Graph-Based Seed Set Expansion for Relation Extraction Using Random Walk Hitting Times

Joel Lang
University of Geneva
7 Route de Drize
1227 Carouge, Switzerland
joel.lang@unige.ch

James Henderson
Xerox Research Centre Europe
6 Chemin de Maupertuis
38240 Meylan, France
james.henderson@xrce.xerox.com

Abstract

Iterative bootstrapping methods are widely employed for relation extraction, especially because they require only a small amount of human supervision. Unfortunately, a phenomenon known as *semantic drift* can affect the accuracy of iterative bootstrapping and lead to poor extractions. This paper proposes an alternative bootstrapping method, which ranks relation tuples by measuring their distance to the seed tuples in a bipartite tuple-pattern graph. In contrast to previous bootstrapping methods, our method is not susceptible to semantic drift, and it empirically results in better extractions than iterative methods.

1 Introduction

The goal of relation extraction is to extract tuples of a particular relation from a corpus of natural language text. A widely employed approach to relation extraction is based on iterative bootstrapping (Brin, 1998; Agichtein and Gravano, 2000; Pasca et al., 2006; Pantel and Pennacchiotti, 2006), which can be applied with only small amounts of supervision and which scales well to very large datasets.

A well-known problem with iterative bootstrapping is a phenomenon known as *semantic drift* (Curran et al., 2007): as bootstrapping proceeds it is likely that unreliable patterns will lead to false extractions. These extraction errors are amplified in the following iterations and the extracted relation will drift away from the intended target. Semantic drift often results in low precision extractions and therefore poses a major limitation of iterative bootstrapping algorithms. Previous work on iterative bootstrapping has addressed the issue of reducing semantic drift for example by bagging the results of various runs employing differing seed tuples, constructing filters which identify false tuples or patterns and adding further constraints to the bootstrapping process (T. McIntosh, 2010; McIntosh and Curran, 2009; Curran et al., 2007).

However, the analysis of Komachi et al. (2008) has shown that semantic drift is an inherent property of iterative bootstrapping algorithms and therefore poses a fundamental problem. They have shown that iterative bootstrapping without pruning corresponds to an eigenvector computation and thus as the number of iterations increases the resulting ranking will always converge towards the same static ranking of tuples, *regardless of the particular choice of seed instances*.

In this paper, we describe an alternative method, that is not susceptible to semantic drift. We represent our data as a bipartite graph, whose vertices correspond to patterns and tuples respectively and whose edges capture cooccurrences and then measure the distance of a tuple to the seed set in terms of random walk hitting times. Experimental results confirm that semantic drift is avoided by our method and show that substantial improvements over iterative forms of bootstrapping are possible.

Proceedings of NAACL-HLT 2013, pages 772–776,
Atlanta, Georgia, 9–14 June 2013.

2 Scoring with Hitting Times

From a given corpus, we extract a dataset consisting of *tuples* and *patterns*. Tuples are pairs of co-occurring strings in the corpus, such as *(Bill Gates, Microsoft)*, which potentially belong to a particular relation of interest. In our case, patterns are simply the sequence of tokens occurring between tuple elements, e.g. *"is the founder of"*. We represent all the tuple types[1] X and all the extraction pattern types Y contained in a given corpus through an undirected, weighted, bipartite graph $G = (V, E)$ with vertices $V = X \cup Y$ and edges $E \subset X \times Y$, where an edge $(x, y) \in E$ indicates that tuple x occurrs with pattern y somewhere in the corpus. Edge weights are defined through a weight matrix W which holds the weight $W_{i,j} = w(v_i, v_j)$ for edges $(v_i, v_j) \in E$. Specifically, we use the count of how many times a tuple occurs with a pattern in the corpus and weights for unconnected vertices are zero.

Our goal is to compute a score vector σ holding a score $\sigma_i = \sigma(x_i)$ for each tuple $x_i \in X$, which quantifies how well the tuple matches the seed tuples. Higher scores indicate that the tuple is more likely to belong to the relation defined through the seeds and thus the score vector effectively provides a ranking of the tuples.

We define scores of tuples based on their *distance*[2] to the seed tuples in the graph. The distance of some tuple x to the seed set S can be naturally formalized in terms of the average time it takes until a random walk starting in S reaches x, the *hitting time*. The random walk is defined through the probability distribution over start vertices and through a matrix of transition probabilities. Edge weights are constrained to be non-negative, which allows us to define the transition matrix P with $P_{i,j} = p(v_j|v_i) = \frac{1}{d_{v_i}} w(v_i, v_j)$, where $d_v = \sum_{v_k \in V} w(v, v_k)$ is the degree of a vertex $v \in V$.

The distance of two vertices is measured in terms of the average time of a random walk between the two. Specifically, we adopt the notion of *T-truncated hitting time* (Sarkar and Moore, 2007) defined as the expected number of steps it takes until a random walk of at most T steps starting at v_i reaches v_j for the first time:

$$h^T(v_j|v_i) = \begin{cases} 0 & \text{iff. } v_j = v_i \text{ or T=0} \\ 1 + \sum_{v_k \in V} p(v_k|v_i) h^{T-1}(v_j|v_k) \end{cases}$$

The truncated hitting time $h^T(v_j|v_i)$ can be approximately computed by sampling M independent random walks starting at v_i of length T and computing

$$\hat{h}^T(v_j|v_i) = \frac{1}{M} \sum_{k=1}^{m} t_k + (1 - \frac{m}{M})T \qquad (1)$$

where $\{t_1 \dots t_m\}$ are the sampled first-hit times of random walks which reach v_j within T steps (Sarkar et al., 2008).

The score $\sigma_{HT}(v)$ of a vertex $v \notin S$ to the seed set S is then defined as the inverse of the average T-truncated hitting time of random walks starting at a randomly chosen vertex $s \in S$:

$$\frac{1}{\sigma_{HT}(v)} = h^T(v|S) = \frac{1}{|S|} \sum_{s \in S} h^T(v|s) \qquad (2)$$

3 Experiments

We extracted tuples and patterns from the fifth edition of the Gigaword corpus (Parker et al., 2011), by running a named entity tagger and extracting all pairs of named entities and extracting occurring within the same sentence which do not have another named entity standing between them. Gold standard seed and test tuples for a set of relations were obtained from YAGO (Suchanek et al., 2007). Specifically, we took all relations for which there are at least 300 tuples, each of which occurs at least once in the corpus. This resulted in the set of relations shown in Table 1, plus the development relation *hasWonPrize*.

For evaluation, we use the *percentile rank of the median test set element* (PRM, see Francois et al. 2007), which reflects the quality of the

[1] Note that we are using tuple and pattern *types* rather than particular mentions in the corpus.

[2] The term is used informally. In particular, hitting times are not a *distance metric*, since they can be asymmetric.

full produced ranking, not just the top N elements and is furthermore computable with only a small set of labeled test tuples [3].

We compare our proposed method based on hitting times (HT) with two variants of iterative bootstrapping. The first one (IB1) does not employ pruning and corresponds to the algorithm described in Komachi et al. (2008). The second one (IB2) corresponds to a standard bootstrapping algorithm which employs pruning after each step in order to reduce semantic drift. Specifically, scores are pruned after projecting from X onto Y and from Y onto X, retaining only the top $N^{(t)} = N_0 t$ scores at iteration t and setting all other scores to zero.

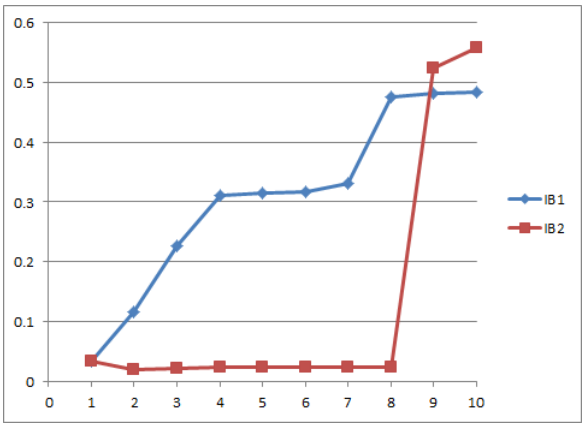

Figure 1: PRM for iterative bootstrapping without pruning (IB1) and with pruning (IB2). A lower PRM is better.

3.1 Parametrizations

The experiments in this section were conducted on the held out development relation *hasWonPrize*. The ranking produced by both forms of iterative bootstrapping IB1 and IB2 depend on the number of iterations, as shown in Figure 1. IB1 achieves an optimal ranking after just one iteration and thereafter scores get worse due to semantic drift. In contrast, pruning helps avoid semantic drift for IB2, which attains an optimal score after 2 iterations and achieves relatively constant scores for several iterations. However, during iteration 9 an incorrect pattern is kept and this at once leads to a drastic loss in accuracy, showing that semantic drift is only deferred and not completely eliminated.

Our method HT has parameter T, corresponding to the truncation time, i.e., maximal number of steps of a random walk. Figure 2 shows the PRM of our method for different values of T. Performance gets better as T increases and is optimal for $T = 12$, whereas for larger values, the performance gets slightly worse again. The figure shows that, if T is large enough (> 5), the PRM is relatively constant and there is no phenomenon comparable to semantic drift, which causes instability in the produced rankings.

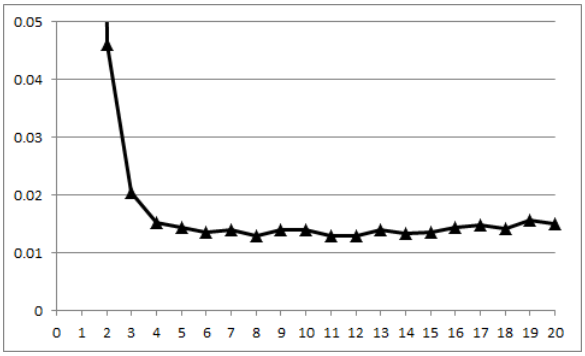

Figure 2: PRM for our method based on hitting times, for different values of the truncation time parameter T.

3.2 Method Comparison

To evaluate the methods, firstly the parameters for each method were set to the optimal values as determined in the previous section. For the experiments here, we again use 200 randomly chosen tuples as the seeds for each relation. All the remaining gold standard tuples are used for testing.

Table 1 shows the PRM for the three methods. For a majority of the relations (12/16) HT attains the best, i.e. lowest, PRM, which confirms that hitting times constitute an accurate way of measuring the distance of tuples to the seed set. IB1 and IB2 each perform best on 2/16 of the relations. A sign test on these results yields that

[3] other common metrics do not satisfy these conditions.

Relation	IB1	IB2	HT
created	1.82	1.71	**0.803**
dealsWith	**0.0262**	0.107	0.0481
diedIn	30.5	**18.4**	20.4
directed	0.171	0.238	**0.166**
hasChild	7.66	32.2	**4.26**
influences	5.93	**5.48**	6.60
isAffiliatedTo	1.54	2.01	**1.30**
isCitizenOf	1.74	1.87	**1.68**
isLeaderOf	1.37	1.91	**0.401**
isMarriedTo	4.69	4.14	**1.27**
isPoliticianOf	**0.0117**	0.110	0.0409
livesIn	3.17	2.48	**1.70**
owns	11.0	2.10	**2.07**
produced	1.55	0.967	**0.240**
wasBornIn	11.3	9.37	**8.42**
worksAt	1.52	2.21	**0.193**

Table 1: PRM in percent for all relations, for all three models. A lower PRM corresponds to a better model, with the best score indicated in bold.

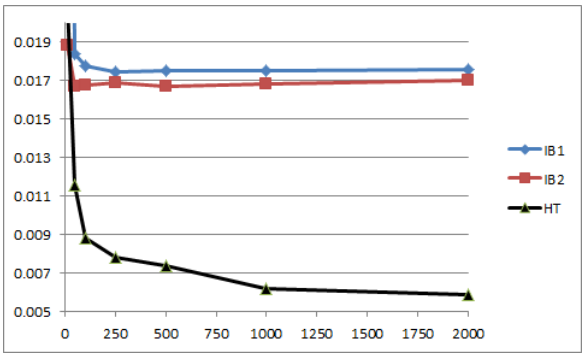

Figure 3: PRM for the three methods, as a function of the size of the seed set for the relation created.

HT is better than both IB1 and IB2 at significance level $\alpha < 0.01$.

Moreover, the ranking produced by HT is stable and not affected by semantic drift, given that even where results are worse than for IB1 or IB2, they are still close to the best performing method. In contrast, when semantic drift occurs, the performance of IB1 and IB2 can deteriorate drastically, e.g. for the *worksAt* relation, where both IB1 and IB2 produce rankings that are a lot worse than the one produced by HT.

3.3 Sensitivity to Seed Set Size

Figure 3 shows the PRM for each of the three methods as a function of the size of the seed set for the relation *created*. For small seed sets, the performance of the iterative methods can be increased by adding more seeds. However, from a seed set size of 50 onwards, performance remains relatively constant. In other words, iterative bootstrapping is not benefitting from the information provided by the additional labeled data, and thus has a poor learning performance. In contrast, for our method based on hitting times, the performance continually improves as the seed set size is increased. Thus, also in terms of learning performance, our method is more sound than iterative bootstrapping.

4 Conclusions

The paper has presented a graph-based method for seed set expansion which is not susceptible to semantic drift and on most relations outperforms iterative bootstrapping. The method measures distance between vertices through random walk hitting times. One property which makes hitting times an appropriate distance measure is their ability to reflect the overall connectivity structure of the graph, in contrast to measures such as the shortest path between two vertices. The hitting time will decrease when the number of paths from the start vertex to the target vertex increases, when the length of paths decreases or when the likelihood (weights) of paths increases. These properties are particularly important when the observed graph edges must be assumed to be merely a sample of all plausible edges, possibly perturbated by noise. This has also been asserted by previous work, which has shown that hitting times successfully capture the notion of similarity for other natural language processing problems such as learning paraphrases (Kok and Brockett, 2010) and related problems such as query suggestion (Mei et al., 2008). Future work will be aimed towards employing our hitting time based method in combination with a richer feature set.

References

Agichtein, E. and Gravano, L. (2000). Snowball: Extracting Relations from Large Plain-text Collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries*.

Brin, S. (1998). Extracting Patterns and Relations from the World-Wide Web. In *Proceedings of the 1998 International Workshop on the Web and Databases*.

Curran, J., Murphy, T., and Scholz, B. (2007). Minimising Semantic Drift with Mutual Exclusion Bootstrapping. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*.

Francois, F., Pirotte, A., Renders, J., and Saerens, M. (2007). Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355 –369.

Kok, S. and Brockett, C. (2010). Hitting the Right Paraphrases in Good Time. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Komachi, M., Kudo, T., Shimbo, M., and Matsumoto, Y. (2008). Graph-based Analysis of Semantic Drift in Espresso-like Bootstrapping Algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

McIntosh, T. and Curran, J. (2009). Reducing Semantic Drift with Bagging and Distributional Similarity. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL*.

Mei, Q., Zhou, D., and Church, K. (2008). Query Suggestion Using Hitting Time. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*.

Pantel, P. and Pennacchiotti, M. (2006). Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*.

Parker, R., Graff, D., Kong, J., Chen, K., and Maeda, K. (2011). English Gigaword Fifth Edition. Technical report, Linguistic Data Consortium.

Pasca, M., Lin, D., Bigham, J., Lifchits, A., and Jain, A. (2006). Organizing and Searching the World Wide Web of Facts – Step One: the One-million Fact Extraction Challenge. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*.

Sarkar, P. and Moore, A. (2007). A Tractable Approach to Finding Closest Truncated-commute-time Neighbors in Large Graphs. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*.

Sarkar, P., Moore, A., and Prakash, A. (2008). Fast Incremental Proximity Search in Large Graphs. In *Proceedings of the 25th International Conference on Machine Learning*.

Suchanek, F., Kasneci, G., and Weikum, G. (2007). Yago: A Core of Semantic Knowledge. In *Proceedings of the International World Wide Web Conference (WWW)*.

T. McIntosh (2010). Unsupervised Discovery of Negative Categories in Lexicon Bootstrapping. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.