Knowledge Base Inference using Bridging Entities

Bhushan Kotnis
Indian Institute of Science
bkotnis@dese.iisc.ernet.in

Pradeep Bansal
Indian Institute of Science
pradeepb@ee.iisc.ernet.in

Partha Talukdar
Indian Institute of Science
ppt@serc.iisc.in

Abstract

Large-scale Knowledge Bases (such as NELL, Yago, Freebase, etc.) are often sparse, i.e., a large number of valid relations between existing entities are missing. Recent research have addressed this problem by augmenting the KB graph with additional edges mined from a large text corpus while keeping the set of nodes fixed, and then using the Path Ranking Algorithm (PRA) to perform KB inference over this augmented graph. In this paper, we extend this line of work by augmenting the KB graph not only with edges, but also with *bridging entities*, where both the edges and bridging entities are mined from a 500 million web text corpus. Through experiments on real-world datasets, we demonstrate the value of bridging entities in improving the performance and running time of PRA in the KB inference task.

1 Introduction

Large-scale knowledge bases (KB) like Freebase (Bollacker et al., 2008), Yago (Suchanek et al., 2007), NELL (Mitchell et al., 2015) can be useful in a variety of applications like natural language question answering, semantic search engines, etc. These knowledge bases consist of millions of real world entities and relationships between them which are stored in the form of a directed graph where links represent relations and nodes represent the entities. Although such KBs contain millions of entities, they are still very sparse, i.e., they are missing a large number of relations between existing entities (West et al., 2014).

Performing inference over the knowledge graph for predicting relations between two entities is one way of densifying the KB graph. For example,

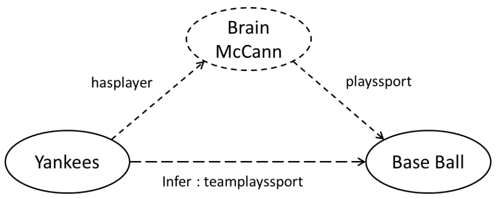

Figure 1: Example showing how addition of the *bridging entity*, *Brian McCain*, and the two edges incident on it can help the PRA algorithm (Lao and Cohen, 2010) to infer the initially missing relation instance *teamPlaysSport(Yankees, BaseBall)*. The original KB graph consisted only of two nodes, *Yankees* and *Baseball*, and no edges.

from *(Germany, playsinTournament, FIFA)* and *(FIFA, tournamentofSport, Soccer)*, we can infer *(Germany, playsSport, Soccer)*. The Path Ranking Algorithm (PRA) (Lao and Cohen, 2010), (Lao et al., 2011) performs such an inference by learning inference rules over the knowledge graph.

If the knowledge graph is sparse, i.e., if there are a very few or no paths between source and target entities, then PRA is unable to predict the existence of a relation. To address this shortcoming, (Lao et al., 2012) augmented the knowledge graph with paths obtained from an external corpus. The added paths consisted of unlexicalized dependency labels obtained from a dependency parsed external corpus. To improve the expressivity of the added paths, instead of the unlexicalized labels, (Gardner et al., 2013) augmented the KB graph with verbs (surface relations) from a corpus containing over 600 million Subject-Verb-Object (SVO) triples. These verbs act as edges that connect previously unconnected entities thereby increasing the connectivity of the KB graph which can potentially improve PRA performance.

However, naïvely adding these edges increases the feature sparsity which degrades the discriminative ability of the logistic regression classifier used in PRA. This can be addressed by adding latent relations obtained by clustering the surface relations, instead of directly adding the surface relations. This reduces feature sparsity and has been shown to improve PRA inference (Gardner et al., 2013) , (Gardner et al., 2014).

In this article we propose a scheme for augmenting the KB using paths obtained by mining noun phrases that connect two SVO triples from an external corpus. We term these noun phrases as *bridging entities* since they bridge two KB relations to form a path. This is different from the scheme in (Gardner et al., 2013) and (Gardner et al., 2014), which adds edges between KB nodes by mining surface relations from an external corpus. We search for such bridging entities in the corpus by performing a limited depth DFS (depth first search) on the corpus graph in an *on-demand* fashion.

We term this procedure as **On-Demand Augmentation (ODA)**, because the search can be performed during test time in an on-demand manner. In contrast, the previous approaches of adding edges or embeddings to the KB (Gardner et al., 2013), and vector space random walk PRA (Gardner et al., 2014) are batch procedures. As we shall see in Section 4, due to a limited search space, on-demand augmentation is much faster compared to algorithms in (Gardner et al., 2013; Gardner et al., 2014). Furthermore, since edges are not added blindly, on-demand augmentation does not increase feature sparsity which is responsible for performance degradation. Our experiments suggest that ODA provides better performance than (Gardner et al., 2013) and nearly the same prediction performance as provided by (Gardner et al., 2014), but in both cases with the added advantage of faster running time and greater flexibility due to its online and on-demand nature. The code along with the results can be obtained at https://github.com/malllabiisc/pra-oda.

2 Related Work

Using surface level relations and noun phrases for extracting meaningful relational facts is not a new idea (Hearst, 1992),(Brin, 1999), (Etzioni et al., 2004). However, none of them make use of Knowledge Bases for improving information extraction.

The Path Ranking Algorithm (PRA) first proposed in (Lao and Cohen, 2010) was used for performing inference over a KB in (Lao et al., 2011). It was extended by (Lao et al., 2012), to improve the inference by augmenting the KB with syntactic information obtained from a dependency parsed corpus. Augmenting the KB for improving PRA inference using surface relations mined from an external corpus and using latent edge labels obtained by performing PCA on the surface relations was explored in (Gardner et al., 2013). Instead of hard mapping of surface relations to latent embeddings, (Gardner et al., 2014) perform a 'soft' mapping using vector space random walks. This allows the random walker to traverse an edge semantically similar to the current edge type more frequently than other edges.

Although, like others, we too use an external corpus to augment the KB, the crucial difference in our approach is that apart from adding surface relations, we also add bridging entities that enable us to create new paths in the KB. Furthermore, the procedure is targeted so that only paths that play a part in inferring the relations that are of interest are added. Thus, the number of paths added in this manner is much lower than the number of surface relations added using the procedure in (Gardner et al., 2013). As we shall see in Section 4, this results in a more effective algorithm with faster runtime.

3 Method

3.1 Background: Path Ranking Algorithm (PRA)

We first present a brief overview of the Path Ranking Algorithm (PRA) (Lao and Cohen, 2010). The PRA uses paths as features for a logistic regression classifier which predicts if the given relation exists between a pair of entities. For a given pair of entities s and t, the *path type* connecting s to t form the feature vector. A *path types* π is an ordered set of relations. Paths with the same ordered relations but different intermediate or terminal entities belong to the same path type. For example, $s_1 \xrightarrow{v_0} x_1 \xrightarrow{v_1} t_1$ and $s_2 \xrightarrow{v_0} x_2 \xrightarrow{v_1} t_2$ belong to path type $\xrightarrow{v_0}\xrightarrow{v_1}$. The value of a feature, is taken to be $P(s \rightarrow t; \pi)$, where $P(s \rightarrow t; \pi)$ is the probability of reaching t from s by traversing paths of type π. PRA approximates these probabilities by running a random walk (RW) on the KB graph. Let $F = \{\pi_1, \pi_2, ..., \pi_k\}$ be the set of all path types. For predicting the existence of relation r between entities s and t, the logistic regression classifier outputs a score which is a measure of the

Query	Candidate Answer	Path added by PRA-ODA with bridging entity (in bold)
sportsteamPositionForSport(right handed pitcher, ?)	baseball	right handed pitcher $\xrightarrow{playsfor}$ **Chicago Cubs** $\xrightarrow{play}$ baseball
riverFlowsThroughCity(Moselle, ?)	Koblenz	Moselle $\xrightarrow{flows\ into}$ **Rhine** $\xrightarrow{meet\ at}$ Koblenz
teamPlaysInLeague(Cleveland Indians, ?)	MLB	Cleveland Indians $\xrightarrow{play}$ **Detroit Tigers** $\xrightarrow{blew}$ MLB

Table 1: Examples of paths involving bridging entities (marked in bold) added to the KB by PRA-ODA.

confidence that r exists between s and t. It does so by first assigning weights to the features in the training phase. The score is given by

$$S(s,t,r) = \sum_{\pi \in F} P(s \to t; \pi) \times \theta_\pi^r \qquad (1)$$

where θ_π^r is the weight learned by the logistic regression classifier during training specially for relation r and path type π. During the test phase, since targets are not available, the PRA gathers candidate targets by performing a random walk and then computes feature vectors and the score.

3.2 PRA-SVO and PRA-VS

PRA-SVO and PRA-VS are the systems proposed in (Gardner et al., 2013) and (Gardner et al., 2014) respectively, where the KB graph is augmented with edges mined from a large subject-verb-object (SVO) triple corpus. In these two systems, only new edges are added over the fixed set of nodes, and the augmentation happens in a batch, offline setting. In contrast, PRA-ODA, the method proposed in the paper, also expands the set of nodes through bridging entities, and performs the augmentation in an on-demand manner.

3.3 PRA On-Demand Augmentation (PRA-ODA)

Training: Let s and t be any two KB entities and let $s^{(n)}$ and $t^{(n)}$ be their corresponding noun phrase representations or aliases. We search for bridging entities $x_1, x_2, ..x_n$ by performing limited depth first search (DFS) starting with s^n such that we obtain a path $s \xrightarrow{ALIAS} s^{(n)} \xrightarrow{v_0} x_1 \xrightarrow{v_1}$ ... $\xrightarrow{v_{n-1}} x_n \xrightarrow{v_n} t^{(n)} \xrightarrow{ALIAS} t$, where v_i are verbs present in the corpus graph. This is done for all $n \leq d_{max} - 1$, where d_{max} is the maximum depth of DFS. We add an 'ALIAS' edge between the KB entity and its noun phase representation. The usefulness of bridging entities is illustrated in Fig. 1.

We mine bridging entities from a corpus containing over 600 million SVO triples which were obtained from the ClueWeb09 corpus (Callan et al., 2009) parsed using the MALT parser (Nivre et al., 2007). We use Mongo DB to store the triples as an adjacency list. During training time, for any relation that is being inferred, both the source and its corresponding target entities are known. A limited depth DFS is performed for *all depths* less then d_{max} on the SVO graph with the aliases of subject entity acting as the starting points. Such aliases are available for the NELL and Freebase knowledge bases. The DFS is said to discover a path if the terminating entity of the path matches any alias of the target entity. We choose to use aliases to perform string match, since it is easy to change the softness of the match by simply adding more aliases. This is done for all training source-target pairs. A few examples of added paths are shown in Table 1.

The SVO graph is noisy since it is obtained by parsing the ClueWeb corpus which was obtained by scraping the web. To reduce noise, we add the top K most frequent discovered SVO path types, where K is a tunable parameter. By SVO path type we refer to a set of ordered verbs mined from the SVO corpus. There is a possibility that the bridging entities, extracted from the corpus, may be present in the KB. If the bridging entity matches any alias, then it is treated as an alias to an existing KB entity. If not, then the bridging entity is added to the KB as a new entity. To avoid overfitting we add negative data to the training set. Furthermore, only high quality expressive bridging entities result in meaningful and discriminative paths. Although the quality of bridging entities depend on the corpus, low quality bridging entities can be filtered out by adding negative training data. Low quality bridging entities connect source target pairs from both positive and negative training sets, and hence are eliminated by the sparse logistic regression classifier. The negative dataset is generated using the closed world assumption by performing a random walk.

After augmenting the KB, we run the training phase of the PRA algorithm to obtain the feature (path) weights computed by the logistic regression

KB Relations	PRA	PRA-SVO	PRA-VS	PRA-ODA
actorstarredinmovie	0.0	**1.0**	**1.0**	**1.0**
atheleteplaysforteam	1.0	1.0	1.0	1.0
citylocatedincountry	0.166	0.25	**1.0**	**1.0**
journalistwritesfor publication	1.0	1.0	1.0	1.0
riverflowsthroughcity	0.333	0.25	**1.0**	**1.0**
sportsteamposition forsport	1.0	1.0	1.0	1.0
stadiumlocatedincity	1.0	1.0	1.0	1.0
statehaslake	0.0	0.0	0.0	0.0
teamplaysinleague	1.0	1.0	1.0	1.0
writerwrotebook	1.0	1.0	1.0	1.0
Average (MRR)	0.649	0.75	**0.9**	**0.9**

Table 2: Comparison of Mean Reciprocal Rank (MRR) metric for 10 relations from NELL (higher is better). PRA-SVO, PRA-VS are the systems proposed in (Gardner et al., 2013; Gardner et al., 2014). PRA-ODA is the approach proposed in this paper. Improvements in PRA-ODA over PRA-SVO is statistically significant with $p < 0.007$, with PRA-SVO as null hypothesis.

classifier.

Query Time: The set of target entities corresponding to a source entity and the relation being predicted is not available during query (test) time. We use all the entities included in the range of the relation being predicted as candidate target entities. For example, if the relation is *riverFlowsThroughCity*, the candidate target set would include entities in the KB that are *cities*. The DFS is now performed starting from source entities as during training, but this time only restricting to paths with positive weights learned during training. Any path (along with bridging entities) found during this search are added to the KB, and the PRA algorithm is now run over this augmented graph.

4 Experiments

We used the implementation of PRA provided by the authors of (Gardner et al., 2014). For our experiments, we used the same 10 NELL relation data as used in (Gardner et al., 2014). The augmentation resulted in the addition of 1086 paths during training and 1430 paths during test time.

We split the NELL data into 60% training data, 15 % development data and 25% test data. Values for d_{max}, and K, the most frequent paths, were obtained by tuning on a development set for 4 relations (athleteplaysforsport,actorstarredinmovie,citylocatedincountry

Timings (seconds)	PRA	PRA-SVO	PRA-VS	PRA-ODA
Training	635.6	574.5	564.2	913.3
Test	354.3	322.0	301.2	436.7
Batch augmentation	n/a	797	797	n/a
Embedding computation	n/a	n/a	812	n/a
Total Time	989.9	1693.5	2474.4	1350

Table 3: Runtime comparison for the entire experiment (lower is better). PRA-SVO, PRA-VS are the systems proposed in (Gardner et al., 2013; Gardner et al., 2014). PRA-ODA is the approach proposed in this paper. Between the two top performing systems, i.e., PRA-ODA and PRA-VS, PRA-ODA is faster by a factor of 1.8.

and journalistwritesforpublication). The hyperparameter values $d_{max} = 2$, $K = 10$ reported the highest MRR and were used for the rest of the relations. For the L_1 and L_2 regularization parameters in the logistic regression classifier, we used the same values as used in (Gardner et al., 2013; Gardner et al., 2014), viz., $L_1 = 0.005$, and $L_2 = 1.0$. This is because the parameters were reported to be robust, and seemed to work well even when the knowledge base was augmented.

We compare the results (PRA-ODA) with the PRA algorithm executed on the NELL KB, NELL KB augmented with surface relations (PRA-SVO) (Gardner et al., 2013) and vector space random walk PRA (PRA-VS) (Gardner et al., 2014). The run times, i.e, the time taken to perform an entire experiment for PRA-SVO and PRA-VS includes the time taken to augment NELL KB with SVO edges. The PRA-VS runtime also includes the time taken for generating embeddings to perform the vector space random walk. As can be seen from Table 2 and Table 3, our scheme, PRA-ODA, provides performance equivalent to PRA-VS with faster running time (speed up of 1.8). In addition to the time taken for the full SVO augmentation, PRA-VS takes additional time to generate embeddings (13 minutes) from the added verbs. We note that the batch augmentation in case of PRA-SVO and PRA-VS, and embedding computation in case of PRA-VS are all specific to the relations in the evaluation set, and hence can't be ignored as a one-time offline cost. In other words, these costs are likely to increase as more relations (and their instances) are included during training and testing. Runtime gains with PRA-ODA are likely to be even more pronounced in such settings.

An additional advantage of the proposed algorithm is that it can also be run on the top of any PRA based algorithm such as the PRA-SVO and PRA-VS.

5 Conclusion

In this paper, we investigated the usefulness of adding paths to a Knowledge Base for improving its connectivity by mining *bridging entities* from an external corpus. While previous KB augmentation methods focused only on augmentation using mined surface verbs while keeping the node set fixed, we extended these approaches by also adding *bridging entities* in an online fashion. We used a large corpus of 500 million web text corpus to mine these additional edges and bridging entities. Through experiments on real-world datasets, we demonstrate that the proposed approach is not only comparable or better than other state-of-the-art baselines, but more importantly provides faster overall runtime compared with the alternatives.

Acknowledgment

This work is supported in part by a gift from Google.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Sergey Brin. 1999. Extracting patterns and relations from the world wide web. In Paolo Atzeni, Alberto Mendelzon, and Giansalvatore Mecca, editors, *The World Wide Web and Databases*, volume 1590 of *Lecture Notes in Computer Science*, pages 172–183. Springer Berlin Heidelberg.

J. Callan, M. Hoy, C. Yoo, and L. Zhao. 2009. Clueweb09 data set. boston.lti.cs.cmu.edu.

Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2004. Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, pages 100–110, New York, NY, USA. ACM.

Matt Gardner, Partha Pratim Talukdar, Bryan Kisiel, and Tom Mitchell. 2013. Improving learning and inference in a large knowledge-base using latent syntactic cues. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 833–838.

Matt Gardner, Partha Pratim Talukdar, Jayant Krishnamurthy, and Tom Mitchell. 2014. Incorporating vector space similarity in random walk inference over knowledge bases. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 397–406.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, COLING '92, pages 539–545, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ni Lao and William W. Cohen. 2010. Relational retrieval using a combination ofpath constrained random walks. *Machine Learning*, 81(1):53–67.

Ni Lao, Tom Mitchell, and William W. Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 529–539, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ni Lao, Amarnag Subramanya, Fernando Pereira, and William W. Cohen. 2012. Reading the web with learned syntactic-semantic inference rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1017–1026, Stroudsburg, PA, USA. Association for Computational Linguistics.

T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. 2015. Never-ending learning. In *Proceedings of AAAI*.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(02):95–135.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. 2014. Knowledge base completion via search-based question answering. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 515–526, New York, NY, USA. ACM.