Incorporating Selectional Preferences in Multi-hop Relation Extraction

Rajarshi Das, Arvind Neelakantan, David Belanger and Andrew McCallum
College of Information and Computer Sciences
University of Massachusetts Amherst
{rajarshi, arvind, belanger, mccallum}@cs.umass.edu

Abstract

Relation extraction is one of the core challenges in automated knowledge base construction. One line of approach for relation extraction is to perform multi-hop reasoning on the paths connecting an entity pair to infer new relations. While these methods have been successfully applied for knowledge base completion, they do not utilize the entity or the entity type information to make predictions. In this work, we incorporate *selectional preferences*, i.e., relations enforce constraints on the allowed entity types for the candidate entities, to multi-hop relation extraction by including entity type information. We achieve a 17.67% (relative) improvement in MAP score in a relation extraction task when compared to a method that does not use entity type information.

1 Introduction

Knowledge Bases (KB's) are structured knowledge sources widely used in applications like question answering (Kwiatkowski et al., 2013; Berant et al., 2013; Bordes et al., 2014) and search engines like *Google Search* and *Microsoft Bing*. This has led to the creation of large KB's like Freebase (Bollacker et al., 2008), YAGO (Suchanek et al., 2007) and NELL (Carlson et al., 2010). KB's contains millions of facts usually in the form of triples $(entity1, relation, entity2)$. However, KB's are woefully incomplete (Min et al., 2013), missing important facts, and hence limiting their usefulness in downstream tasks.

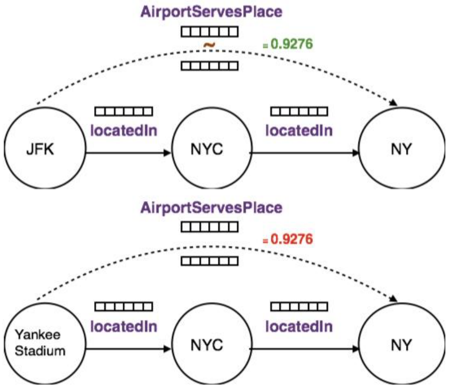

Figure 1: The two paths above consist of the same relations ($\texttt{locatedIn} \rightarrow \texttt{locatedIn}$) and, hence, the model of Neelakantan (2015) will assign them the same score for the relation *AirportServesPlace* without considering the fact that *Yankee Stadium* is not an airport.

To overcome this difficulty, Knowledge Base Completion (KBC) methods aim to complete the KB using existing facts. For example, we can infer nationality of a person from their place of birth. A common approach in many KBC methods for relation extraction is reasoning on individual relations (single-hop reasoning) to predict new relations (Mintz et al., 2009; Bordes et al., 2013; Riedel et al., 2013; Socher et al., 2013). For example, predicting *Nationality(X, Y)* from *BornIn(X, Y)*. The performance of relation extraction methods have been greatly improved by incorporating *selectional preferences*, i.e., relations enforce constraints on the allowed entity types for the candidate entities, both in sentence level (Roth and Yih, 2007; Singh et al., 2013) and KB relation extraction (Chang et al., 2014), and in learning entailment rules (Berant et al., 2011).

Another line of work in relation extraction performs reasoning on the paths (multi-hop reasoning on paths of length $\geq$ 1) connecting an entity pair (Lao et al., 2011; Lao et al., 2012; Gardner et al., 2013; Gardner et al., 2014; Neelakantan et al., 2015; Guu et al., 2015). For example, these models can infer the relation `PlaysInLeague(Tom Brady, NFL)` from the facts `PlaysForTeam(Tom Brady, New England Patriots)` and `PartOf(New England Patriots, NFL)`. All these methods utilize only the relations in the path and do not include any information about the entities.

In this work, we extend the method of Neelakantan (2015) by incorporating entity type information. Their method can generalize to paths unseen in training by composing embeddings of relations in the path non-linearly using a Recurrent Neural Network (RNN) (Werbos, 1990). While entity type information has been successfully incorporated into relation extraction methods that perform single hop reasoning, here, we include them for multi-hop relation extraction. For example, Figure 1 illustrates an example where reasoning without type information would score both the paths equally although the latter path should receive a lesser score since there is an entity type mismatch for the first entity. Our approach constructs vector representation of paths in the KB graph from representations of relations and entity types occurring in the path. We achieve a 17.67% improvement in Mean Average Precision (MAP) scores in a relation extraction task when compared to a method that does not use entity type information. Lastly, the *SHERLOCK* system (Schoenmackers et al., 2010) also discovers multi-hop clauses using typed predicates from web text, but, unlike our RNN approach it employs a Inductive Logic Programming method.

2 Model

This paper extends the Recurrent Neural Network model of Neelakantan (2015) by jointly reasoning over the relations and entity types occurring in the paths between an entity pair. Paths are represented as dense vectors formed by composing embeddings of relations and entities occurring at each step. Figure 2 illustrates the encoder architecture for a path between an entity pair. The $[\cdot]$ in figure 2 denotes the concatenate operation. As will be described later, we also try representing the entity by its observed types.

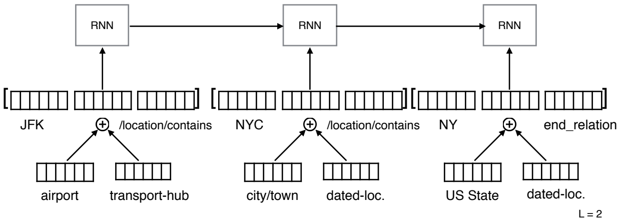

Figure 2: The encoder network for a path between an entity pair. The inputs to the network are embeddings of entities, entity types and relations. This architecture corresponds to equation 4 below. The network for other equations can be obtained by setting the appropriate input embeddings to zeros. Also note we have a dummy relation token `end_relation` for the last entity of the path. In the network above, at each time step, the entity embedding is concatenated with the sum of its type embeddings, followed by the embeddings of the relation type and are fed as input to the recurrent network

The relation types considered in our work are either fixed symbolic types defined in the Freebase schema such as `/people/person/nationality` or a free text relation from Clueweb (Orr et al., 2013) such as `born in`. In Freebase, an entity is associated with several types. For example, the entity `Barack Obama` has types such as `President`, `Author` and `Award Winner`. In our work, we consider the top l types (sorted by corpus frequency) for an entity and we obtain a combined representation by summing the embeddings of types.

Let $v_r(\delta) \in \mathbb{R}^d$ denote the vector representation of relation type δ. Let $v_e\left(e\right) \in \mathbb{R}^m$ denote the vector representation of an entity e and $v_{et}\left(e\right) \in \mathbb{R}^n$ denote the combined representation of the types of e obtained by taking the sum of the representation of its top l types. Let π be a path between the entity pair $(e1, e2)$ containing the relation types $\delta_1, \delta_2, \ldots, \delta_N$.

In the following section, we first briefly describe the model proposed by Neelakantan (2015) (RNN model henceforth) followed by our extensions to it.

2.1 RNN Model

The RNN model only considers the representations of relation type present in the path. More precisely, the vector representation $h_t \in \mathbb{R}^p$ of path $\delta_1, \delta_2, \ldots, \delta_t$ $(1 \leq t \leq N)$ is computed recursively as

$$h_t = f\left(W_{hh}h_{t-1} + W_{rh}v_r\left(\delta_t\right)\right) \qquad (1)$$

The vector representation of the entire path is h_N where N is the length of the path. Here $W_{h,h} \in \mathbb{R}^{p \times p}$ and $W_{rh} \in \mathbb{R}^{p \times d}$ are composition matrices between the previous step in the path and the relation vector at the current step respectively and f is a non-linear activation function.

Extension with entity (and types)

The previous model can be extended to incorporate the embeddings of entities along with relations occurring at each step in the path. We consider learning a separate representation for every entity and representing an entity using its entity types.

- RNN + Entity: In this model, we add the embedding of the entity.

$$h_t = f\left(W_{hh}h_{t-1} + W_{rh}v_r\left(\delta_t\right) + W_{eh}v_e\left(e_t\right)\right) \qquad (2)$$

- RNN + Type: In this model, we add the embedding of the *entity* obtained from its types at each step.

$$h_t = f\left(W_{hh}h_{t-1} + W_{rh}v_r\left(\delta_t\right) + W_{th}v_{et}\left(e_t\right)\right) \qquad (3)$$

- RNN + Entity + Type: In this model, we use both the representations of the entity.

$$\begin{aligned} h_t = f(&W_{hh}h_{t-1} + W_{rh}v_r\left(\delta_t\right) \\ &+ W_{eh}v_e\left(e_t\right) + W_{th}v_{et}\left(e_t\right)) \end{aligned} \qquad (4)$$

Here e_t denotes the t^{th} entity occurring in the path between an entity pair and $W_{eh} \in \mathbb{R}^{p \times m}, W_{th} \in \mathbb{R}^{p \times n}$ are new composition matrices due to the entity and its types respectively. In all of our experiments f is the sigmoid activation function.

2.2 Model Training

We train a separate RNN model for each target relation[1]. The parameters for each model are the embedding of the relations, entities and types, and the various composition matrices (as applicable) . They are trained to maximize the likelihood of the training data.The score of a path π w.r.t to the target relation δ is

$$score(\pi, \delta) = \sigma\left(v\left(\pi\right) \cdot v\left(\delta\right)\right) \qquad (5)$$

We then choose the path which has the highest score similar to (Weston et al., 2013; Neelakantan et al., 2014). Selecting just one path (out of typically hundreds to thousands of paths) between entity pairs might lead to our model ignoring informative paths, especially during the initial stages of training. To alleviate this issue we also experiment by selecting the top k paths that have the highest score for a given entity pair and relation with the resultant score being the average of the top k scores.

3 Experiments & Results

In all of our experiments, we set the dimension of the relations, entity and their type embeddings to be 50. For a fair comparison with our model, which has more number of parameters due to the entity and/or type embeddings, we experiment by varying the dimension of the relation embeddings between 50, 100 and 150 for the baseline model. We use Adam (Kingma and Ba, 2014) for optimization with the default hyperparameter settings. The models are trained for 15 epochs beyond which we observed overfitting on a held-out development set. We set $l = 7$ and $k = 5$ in our experiments. We experiment with 12 target relations.

3.1 Data

We run our experiments on the dataset released by Neelakantan el al. (2015) which is a subset of Freebase enriched with information from ClueWeb. The dataset comprises of entity pairs with a set of paths connecting them in the knowledge graph. The negative examples comprise of entity pairs for which the given query relation does not hold. However the paths had the entity information missing

[1] We are working on having a single model which can predict all relations as that would be more ideal than having a single specialized RNN for each relation

Stats	Full dataset	Current experiments
# test relations	46	12
# entity pairs	3.22M	839K
# entity pairs (train)	605K	161K
# entity pairs (test)	2M	533K
Avg. paths /relation	3.77M	3.43 M

Table 1: Statistics of the dataset

from them and only contained the relation types occurring in them. For example, consider the path $SatyaNadella \xrightarrow{ceoAt} Microsoft \xrightarrow{locatedIn} Seattle \xrightarrow{cityIn} Washington$. The original dataset had the entities in-between such as $Microsoft$ and $Seattle$ missing from it.

We augment the dataset with the entities present in them. To gather the entities, we do a depth first traversal starting from the first entity of the entity pair and following the relation types until we reach the last entity of the pair. In cases of one-to-many relations we choose the next entity to be traversed at random. Due to the combinatorial search space we limit the total number of edges traversed beyond which we ignore the path. Therefore the number of paths between an entity pair would be less than in the original dataset. However, we are continuously augmenting the dataset and the latest version of the dataset can be downloaded from `http://iesl.cs.umass.edu/downloads/akbc16/`. Table 1 displays some statistics of the dataset gathered till now and also the subset that was used for running the current experiments.

3.2 Link Prediction

We compare our models with the baseline model on predicting whether an entity pair participates in a target relation. We rank the entity pairs in the test set based on their scores and calculate the Mean Average Precision (MAP) score for the ranking following previous work (Riedel et al., 2013; Neelakantan et al., 2015). Table 2 lists the MAP scores of both the models averaged over 12 freebase relation types.

Incorporating *selectional preferences* by adding entity types gives a significant boost in scores (17.67 % over the baseline model.). However, we see a drop in performance on adding just entities. This is primarily because during test time we encounter a lot of previously unseen entities and hence we do not have learned embeddings for them. We overcome this problem by representing the entity using its observed types in Freebase. In future work, we would consider using pre-trained entity embeddings and also by representing the entity additionally using context words (Yaghoobzadeh and Schütze, 2015).

Model		**MAP**
Max	RNN (50)	0.5991
	RNN (100)	0.6020
	RNN (150)	0.6272
	RNN + Entity	0.5593
	RNN + Entity + Type	0.5995
	RNN + Types	**0.7084**
Top-K	RNN (50)	0.6241
	RNN (100)	0.6184
	RNN (150)	0.6312
	RNN + Entity	0.5968
	RNN + Entity + Type	0.6322
	RNN + Types	**0.7014**

Table 2: Mean Average Precision scores averaged over 12 relations. The number in the parentheses denotes the dimension of the embedding of the relations type in the baseline model.

.

Although considering top-k paths improves the performance of the baseline model, we observe that they provide almost similar scores with entity types. We run our experiments with $k = 5$ and we hope that the results would get better if we tune for k.

3.3 Predictive Paths

Table 3 shows maximum scoring paths for four entity pair and freebase relation triples chosen by the baseline and our model. We often find that the paths chosen by the baseline model have noisier textual relation, (like ‘`London`’[2] ,‘`and at the`’) and have entities belonging to very different types than expected by the query relation. For example, in table 3, the path chosen by the baseline model for `'/aviation/airport/serves'` goes to a music education school, and a water body and for `'/education/campus/institution'`, it goes to a country in which the institution is situated followed by a notable person in the country (unrelated to the query relation). It is quite clear that

[2] The freetext relation is different from the entity ‘London’ also occurring in the path

Relation : `/aviation/airport/serves` (Does the airport serve the location?)

Baseline Path: (0.5174)

Sandy_Lake_Airport $\xrightarrow{\texttt{(/location/contains)}^{-1}}$ Ontario $\xrightarrow{\texttt{and at the}}$ Toronto_Royal_Conservatory_Of_Music $\xrightarrow{\texttt{(including the)}^{-1}}$ Canada $\xrightarrow{\texttt{(geography/lake/basin\_countries)}^{-1}}$ Big_Trout_Lake $\xrightarrow{\texttt{and to}}$ Sandy_Lake_First_Nation.

Our Model Path: (**0.9502**)

Sandy_Lake_Airport $\xrightarrow{\texttt{(/location/contains)}^{-1}}$ Ontario $\xrightarrow{\texttt{(in northwestern)}^{-1}}$ Sandy_Lake_First_Nation.

Relation : `/aviation/airport/serves`

Baseline Score: (0.4348), Our Model Score: (**0.9731**) (Same path chosen by both models)

St._Mary's_Airport $\xrightarrow{\texttt{(/location/contains)}^{-1}}$ Wade_Hampton_Census_Area $\xrightarrow{\texttt{/location/us\_county/hud\_county\_place}}$ St._Mary's

Relation : `/education/campus/institution` (Is the educational institution located in this campus?)

Baseline Path: (0.4869)

Gray's Inn $\xrightarrow{\texttt{London}}$ England $\xrightarrow{\texttt{(/people/person/nationality)}^{-1}}$ Roger Fry $\xrightarrow{\texttt{/people/deceased\_person/place\_of\_death}}$ London $\xrightarrow{\texttt{/location/contains}}$ City_Law_School

Our Model Path: (**0.9676**)

Gray's Inn $\xrightarrow{\texttt{(/location/contains)}^{-1}}$ London_Borough_of_Camden $\xrightarrow{\texttt{/location/contains}}$ City_Law_School

Relation : `/geography/river/mouth` (Does the river (tributary) flow into the other river?)

Baseline Path: (0.4578)

Gard_River $\xrightarrow{\texttt{/geography/river/basin\_countries}}$ Romania $\xrightarrow{\texttt{(/geography/river/basin\_countries)}^{-1}}$ Jijia_River

Our Model Path: (**0.9231**)

Gard_River $\xrightarrow{\texttt{(/location/contains)}^{-1}}$ Botosani_County $\xrightarrow{\texttt{/location/contains}}$ Jijia_River

Table 3: Predictive paths chosen by the baseline and our model for four entity pair and relation triples. The relations are edge labels and the entities occur in between them and at the ends. The freebase relations starts with '`/`', (`/location/contains`, for e.g.). Inverse relations are denoted by $^{-1}$ i.e. $r(x, y) \implies r^{-1}(y, x), \forall(x, y) \in r$. The scores are given in parentheses (higher is better). Sometimes, both models find the same path (second example in `/aviation/airport/serves`), but we often find that our model correctly scores it higher.

.

adding entity types helps us incorporate *selectional preference* and hence eliminate lot of noisy paths. We also find that sometimes both models finds the same max scoring path but our model assigns more confidence (higher scores) to them leading to better MAP scores.[3]

4 Conclusion

In this work, we incorporate *selectional preferences* to a multi-hop relation extraction method. We have released the dataset we collected for this project. We achieve a 17.67% relative improvement in MAP score in a relation extraction task when compared to a method that does not use entity type information.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by DARPA under agreement number FA8750-13-2-0020. The U.S. Govt. is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon, in part by DARPA contract number HR0011-15-2-0036, and in part by the NSF grant number IIS-1514053. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

[3] The reader can browse more examples at `http://people.cs.umass.edu/~rajarshi/paths.html`.

References

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *NAACL*.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, and Christopher D. Manning. 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 *ICDM*.

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

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

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

Kai-Wei Chang, Wen tau Yih, Bishan Yang, and Christopher Meek. 2014. Typed tensor decomposition of knowledge bases for relation extraction. In *EMNLP*.

Matt Gardner, Partha Pratim Talukdar, Bryan Kisiel, and Tom M. Mitchell. 2013. Improving learning and inference in a large knowledge-base using latent syntactic cues. In *EMNLP*.

Matt Gardner, Partha Talukdar, Jayant Krishnamurthy, and Tom Mitchell. 2014. Incorporating vector space similarity in random walk inference over knowledge bases. In *EMNLP*.

K. Guu, J. Miller, and P. Liang. 2015. Traversing knowledge graphs in vector space. In *EMNLP*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke. Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *EMNLP*.

Ni Lao, Tom Mitchell, and William W. Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *EMNLP*, Stroudsburg, PA, USA.

Ni Lao, Amarnag Subramanya, Fernando Pereira, and William W. Cohen. 2012. Reading the web with learned syntactic-semantic inference rules. In *EMNLP*.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *NAACL*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *EMNLP*, Doha, Qatar.

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional vector space models for knowledge base completion. In *ACL*, Beijing, China.

Dave Orr, Amar Subramanya, Evgeniy Gabrilovich, and Michael Ringgaard. 2013. 11 billion clues in 800 million documents: A web research corpus annotated with freebase concepts. `http://googleresearch.blogspot.com/2013/07/11-billion-clues-in-800-million.html`.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *NAACL*.

Dan Roth and Wen-tau Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. In *In Introduction to SRL*.

Stefan Schoenmackers, Oren Etzioni, Daniel S. Weld, and Jesse Davis. 2010. Learning first-order horn clauses from web text. In *EMNLP*.

Sameer Singh, Sebastian Riedel, Brian Martin, Jiaping Zheng, and Andrew McCallum. 2013. Joint inference of entities, relations, and coreference. In *AKBC, CIKM*.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *WWW*.

P. Werbos. 1990. Backpropagation through time: what does it do and how to do it. In *Proceedings of IEEE*, volume 78.

Jason Weston, Ron Weiss, and Hector Yee. 2013. Nonlinear latent factorization by embedding multiple user interests. In *RecSys*.

Yadollah Yaghoobzadeh and Hinrich Schütze. 2015. Corpus-level fine-grained entity typing using contextual information. In *EMNLP, 2015*, pages 715–725.