 1236–1242.

[9] Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *AAAI*, 1112–1119.

[10] Yang, B.; Yih, W.; He, X.; Gao, J.; and Deng,L. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv*:1412.6575.

[11] Q. Wang, Z. Mao, B. Wang and L. Guo, "Knowledge Graph Embedding: A Survey of Approaches and Applications," in IEEE Transactions on Knowledge and Data Engineering, vol. 29, no. 12, pp. 2724-2743, 1 Dec. 2017, doi: 10.1109/TKDE.2017.2754499.