Learning Relation Entailment Graphs
CSE 515 Final Report

Mitchell Koch, Justin Huang
{mhkoch,jstn}@uw.edu

Friday, June 14, 2013

Abstract

We study the problem of learning relation entailment graphs to represent relationships between predicates extracted from text, for the purpose of improving relation query performance. We use Bayesian network structure learning to learn a graph of relation entailments given results from a relation extraction system, and also leverage the WordNet resource, mapping relation strings to WordNet senses in order to use entailments in WordNet. Using these methods to expand relation queries, we find that Bayesian network structure learning reduces precision, because we were not able to find a way to bias the directionality of edges in a way appropriate for the problem. Using WordNet sense entailments with a logistic regression classifier to select which paths are likely, we increase the number of results returned by over 6x while increasing precision from 87.88% to 91.43% for 15 example queries from Open IE [4].

1 Introduction and Motivation

Relation extraction systems learn facts from natural language text of the form relation(arg1, arg2), such as capture(Caesar, Egypt). The relations can be within a fixed and finite schema, such as Freebase relations as in [8] or they can be any strings in the text likely to represent a relation as in Open IE [4]. Recently there has been work in learning both kinds of relations together using a matrix factorization approach [10].

The goal of relation entailment is to learn a directed graph between relations such that if a relation holds for certain arguments, its child relations should hold for the same arguments. Relation entailment should be helpful to improve relation extraction in any of the above contexts, in that results from relation extraction can be used to generate more results by following the relation entailment graph. This can also be helpful in serving the larger goal of question answering. Not every question can be answered directly from text, making inference using entailments important.

In this project, we focus on increasing the precision and yield of relation extraction results from Open IE using relation entailment and evaluate based on the database query task described below. Since relation entailment involves creating a directed graph over relations, a natural method to try is Bayesian network structure learning. We also use entailments over WordNet senses and learn what entailments are useful for relation entailment using logistic regression.

1.1 Database Query Task

We define the following database query task to evaluate a relation entailment graph. Let R be a set of relation strings, and A be a set of argument strings. We are given a database D of facts, where each fact is of the form $r(x, y)$ for some $r \in R$ and $x, y \in A$.

Given a query for a relation r with arguments x and y, we can expand the query to additionally search for all $r'(x, y)$ such that there exists an edge from r' to r in the entailment graph. An example of this process is shown in figure 1. We can then evaluate the correctness of the returned results.

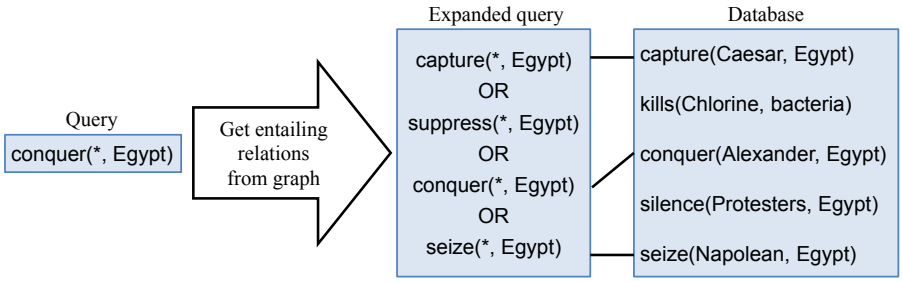

Figure 1: An example of a query being expanded.

2 Methods and Algorithms

2.1 Constraints on WordNet

WordNet [5] is a hand-crafted resource that distinguishes between different word senses and provides synonyms and entailments between them. We denote a WordNet sense with a number where *#1* is the most common. For example *note#4* is the fourth most common sense of "note," which means "to write down". Each WordNet sense has its sense number, a count indicating how often that sense is used, and a probability, which is the count divided by the sum of counts for all senses of the same word.

2.1.1 WordNet Entailment Graph

A troponym is a specialization of a word, e.g. "to fly" is a troponym of "to travel." We say that WordNet sense w_1 entails w_2 if w_1 is a synonym of w_2, if w_1 is a troponym of w_2 in WordNet, or if there exists some w_3 such that w_1 entails w_3 and w_3 entails w_2. A path in the entailment graph is defined as the sequence of entailments between w_1 and w_2. The entailment graph is disconnected. Figure 2 shows one component of the graph, and figure 3 shows an example of a path in the graph, from *note#4* to *write#2*.

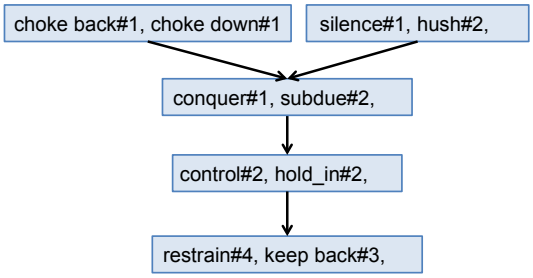

Figure 2: Component of the WordNet entailment graph with the first sense of conquer, *conquer#1*. Boxes represent synonym sets, arrows represent entailments. Trailing commas indicate there are more synonyms not shown.

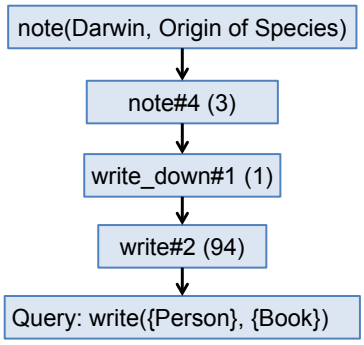

Figure 3: Example path in entailment graph

Although WordNet already defines an entailment graph between verb senses, one hurdle to using it is determining what sense to use. For example, given the verb "take," is it *take#21* (take by force) or *take#2* (take time)? Each sense will lead to very different entailments. First we assume a string can take on any of its WordNet senses, which may lead to noisier entailments. Then we train a logistic regression classifier to help filter out noisy entailments.

2.1.2 Logistic Regression Classifier

For a certain set of queries, we use all possible entailments from WordNet on the database query task and label each fact as relevant to the query or not relevant to the query. This data is subsequently used to train the logistic regression model. For each result, we record the path that was taken in the entailment graph and produce the following features:

1. Path length
2. Average sense number
3. Average WordNet probability
4. Maximum sense number
5. Minimum WordNet probability

The length of the path is the number of nodes in the path. For example, in figure 3, the length is 3. Because WordNet senses are ranked in order frequency, we use the average and maximum sense numbers as features, the idea being that an uncommon WordNet sense (e.g. *take#42*) could be a weak link in the entailment path. A similar intuition holds for the probabilities of the WordNet senses.

The output of logistic regression is a set of weights w for each feature. Let x be the vector of features described above. Then the probability of the label we would like to predict is given by:

$$P(x) = \frac{e^{w \cdot x}}{1 + e^{w \cdot x}} \tag{1}$$

2.2 Bayesian Network Structure Learning

The problem of learning a network from data can be formulated as an optimization problem: Given data $\mathcal{D}$, we want to find a Bayesian network $\mathcal{B}^* = (\mathcal{G}^*, \Theta^*)$ to maximize the posterior probability of the Bayesian network given the data:

$$\begin{aligned} \mathcal{B}^* = \arg\max_{\mathcal{B}} P(\mathcal{B}|\mathcal{D}) &= \arg\max_{\mathcal{B}} P(\mathcal{D}|\mathcal{B})P(\mathcal{B}), & (2)\\ &= \arg\max_{\mathcal{G},\Theta} P(\mathcal{D}|\mathcal{G},\Theta)P(\mathcal{G},\Theta), & (3)\\ &= \arg\max_{\mathcal{G},\Theta} P(\mathcal{D}|\mathcal{G},\Theta)P(\mathcal{G})P(\Theta|\mathcal{G}). & (4) \end{aligned}$$

The posterior probability of the graph prior assuming independence of parameter priors, global parameter independence [6], and parameter modularity [7] yields

$$P(\mathcal{G}|\mathcal{D}) = P(\mathcal{G}) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(N'_{ij})}{\Gamma(N'_{ij} + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(N'_{ijk} + N_{ijk})}{\Gamma(N'_{ijk})}, \tag{5}$$

where q_i is the number of values for the parents of node i, r_i is the number of values for node i, N'_{ijk} is the Dirichlet distribution order for variable i with value k and parent value j, and $N'_{ij} = \sum_{k=1}^{r_i} N'_{ijk}$. N_{ijk} is the number of instances in $\mathcal{D}$ where i with value k has parents with value j and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. This is derived in [2] and [13]. In [13], this score metric is called the General Dirichlet Prior Score Metric (DPSM).

The number of model structures is super-exponential in the number of variables, making exact optimization intractable in general. Approximate approaches include direct search using hill climbing as well as Markov Chain Monte Carlo (MCMC) techniques as in [6]. For smaller graphs, with less than 20 nodes, dynamic programming techniques such as in [3] can be used to compute all marginal posterior edge probabilities exactly.

In order to map the problem of relation entailment to Bayesian network structure learning, we represent relation extraction results as a matrix with entity-entity pairs in the rows and relations in the columns as in [10]. This is used as data $\mathcal{D}$ for Bayesian network structure learning with the relations being the variables in the graph and the entity-entity pairs being the training instances. We want to be able to handle large relation entailment graphs with more than 20 relations, and we do not know the node ordering a priori, so we use MCMC, sampling structures from the posterior distribution $P(\mathcal{G}|\mathcal{D})$.

3 Experiments and Results

We evaluate our relation entailment graphs on the database query task, as described in section 1.1. The queries we chose to evaluate over are listed in table 1. The first five queries we chose originally, and the final ten we chose to exercise the relation entailment graph we obtained through Bayesian network structure learning. Without the final 10 queries, structure learning performs the same as the baseline, since there are no expansions as shown in table 2.

Our baseline is a system that does no query expansion, only searching the database for an exact string match of the relation phrase. We compare this to systems that expand queries based on the entailment graphs from 1) all of WordNet, 2) WordNet with logistic regression, and 3) Bayesian network structure learning.

We manually labeled the results from out WordNet-based system without logistic regression, and trained the logistic regression model on the labeled data using 10-fold cross validation.

We used the Bayes Net Toolbox for Matlab [9] to perform Bayesian network structure learning. The input to the structure learning was a counts matrix, with relation phrases for rows and entity-entity pairs for columns. However, we were constrained by high memory usage when running structure learning on graphs with more than 150 relation phrases. As a result, for some queries, we had to randomly sample 150 relation phrases. The search was done using 100 samples of MCMC, with a burn-in of 10 samples. We used tabular CPDs, Boolean variables representing whether the relation holds, and searched according to

Table 1: Benchmark queries for database query task. The first five queries we chose originally, and the final ten we chose to exercise our structure learning graph.

Query
has-written(type: Person, type: Book)
play(Tom Hanks, ?)
conquer(?, Egypt)
killed(?, Voldemort)
grown-in(coffee, type: Country)
be-one-of(coffee, type:Country)
work-on(type:Person, type:Book)
be-solidly-establish in(coffee, type:Country)
believe-in(type:Person, type:Book)
play-major-economic-role-in(coffee, type:Country)
be-come-in(type:Person, type:Book)
discuss-character-of(Tom Hanks, ?)
be-originally-find-in(coffee, type:Country)
tell(Tom Hanks, ?)
endorse(Tom Hanks, ?)

equation 5 with a uniform prior ($N'_{ijk} = 1$). We smoothed the counts by adding $\epsilon = 0.1$ to the counts matrix.

Our results are summarized in table 4. These results show that using entailments from WordNet improved recall substantially. We were surprised that using WordNet entailments did not decrease precision more than it did, but that likely stems from our choice of queries. For example, while not many facts in the database specifically say a person "has written" a book, many facts say a person "writes" a book. Since we consider those to be correct answers, those results push precision up. Adding the logistic regression classifier presented us with a precision/yield trade-off but does result in a higher precision than the baseline with over 6x more results.

This set of results shows that our structure learning will increase the number of results returned, but at a high cost to precision. Our results also show that for the more specific queries presented in the final 10 entries of table 1, almost all of the extra results given by WordNet entailments are bad (see table 3). The classifier partially recovered the loss in precision.

4 Discussion and Conclusion

In this project we tried two approaches to learning relation entailment graphs: Bayesian network structured learning and using logistic regression to intelli-

Table 2: Results on database query task for the first five queries in table 1 (initial benchmark queries).

System	# results returned	Precision
Baseline	115	83.48%
Structure learning	115	83.48%
WordNet	1105	86.85%
WordNet + logistic regression	1035	91.50%

Table 3: Results on database query task for the final ten queries in table 1 (more specific queries to exercise structure learning).

System	# results returned	Precision
Baseline	55	98.18%
Structure learning	73	76.71%
WordNet	108	56.48%
WordNet + logistic regression	58	91.38%

gently map entailments to WordNet. Using these methods to expand relation queries, we find that Bayesian network structure learning reduces precision, because we were not able to find a way to bias the directionality of edges in a way appropriate for the problem; however, using WordNet with logistic regression increased the number of results returned by over 6x while increasing precision from 87.88% to 91.43% for the 15 example queries.

Our results suggest that the choice of benchmark queries can make a big difference in the results. However, we noticed that using WordNet with a classifier does seem to increase yield at an acceptable level of precision. Although it's hard to say what queries should be used in our experiment, the first five queries in table 1 are arguably more common than the final 10.

Berant, et al. use both symmetric and asymmetric metrics to represent the relation entailment problem in order to capture edge directionality [1]. They use an asymmetric *Cover* measure, which assumes that if one predicate entails another, relatively many of the features of the entailing predicate should be covered by that of the entailed predicate. This could be adapted as a term for structure learning to bias the scores based on if instances (entity-entity pairs) of one relation are covered by another, or this could be performed as a post-processing stage on the learned relation entailment graph. Another possibility could be to learn a supervised classifier over edge direction using features including instance cover as well as other metrics like those in [1].

An alternative way to model this problem of relation entailment could be

Table 4: Results on database query task for all queries in table 1.

System	# results returned	Precision
Baseline	165	87.88%
Structure learning	183	80.33%
WordNet	1188	84.09%
WordNet + logistic regression	1097	91.43%

based on using the relational Markov network framework of [11] to predict links as in [12]. In [12] relational schemas are defined with object types and attributes. They use the domain of university web page, where both pages and links are objects, and attributes can be information such as from a bag of words. A RMN is defined to provide cliques and potentials between attributes of related entities using templates, such as between class labels of linked pages. The parameters are learned from data over a single relational instantiation using belief propagation. We could model the entities and relations that we are dealing with in a similar way. Instead of pages we would have entities, instead of links we would have relations or relation strings, and potentially we could have entity types, either from Freebase or latent for labels. The cliques could group together entities appearing in the same sentence or document. Then the relation extraction problem would be to learn more links given some observed links, and the relation entailment problem could be approached by extracting the probabilities of one kind of link given another and forming a corresponding relation entailment graph that agrees.

The goal of this work is to work in multiple relation extraction contexts so we would like to apply this to relation extraction results with fixed schemas such as in [8, 10] as well. This should not require any reformulation of the problem for Bayesian network structure learning, since the matrix formulation was based on that of [10], but for using WordNet, we would need to map Freebase relations to WordNet senses, probably based on the canonical relation names of the Freebase relations instead of direct string matching as we do for relation phrases from Open IE.

References

[1] J. Berant, I. Dagan, and J. Goldberger. Learning entailment relations by global graph structure optimization. *Comput. Linguist.*, 38(1):73–111, Mar. 2012.

[2] G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 7:299–347, 1992.

[3] D. Eaton and K. Murphy. Bayesian structure learning using dynamic programming and MCMC. In *UAI*, 2007.

[4] O. Etzioni, M. Banko, S. Soderland, and D. S. Weld. Open information extraction from the web. *Commun. ACM*, 51(12):68–74, Dec. 2008.

[5] C. Fellbaum, editor. *WordNet: an electronic lexical database*. MIT Press, 1998.

[6] N. Friedman and D. Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–126, 2003.

[7] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: the combination of knowledge and data. *Machine Learning*, 20(3):197–243, 1995.

[8] R. Hoffmann, C. Zhang, X. Ling, L. S. Zettlemoyer, and D. S. Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*, pages 541–550, 2011.

[9] K. Murphy. The Bayes Net Toolbox for Matlab. *Computing Science and Statistics*, 33(2):1024–1034, 2001.

[10] S. Riedel, L. Yao, B. M. Marlin, and A. McCallum. Relation extraction with matrix factorization and universal schemas. In *Joint Human Language Technology Conference/Annual Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL '13)*, June 2013.

[11] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, UAI'02, pages 485–492, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[12] B. Taskar, M. F. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *NIPS*, 2003.

[13] S. Yang and K.-C. Chang. Comparison of score metrics for Bayesian network learning. *IEEE Transactions on Systems, Man and Cybernetics: Part A: Systems and Humans*, 32(3):419–428, 2002.