Reading The Web with Learned Syntactic-Semantic Inference Rules

Ni Lao[1]*, **Amarnag Subramanya**[2], **Fernando Pereira**[2], **William W. Cohen**[1]
[1]Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA
[2]Google Research, 1600 Amphitheatre Parkway, Mountain View, CA 94043, USA
nlao@cs.cmu.edu, {asubram, pereira}@google.com, wcohen@cs.cmu.edu

Abstract

We study how to extend a large knowledge base (Freebase) by reading relational information from a large Web text corpus. Previous studies on extracting relational knowledge from text show the potential of syntactic patterns for extraction, but they do not exploit background knowledge of other relations in the knowledge base. We describe a distributed, Web-scale implementation of a path-constrained random walk model that learns syntactic-semantic inference rules for binary relations from a graph representation of the parsed text and the knowledge base. Experiments show significant accuracy improvements in binary relation prediction over methods that consider only text, or only the existing knowledge base.

1 Introduction

Manually-created knowledge bases (KBs) often lack basic information about some entities and their relationships, either because the information was missing in the initial sources used to create the KB, or because human curators were not confident about the status of some putative fact, and so they excluded it from the KB. For instance, as we will see in more detail later, many person entries in Freebase (Bollacker et al., 2008) lack nationality information. To fill those KB gaps, we might use general rules, ideally automatically learned, such as "if *person* was born in *town* and *town* is in *country* then the *person* is a national of the *country*." Of course, rules like this may be defeasible, in this case for example because of naturalization or political changes. Nevertheless, many such imperfect rules can be learned and combined to yield useful KB completions, as demonstrated in particular with the *Path-Ranking Algorithm* (PRA) (Lao and Cohen, 2010; Lao et al., 2011), which learns such rules on heterogenous graphs for link prediction tasks.

Alternatively, we may attempt to fill KB gaps by applying relation extraction rules to free text. For instance, Snow et al. (2005) and Suchanek et al. (2006) showed the value of syntactic patterns in extracting specific relations. In those approaches, KB tuples of the relation to be extracted serve as positive training examples to the extraction rule induction algorithm. However, the KB contains much more knowledge about other relations that could potentially be helpful in improving relation extraction accuracy and coverage, but that is not used in such purely text-based approaches.

In this work, we use PRA to learn weighted rules (represented as graph path patterns) that combine both semantic (KB) and syntactic information encoded respectively as edges in a graph-structured KB, and as syntactic dependency edges in dependency-parsed Web text. Our approach can easily incorporate existing knowledge in extraction tasks, and its distributed implementation scales to the whole of the Freebase KB and 60 million parsed documents. To the best of our knowledge, this is the first successful attempt to apply relational learning methods to heterogeneous data with this scale.

*This research was carried out during an internship at Google Research

Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 1017–1026, Jeju Island, Korea, 12–14 July 2012.

1.1 Terminology and Notation

In this study, we use a simplified KB consisting of a set C of concepts and a set R of labels. Each label r denotes some binary relation partially represented in the KB. The concrete KB is a directed, edge-labeled graph $G = (C, T)$ where $T \subseteq C \times R \times C$ is the set of labeled edges (also known as *triples*) (c, r, c'). Each triple represents an instance $r(c, c')$ of the relation $r \in R$. The KB may be incomplete, that is, $r(c, c')$ holds in the real world but $(c, r, c') \notin T$. Our method will attempt to learn rules to infer such missing relation instances by combining the KB with parsed text.

We denote by r^{-1} the inverse relation of r: $r(c, c') \Leftrightarrow r^{-1}(c', c)$. For instance $Parent^{-1}$ is equivalent to *Children*. It is convenient to take G as containing triple (c', r^{-1}, c) whenever it contains triple (c, r, c').

A *path type* in G is a sequence $\pi = \langle r_1, \ldots, r_m \rangle$. An *instance* of the path type is a sequence of nodes $c_0, \ldots, c_m$ such that $r_i(c_{i-1}, c_i)$. For instance, "the persons who were born in the same town as the query person", and "the nationalities of persons who were born in the same town as the query person" can be reached respectively through paths matching the following types

$$\pi_1 : \langle BornIn, BornIn^{-1} \rangle$$
$$\pi_2 : \langle BornIn, BornIn^{-1}, Nationality \rangle$$

1.2 Learning Syntactic-Semantic Rules with Path-Constrained Random Walks

Given a query concept $s \in C$ and a relation $r \in R$, PRA begins by enumerating a large set of bounded-length path types. These path types are treated as ranking "experts," each generating some random instance of the path type starting from s, and ranking end nodes t by their weights in the resulting distribution. Finally, PRA combines the weights contributed by different "experts" by using logistic regression to predict the probability that the relation $r(s, t)$ holds.

In this study, we test the hypothesis that PRA can be used to find useful "syntactic-semantic patterns" – that is, patterns that exploit both semantic and syntactic relationships, thereby using semantic knowledge as background in interpreting syntactic

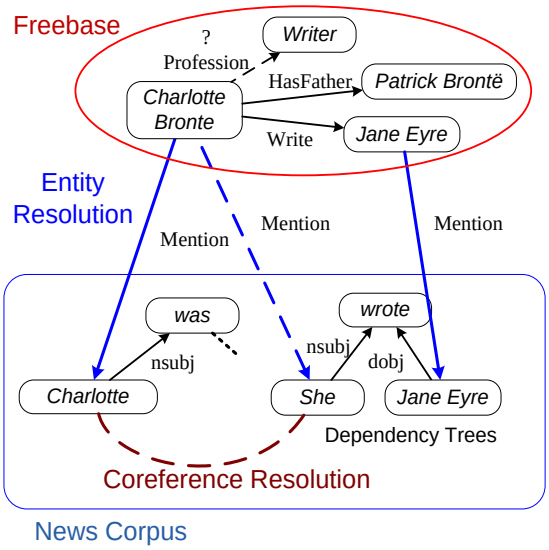

Figure 1: Knowledge base and parsed text as a labeled graph. For clarity, some word nodes are omitted.

relationships. As shown in Figure 1, we extend the KB graph G with nodes and edges from text that has been syntactically analyzed with a dependency parser[1] and where pronouns and other anaphoric referring expressions have been clustered with their antecedents. The text nodes are word/phrase instances, and the edges are syntactic dependencies labeled by the corresponding dependency type. Mentions of entities in the text are linked to KB concepts by mention edges created by an entity resolution process.

Given for instance the query $Profession(CharlotteBronte, ?)$, PRA produces a ranked list of answers that may have the relation *Profession* with the query node *CharlotteBronte*. The features used to score answers are the random walk probabilities of reaching a certain profession node from the query node by paths with particular path types. PRA can learn path types that combine background knowledge in the database with syntactic patterns in the text corpus. We now exemplify some path types involving relations described in Table 3. Type $\langle M, conj, M^{-1}, Profession \rangle$ is *active* (matches paths) for professions of persons who are mentioned in conjunction with the query person as in "collaboration between McDougall and Simon

[1] Stanford dependencies (de Marneffe and Manning, 2008).

Philips". For a somewhat subtler example, type $\langle M, TW, CW^{-1}, Profession^{-1}, Profession \rangle$ is active for persons who are mentioned by their titles as in "President Barack Obama". The type subsequence $\langle Profession^{-1}, Profession \rangle$ ensures that only profession concepts are activated. The features generated from these path types combine syntactic dependency relations (*conj*) and textual information relations (*TW* and *CW*) with semantic relations in the KB (*Profession*).

Experiments on three Freebase relations (profession, nationality and parents) show that exploiting existing background knowledge as path features can significantly improve the quality of extraction compared with using either Freebase or the text corpus alone.

1.3 Related Work

Information extraction from varied unstructured and structured sources involves both complex relational structure and uncertainty at all levels of the extraction process. Statistical Relational Learning (SRL) seeks to combine statistical and relational learning methods to address such tasks. However, most SRL approaches (Friedman et al., 1999; Richardson and Domingos, 2006) suffer the complexity of inference and learning when applied to large scale problems. Recently, Lao and Cohen (2010) introduced Path Ranking algorithm, which is applicable to larger scale problems such as literature recommendation (Lao and Cohen, 2010) and inference on a large knowledge base (Lao et al., 2011).

Much of the previous work on automatic relation extraction was based on certain lexico-syntactic patterns. Hearst (1992) first noticed that patterns such as "NP and other NP" and "NP such as NP" often imply hyponym relations (NP here refers to a noun phrase). However, such approaches to relation extraction are limited by the availability of domain knowledge. Later systems for extracting arbitrary relations from text mostly use shallow surface text patterns (Etzioni et al., 2004; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002). The idea of using sequences of dependency edges as features for relation extraction was explored by Snow et al. (2005) and Suchanek et al. (2006). They define features to be shortest paths on dependency trees which connect pairs of NP candidates.

This study is most closely related to work of Mintz et al. (2009), who also study the problem of extending Freebase with extraction from parsed text. As in our work, they use a logistic regression model with path features. However, their approach does not exploit existing knowledge in the KB. Furthermore, their path patterns are used as binary-values features. We show experimentally that fractional-valued features generated by random walks provide much higher accuracy than binary-valued ones.

Culotta et al. (2006)'s work is similar to our approach in the sense of relation extraction by discovering relational patterns. However while they focus on identifying relation mentions in text (microreading),this work attempts to infer new tuples by gathering path evidence over the whole corpus (macroreading). In addition, their work involves a few thousand examples, while we aim for Web-scale extraction.

Do and Roth (2010) use a KB (YAGO) to aid the generation of features from free text. However their method is designed specifically for extracting hierarchical taxonomic structures, while our algorithm can be used to discover relations for general general graph-based KBs.

In this paper we extend the PRA algorithm along two dimensions: combining syntactic and semantic cues in text with existing knowledge in the KB; and a distributed implementation of the learning and inference algorithms that works at Web scale.

2 Path Ranking Algorithm

We briefly review the Path Ranking algorithm (PRA), described in more detail by Lao and Cohen (2010). Each path type $\pi = \langle r_1, r_2, ..., r_\ell \rangle$ specifies a real-valued *feature*. For a given query-answer node pair (s, t), the value of the feature π is $P(s \rightarrow t; \pi)$, the probability of reaching t from s by a random walk that instantiates the type. More specifically, suppose that the random walk has just reached v_i by traversing edges labeled $r_1, \ldots, r_i$ with s=v_0. Then v_{i+1} is drawn at random from all nodes reachable from v_i by edges labeled r_{i+1}. A path type π is *active* for pair (s, t) if $P(s \rightarrow t; \pi) > 0$.

Let $B = \{\perp, \pi_1, ..., \pi_n\}$ be the set of all path types of length no greater than ℓ that occur in the graph together with the dummy type $\perp$, which

represents the bias feature. For convenience, we set $P(s \to t; \perp) = 1$ for any nodes s, t. The score for whether query node s is related to another node t by relation r is given by

$$score(s,t) = \sum_{\pi \in B} P(s \to t; \pi)\theta_\pi \quad ,$$

where θ_π is the weight of feature π. The model parameters to be learned are the vector $\boldsymbol{\theta} = \langle \theta_\pi \rangle_{\pi \in B}$. The procedures used to discover B and estimate $\boldsymbol{\theta}$ are described in the following. Finally, note that we train a *separate* PRA model for each relation r.

Path Discovery: Given a graph and a target relation r, the total number of path types is an exponential function of the maximum path length ℓ and considering all possible paths would be computationally very expensive. As a result, B is constructed using only path types that satisfy the following two constraints:

1. the path type is active for more than K training query nodes, and
2. the probability of reaching any correct target node t is larger than a threshold α on average for the training query nodes s.

We will discuss how K, α and the training queries are chosen in Section 5. In addition to making the training more efficient, these constraints are also helpful in removing low quality path types.

Training Examples: For each relation r of interest, we start with a set of node pairs $S_r = \{(s_i, t_i)\}$. From S_r, we create the training set $D_r = \{(\mathbf{x}_i, y_i)\}$, where $\mathbf{x}_i = \langle P(s_i \to t_i; \pi) \rangle_{\pi \in B}$ is the vector of path feature values for the pair (s_i, t_i), and y_i indicates whether $r(s_i, t_i)$ holds.

Following previous work (Lao and Cohen, 2010; Mintz et al., 2009), node pairs that are in r in the KB are legitimate positive training examples[2]. One can generate negative training examples by considering all possible pairs of concepts whose type is compatible with r (as given by the schema) and are not present in the KB. However this procedure leads to a very large number of negative examples (e.g., for the parents relation, any pair of person concepts which are related by this relation would be valid negative examples) which not only makes training very expensive but also introduces an incorrect bias in the training set. Following Lao and Cohen (2010) we use a simple biased sampling procedure to generate negative examples: first, the path types discovered in the previous (path discovery) step are used to construct an initial PRA model (all feature weights are set to 1.0); then, for each query node s_i, this model is used to retrieve candidate answer nodes, which are then sorted in descending order by their scores; finally, nodes at the $k(k+1)/2$-th positions are selected as negative samples, where $k = 0, 1, 2, ...$

Logistic Regression Training: Given a training set D, we estimate parameters $\boldsymbol{\theta}$ by maximizing the following objective

$$\mathcal{F}(\boldsymbol{\theta}) = \frac{1}{|D|} \sum_{(\mathbf{x},y) \in D} f(\mathbf{x}, y; \boldsymbol{\theta}) - \lambda_1 \|\boldsymbol{\theta}\|_1 - \lambda_2 \|\boldsymbol{\theta}\|_2^2$$

where λ_1 and λ_2 control the strength of the L_1-regularization which helps with structure selection and L_2^2-regularization which prevents overfitting. The log-likelihood $f(\mathbf{x}, y; \boldsymbol{\theta})$ of example $(\mathbf{x}, y)$ is given by

$$f(\mathbf{x}, y, \boldsymbol{\theta}) = y \ln p(\mathbf{x}, \boldsymbol{\theta}) + (1 - y) \ln(1 - p(\mathbf{x}, \boldsymbol{\theta}))$$

$$p(\mathbf{x}, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}^T \mathbf{x})}{1 + \exp(\theta^T \mathbf{x})} \quad .$$

Inference: After a model is trained for a relation r in the knowledge base, it can be used to produce new instances of r. We first generate unlabeled queries s which belong to the domain of r. Queries which appear in the training set are excluded. For each unlabeled query node s, we apply the trained PRA model to generate a list of candidate t nodes together with their scores. We then sort all the predictions (s, t) by their scores in descending order, and evaluate the top ones.

3 Extending PRA

As described in the previous section, the PRA model is trained on positive and negative queries generated from the KB. As Freebase contains millions of

[2] In our experiments we subsample the positive examples. See section 3.2 for more details.

concepts and edges, training on all the generated queries is computationally challenging. Further, we extend the Freebase graph with parse paths of mentions of concepts in Freebase in millions of Web pages. Yet another issue is that the training queries generated using Freebase are inherently biased towards the distribution of concepts in Freebase and may not reflect the distribution of mentions of these concepts in text data. As one of the goals of our approach is to learn relation instances that are missing in Freebase, training on such a set biased towards the distribution of concepts in Freebase may not lead to good performance. In this section we explain how we modified the PRA algorithm to address those issues.

3.1 Scaling Up

Most relations in Freebase have a large set of existing triples. For example, for the *profession* relation, there are around 2 million persons in Freebase, and about 0.3 million of them have known professions. This results in more than 0.3 million training queries (persons), each with one or more positive answers (professions), and many negative answers, which make training computationally challenging. Generating all the paths for millions of queries over a graph with millions of concepts and edges further complicates the computational issues. Incorporating the parse path features from the text only exacerbates the matter. Finally once we have trained a PRA model for a given relation, say *profession*, we would like to infer the professions for all the 1.7 million persons whose professions are not known to Freebase (and possibly predict changes to the profession information of the 0.3 million people whose professions were given).

We use distributed computing to deal with the large number of training and prediction queries over a large graph. A key observation is that the different stages of the PRA algorithm are based on independent computations involving individual queries. Therefore, we can use the MapReduce framework to distribute the computation (Dean and Ghemawat, 2008). For path discovery, we modify Lao et al.'s path finding (2011) approach to decouple the queries: instead of using one depth-first search that involves all the queries, we first find all paths up to certain length for each query node in the map stage, and then collect statistics for each path from all the query nodes in the reduce stage. We used a 500-machine, 8GB/machine cluster for these computations.

Another challenge associated with applying PRA to a graph constructed using a large amounts of text is that we cannot load the entire graph on a single machine. To circumvent this problem, we first index all parsed sentences by the concepts that they mention. Therefore, to perform a random walk for a query concept s, we only load the sentences which mention s.

3.2 Sampling Training Data

Using the r-edges in the KB as positive examples distorts the training set. For example, for the *profession* relation, there are 0.3 million persons for whom Freebase has profession information, and amongst these 0.24 million are either politicians or actors. This may not reflect the distribution of professions of persons mentioned in Web data. Using all of these as training queries will most certainly bias the trained model towards these professions as PRA is trained discriminatively. In other words, training directly with this data would lead to a model that is more likely to predict professions that are popular in Freebase. To avoid this distortion, we use stratified sampling. For each relation r and concept $t \in C$, we count the number of r edges pointing to t

$$N_{r,t} = |\{(s, r, t) \in T\}| \quad .$$

Given a training query (s, r, t) we sample it according to

$$P_{r,t} = \min\left(1, \frac{\sqrt{m + N_{r,t}}}{N_{r,t}}\right)$$

We fix $m = 100$ in our experiments. If we take the profession relation as an example, the above implies that for popular professions, we only sample about $\sqrt{N_{r,t}}$ out of the $N_{r,t}$ possible queries that end in t, whereas for the less popular professions we would accept all the training queries.

3.3 Text Graph Construction

As we are processing Web text data (see following section for more detail), the number of mentions

of a concept follows a somewhat heavy-tailed distribution: there are a small number of very popular concepts (head) and a large number of not so popular concepts (tail). For instance the concept *BarackObama* is mentioned about 8.9 million times in our text corpus. To prevent the text graph from being dominated by the head concepts, for each sentence that mentions concept $c \in C$, we accept it as part of the text graph with probability:

$$P_c = \min\left(1, \frac{\sqrt{k + S_c}}{S_c}\right)$$

where S_c is the number of sentences in which c is mentioned in the whole corpus. In our experiments we use $k = 10^5$. This means that if $S_c \gg k$, then we only sample about $\sqrt{S_c}$ of the sentences that contain a mention of the concept, while if $S_c \ll k$, then all mentions of that concept will likely be included.

4 Datasets

We use Freebase as our knowledge base. Freebase data is harvested from many sources, including Wikipedia, AMG, and IMDB.[3] As of this writing, it contains more than 21 million concepts and 70 million labeled edges. For a large majority of concepts that appear both in Freebase and Wikipedia, Freebase maintains a link to the Wikipedia page of that concept.

We also collect a large Web corpus and identify 60 million pages that mention concepts relevant to this study. The free text on those pages are POS-tagged and dependency parsed with an accuracy comparable to that of the current Stanford dependency parser (Klein and Manning, 2003). The parser produces a dependency tree for each sentence with each edge labeled with a standard dependency tag (see Figure 1).

In each of the parsed documents, we use POS tags and dependency edges to identify potential referring noun phrases (NPs). We then use a within-document coreference resolver comparable to that of Haghighi and Klein (2009) to group referring NPs into co-referring clusters. For each cluster that contains a proper-name mention, we find the Freebase concept or concepts, if any, with a name or alias that matches the mention. If a cluster has multiple possible matching Freebase concepts, we choose a single sense based on the following simple model. For each Freebase concept $c \in C$, we compute $N(c, m)$, the number of times the concept c is referred by mention m by using both the alias information in Freebase and the anchors of the corresponding Wikipedia page for that concept. Based on $N(c, m)$ we can calculate the empirical probability $p(c|m) = N(c, m)/\sum_{c'} N(c', m)$. If u is a cluster with mention set $M(u)$ in the document, and $C(m)$ the set of concepts in KB with name or alias m, we assign u to concept $c^* = \underset{c \in C(m), m \in M(u)}{\operatorname{argmax}} p(c|m)$, provided that there exists at least one $c \in C(m)$ and $m \in M(u)$ such that $p(c|m) > 0$. Note that $M(c)$ only contains the proper-name mentions in cluster c.

[3] www.wikipedia.org, www.allmusic.com, www.imdb.com.

Table 1: Size of training and test sets for each relation.

Task	Training Set	Test Set
Profession	22,829	15,219
Nationality	14,431	9,620
Parents	21,232	14,155

5 Results

We use three relations *profession*, *nationality* and *parents* for our experiments. For each relation, we select its current set of triples in Freebase, and apply the stratified sampling (Section 3.2) to each of the three triple sets. The resulting triple sets are then randomly split into training (60% of the triples) and test (the remaining triples). However, the *parents* relation yields 350k triples after stratified sampling, so to reduce experimental effort we further randomly sub-sample 10% of that as input to the train-test split. Table 1 shows the sizes of the training and test sets for each relation.

To encourage PRA to find paths involving the text corpus, we do not count relation *M* (which connects concepts to their mentions) or M^{-1} when calculating path lengths. We use L_1/L_2^2-regularized logistic regression to learn feature weights. The PRA hyperparameters (α and K as defined in Section 2) and regularizer hyperparameters are tuned by threefold cross validation (CV) on the training set. We average the models across all the folds and choose the model that gives the best

Table 2: Mean Reciprocal Rank (MRR) for different approaches under closed-world assumption. Here KB, Text and KB+Text columns represent results obtained by training a PRA model with only the KB, only text, and both KB and text. KB+Text[b] is the binarized PRA approach trained on both KB and text. The best performing system (results shown in bold font) is significant at 0.0001 level over its nearest competitor according to a difference of proportions significance test.

Task	KB	Text	KB+Text	KB+Text[b]
Profession	0.532	0.516	**0.583**	0.453
Nationality	0.734	0.729	**0.812**	0.693
Parents	0.329	0.332	**0.392**	0.319

performance on the training set for each relation.

We report results of two evaluations. First, we evaluate the performance of the PRA algorithm when trained on a subset of existing Freebase facts and tested on the rest. Second, we had human annotators verify facts proposed by PRA that are not in Freebase.

5.1 Evaluation with Existing Knowledge

Previous work in relation extraction from parsed text (Mintz et al., 2009) has mostly used binary features to indicate whether a pattern is present in the sentences where two concepts are mentioned. To investigate the benefit of having fractional valued features generated by random walks (as in PRA), we also evaluate a *binarized PRA* approach, for which we use the same syntactic-semantic pattern features as PRA does, but binarize the feature values from PRA: if the original fractional feature value was zero, the feature value is set to zero (equivalent to not having the feature in that example), otherwise it is set to 1.

Table 2 shows a comparison of the results obtained using the PRA algorithm trained using only Freebase (**KB**), using only the text corpus graph (**Text**), trained with both Freebase and the text corpus (**KB+Text**) and the binarized PRA algorithm using both Freebase and the text corpus (**KB+Text[b]**). We report Mean Reciprocal Rank (MRR) where, given a set of queries Q,

$$\text{MRR} = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{\text{rank of } q\text{'s first correct answer}} \quad .$$

Comparing the results of first three columns we see that combining Freebase and text achieves significantly better results than using either Freebase or text alone. Further comparing the results of last two columns we also observe a significant drop in MRR for the binarized version of PRA. This clearly shows the importance of using the random walk probabilities. It can also be seen that the MRR for the parents relation is lower than those for other relations. This is mainly because there are larger number of potential answers for each query node of *Parent* relation than for each query node of the other two relations – all persons in Freebase versus all professions or nationalities. Finally, it is important to point out that our evaluations are actually *lower bounds* of actual performance, because, for instance, a person might have a profession besides the ones in Freebase and in such cases, this evaluation does not give any credit for predicting those professions — they are treated as errors. We try to address this issue with the manual evaluations in the next section.

Table 2 only reports results for the maximum path length $\ell = 4$ case. We found that shorter maximum path lengths give worse results: for instance, with $\ell = 3$ for the profession relation, MRR drops to 0.542, from 0.583 for $\ell = 4$ when using both Freebase and text. This difference is significant at the 0.0001 level according to a difference of proportions test. Further we find that using longer path length takes much longer time to train and test, but does not lead to significant improvements over the $\ell = 4$ case. For example, for profession, $\ell = 5$ gives a MRR of 0.589.

Table 3 shows the top weighted features that involve text edges for PRA models trained on both Freebase and the text corpus. To make them easier to understand, we group them based on their functionality. For the profession and nationality tasks, the conjunction dependency relation (in group 1,4) plays an important role: these features first find persons mentioned in conjunction with the query

Table 3: Top weighted path types involving text edges for each task grouped according to functionality. *M* relations connect each concept in knowledge base to its mentions in the corpus. *TW* relations connect each token in a sentence to the words in the text representation of this token. *CW* relations connect each concept in knowledge base to the words in the text representation of this concept. We use lower case names to denote dependency edges, word capitalized names to denote KB edges, and "$^{-1}$" to denote the inverse of a relation.

Profession	**Top Weighted Features**	**Comments**
1	$\langle M, conj, M^{-1}, Profession\rangle$ $\langle M, conj^{-1}, M^{-1}, Profession\rangle$	Professions of persons mentioned in conjunction with the query person: *"McDougall and Simon Phillips collaborated ..."*
2	$\langle M, TW, CW^{-1}, Profession^{-1}, Profession\rangle$	Active if a person is mentioned by his profession: *"The president said ..."*
3	$\langle M, TW, TW^{-1}, M^{-1}, Children, Profession\rangle$ $\langle M, TW, TW^{-1}, M^{-1}, Parents, Profession\rangle$ $\langle M, TW, TW^{-1}, M^{-1}, Advisors, Profession\rangle$	First find persons with similar names or mentioned in similar ways, then aggregate the professions of their children/parents/advisors: starting from the concept *BarackObama*, words such as *"Obama"*, *"leader"*, *"president"*, and *"he"* are reachable through path $\langle M, TW\rangle$
Nationality	**Top Weighted Features**	**Comments**
4	$\langle M, conj, TW, CW^{-1}, Nationality\rangle$ $\langle M, conj^{-1}, TW, CW^{-1}, Nationality\rangle$	The nationalities of persons mentioned in conjunction with the query person: *"McDougall and Simon Phillips collaborated ..."*
5	$\langle M, nc^{-1}, TW, CW^{-1}, Nationality\rangle$ $\langle M, tmod^{-1}, TW, CW^{-1}, Nationality\rangle$ $\langle M, nn, TW, CW^{-1}, Nationality\rangle$	The nationalities of persons mentioned close to the query person through other dependency relations.
6	$\langle M, poss, poss^{-1}, M^{-1}, PlaceOfBirth, ContainedBy\rangle$ $\langle M, title, title^{-1}, M^{-1}, PlaceOfDeath, ContainedBy\rangle$	The birth/death places of the query person with restrictions to different syntactic constructions.
Parents	**Top Weighted Features**	**Comments**
7	$\langle M, TW, CW^{-1}, Parents\rangle$	The parents of persons with similar names or mentioned in similar ways: starting from the concept *CharlotteBronte* words such as *"Bronte"*, *"Charlotte"*, *"Patrick"*, and *"she"* are reachable through path $\langle M, TW\rangle$.
8	$\langle M, nsubj, nsubj^{-1}, TW, CW^{-1}\rangle$ $\langle M, nsubj, nsubj^{-1}, M^{-1}, CW, CW^{-1}\rangle$ $\langle M, nc^{-1}, nc, TW, CW^{-1}\rangle$ $\langle M, TW, CW^{-1}\rangle$ $\langle M, TW, TW^{-1}, TW, CW^{-1}\rangle$	Persons with similar names or mentioned in similar ways to the query person with various restrictions or expansions. $\langle nsubj, nsubj^{-1}\rangle$ and $\langle nc^{-1}, nc\rangle$ require the query to be subject and noun compound respectively. $\langle TW^{-1}, TW\rangle$ expands further by word similarities.

person, and then find their professions or nationalities. The features in group 2 capture the fact that sometimes people are mentioned by their professions. The subpath $\langle Profession^{-1}, Profession \rangle$ ensures that only profession related concepts are activated. Features in group 3 first find persons with similar names or mentioned in similar ways to the query person, and then aggregate the professions of their children, parents, or advisors. Features in group 6 can be seen as special versions of feature $\langle PlaceOfBirth, ContainedBy \rangle$ and $\langle PlaceOfDeath, ContainedBy \rangle$. The subpaths $\langle M, poss, poss^{-1}, M^{-1} \rangle$ and $\langle M, title, title^{-1}, M^{-1} \rangle$ return the random walks back to the query node only if the mentions of the query node have *poss* (stands for *possessive modifier*, e.g. "Bill's clothes") or *title* (stands for *person's title*, e.g. "President Obama") edges in text; otherwise these features are inactive. Therefore, these features are active only for specific subsets of queries. Features in group 8 generally find persons with similar names or mentioned in similar ways to the query person. However, they further expand or restrict this person set in various ways.

Typically, each trained model includes hundreds of paths with non-zero weights, so the bulk of classifications are not based on a few high-precision-recall patterns, but rather on the combination of a large number of lower-precision high-recall or high-precision lower-recall rules.

5.2 Manual Evaluation

We performed two sets of manual evaluations. In each case, an annotator is presented with the triples predicted by PRA, and asked if they are correct. The annotator has access to the Freebase and Wikipedia pages for the concepts (and is able to issue search queries about the concepts).

In the first evaluation, we compared the performance of two PRA models, one trained using the stratified sampled queries and another trained using a randomly sampled set of queries for the profession relation. For each model, we randomly sample 100 predictions from the top 1000 predictions (sorted by the scores returned by the model). We found that the PRA model trained with stratified sampled queries has 0.92 precision, while the other model has only 0.84 precision (significant at the 0.02 level). This shows that stratified sampling leads to improved performance.

Table 4: Human judgement for predicted new beliefs.

Task	p@100	p@1k	p@10k
Profession	0.97	0.92	0.84
Nationality	0.98	0.97	0.90
Parents	0.86	0.81	0.79

We also evaluated the new beliefs proposed by the models trained for all the three relations using stratified sampled queries. We estimated precision for the top 100 predictions and randomly sampled 100 predictions each from the top 1,000 and 10,000 predictions. Here we use the PRA model trained using both KB and text. The results of this evaluation are shown in Table 4. It can be seen that the PRA model is able to produce very high precision predications even when one considers the top 10,000 predictions.

Finally, note that our model is inductive. For instance, for the profession relation, we are able to predict professions for the around 2 million persons in Freebase. The top 1000 profession facts extracted by our system involve 970 distinct people, the top 10,000 facts involve 8,726 distinct people, and the top 100,000 facts involve 79,885 people.

6 Conclusion

We have shown that path constrained random walk models can effectively infer new beliefs from a large scale parsed text corpus with background knowledge. Evaluation by human annotators shows that by combining syntactic patterns in parsed text with semantic patterns in the background knowledge, our model can propose new beliefs with high accuracy. Thus, the proposed random walk model can be an effective way to automate knowledge acquisition from the web.

There are several interesting directions to continue this line of work. First, bidirectional search from both query and target nodes can be an efficient way to discover long paths. This would especially useful for parsed text. Second, relation paths that contain constant nodes (lexicalized features) and conjunction of random walk features are potentially very useful for extraction tasks.

Acknowledgments

We thank Rahul Gupta, Michael Ringgaard, John Blitzer and the anonymous reviewers for helpful comments. The first author was supported by a Google Research grant.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*, DL '00, pages 85–94, New York, NY, USA. ACM.

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.

Aron Culotta, Andrew McCallum, and Jonathan Betz. 2006. Integrating probabilistic extraction models and data mining to discover relations and patterns in text. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 296–303, New York City, USA, June. Association for Computational Linguistics.

Marie-Catherine de Marneffe and Chris Manning. 2008. Stanford dependencies. `http://www.tex.ac.uk/cgi-bin/texfaq2html?label=citeURL`.

Jeffrey Dean and Sanjay Ghemawat. 2008. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January.

Quang Do and Dan Roth. 2010. Constraints based taxonomic relation classification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1099–1109, Cambridge, MA, October. Association for Computational Linguistics.

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.

Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. 1999. Learning Probabilistic Relational Models. In *IJCAI*, volume 16, pages 1300–1309.

Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1152–1161, Singapore, August. Association for Computational Linguistics.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING-92*, pages 539–545. Association for Computational Linguistics, August.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 423–430. Association for Computational Linguistics, July.

Ni Lao and William Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine Learning*, 81:53–67.

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

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore, August. Association for Computational Linguistics.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 41–47, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine Learning*, 62:107–136.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304, Cambridge, MA. NIPS Foundation, MIT Press.

Fabian M. Suchanek, Georgiana Ifrim, and Gerhard Weikum. 2006. Combining linguistic and statistical analysis to extract relations from web documents. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 712–717, New York, NY, USA. ACM.