University of Texas at Austin KBP 2013 Slot Filling System: Bayesian Logic Programs for Textual Inference

Yinon Bentor **Amelia Harrison** **Shruti Bhosale** **Raymond Mooney**
Department of Computer Science
The University of Texas at Austin
{yinon,ameliaj,shruti,mooney}@cs.utexas.edu

Abstract

This document describes the University of Texas at Austin 2013 system for the Knowledge Base Population (KBP) English Slot Filling (SF) task. The UT Austin system builds upon the output of an existing relation extractor by augmenting relations that are explicitly stated in the text with ones that are inferred from the stated relations using probabilistic rules that encode commonsense world knowledge. Such rules are learned from linked open data and are encoded in the form of Bayesian Logic Programs (BLPs), a statistical relational learning framework based on directed graphical models. In this document, we describe our methods for learning these rules, estimating their associated weights, and performing probabilistic and logical inference to infer unseen relations. In the KBP SF task, our system was able to infer several unextracted relations, but its performance was limited by the base level extractor.

1 Introduction

In 2013, UT Austin was a first-time participant in the English Slot Filling task of the Text Analysis Conference (TAC) Knowledge Base Population (KBP) evaluation. The system we developed aims to infer relations that are missed by a standard relation extraction system or that can be guessed using general world knowledge. Our approach, called Bayesian Logic Programs for Textual Inference (BLP-TI) constructs Bayesian Logic Programs, a formalism that combines the power of first order Horn logic with probabilistic inference using directed graphical models, to model probabilistic commonsense rules such as “the parents of a child are often spouses” or “children often live in the same state as their parents”. It is our hope that such learned rules can increase the recall of relation extraction and provide useful information to other downstream systems. This work builds upon the approaches described in Raghavan et al. (2012).

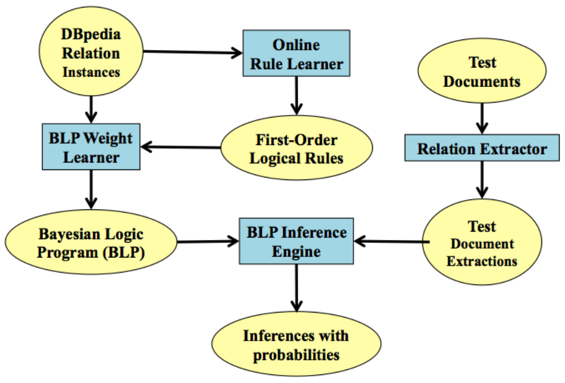
DBpedia Relation Instances
Online Rule Learner
Test Documents
BLP Weight Learner
First-Order Logical Rules
Relation Extractor
Bayesian Logic Program (BLP)
BLP Inference Engine
Test Document Extractions
Inferences with probabilities

Figure 1: System overview

2 System Architecture

The BLP-TI system architecture is shown in Figure 1. We train our system using processed relation instances from DBpedia 3.8 (Lehmann et al., 2013), as described in Section 4. An online inference-rule learning system (Raghavan and Mooney, 2013) proposes a set of first-order definite-clause rules from these relation instances, and subsequently a MLE weight learner uses counts of relation instances to assign weights to each rule. Combined, these form a Bayesian Logic Program (Kersting and De Raedt, 2007). In testing, we apply our learned BLP to extractions from an off-the-shelf relation extraction system, giving us additional inferred facts. In our `UTAUSTIN1` submission, we always preferred inferences made by the BLP-TI system to extractions. Our `UTAUSTIN2` submission serves as a baseline that contains only the output of relation extraction.

3 Bayesian Logic Programs

Bayesian logic programs (BLPs) are a formalism that unifies directed probabilistic graphical models and traditional Prolog-style logic programming, creating *templates* for graphical models. A BLP defines an abstract model composed of first-order Horn clauses and associated conditional probability tables (CPTs) for each rule c such that CPT(c) = P(head(c)|body(c)). In BLP clauses, all variables are universally quantified and range restricted such that $variables\{head\} \subseteq variables\{body\}$.

In BLP inference, these abstract models are instantiated with a query and a set of ground facts. Logical deduction (SLD resolution) is used to construct the graphical model by tracing the proof structure. Evidence from multiple rules is combined using a *noisy-or* combining function. Standard methods for graphical model inference in Bayes nets, such as SampleSearch (Gogate and Dechter, 2007), may then be applied.

By constructing the ground graphical model at query time and by only permitting Horn clauses in the rules, it is possible to limit the size and complexity of the ground networks, resulting in more tractable inference compared to Markov Logic Networks (MLNs) (Domingos and Lowd, 2009). Additionally, a study by Raghavan et al. (2012) shows superior performance of BLPs over MLNs on a relation inference task in a similar domain.

4 Training Data and Dependencies

Our rule learner and weight learner are both trained using relations from DBpedia. We first map relation types from the DBpedia ontology to the KBP ontology using deterministic hand-coded rules, discarding tuples that do not have a reasonable mapping in KBP. We further filter the resulting tuples to ensure that the arguments are type consistent with KBP types, resulting in a total of 912,375 training tuples for 28 KBP relation types.

The BLP-TI system expands the output of other KBP systems. We chose to use parts of the output of the open-source `BLENDER 1.5` system (Ji and Grishman, 2011) as input to our system at test time.

Our system did not make explicit use of the reference knowledge base supplied as part of the KBP evaluation, but it was provided to the `BLENDER` system.

5 Algorithms

5.1 Relation Extraction

For extracting base slot-fillers for queries, we used components from the KBP toolkit developed at the BLENDER Lab. We provide brief descriptions of these components and their application below, but direct the reader to the CUNY 2010 team's TAC workshop paper for details (Chen et al., 2010).

We first indexed the KBP reference knowledge base and the KBP source corpus using Apache Lucene[1], an open-source search engine, as in (Chen et al., 2010). This pipeline extracts named entities from context documents and performs query expansion using several heuristic techniques. The expanded queries are executed in Apache Lucene to retrieve potentially relevant documents. Next, each query was processed by two of the `BLENDER` pipelines:

- **Pattern Matching Pipeline**: The document text is matched against patterns that are shipped with the KBP toolkit distribution, which were learned from ground-truth sets of query-answer pairs. Confidence scores are assigned based on how many patterns were matched and the confidence scores of those patterns.

- **Information Extraction Pipeline**: The BLENDER system incorporates the information extraction systems of (Grishman et al., 2005; Ji and Grishman, 2008), which is trained on ACE 2005 relations. The output of this system is mapped to KBP relations using a mapping described in (Chen et al., 2010).

[1] http://lucene.apache.org/

A filtering step checks slot fillers against precompiled dictionaries of country, city, and state names, as well as additional filtering as described in (Chen et al., 2010). Finally, `BLENDER`'s trained reranker re-scores the extracted slot-fillers from the two pipelines based on features like the pipeline used, confidence score assigned by the pipeline, slot type, query entity type and others.

Our KBP submission did not use the `BLENDER` system's question answering pipeline, external knowledge from Freebase, or the MLN-based cross-system and cross-slot reasoning systems described in (Chen et al., 2010).

5.2 Rule Structure Learning

Our rule learning approach follows the online approach of Raghavan and Mooney (2013), but replaces extractions from documents with sets of tuples from DBpedia.

We process 1000 relations at a time, building a directed graph whose nodes represent relation instances. Directed edges are added between relation instances that share one or more constant arguments, with edge direction chosen such that the edge's head is the more frequently seen relation instance. The edge direction encodes the heuristic that relations that are more frequently explicitly stated should help us infer relations that are less frequently stated.

After all sets of relations are processed, the rule learner traverses the resulting graph and constructs rules where the each rule head corresponds to a tail in the graph and each rule body conjoins the nodes traversed to reach that tail. All constants in the resulting rules are replaced with unique variables to create first order rules.

Because BLPs are range restricted as described in Section 3, the rule learner retains only those rules where all variables in the head appear in the body, discarding all other non-conforming rules. The rule learner maintains a count of how often each rule is satisfied in the training set, and can be set to discard rules that do not meet a user-defined threshold.

This online rule learning algorithm has been shown to outperform a Inductive Logic Programming (ILP) system (Mccreath and Sharma, 1998) on some relation types, and, critically, to scale to much larger data sets than is possible with ILP systems.

We refer the reader to Raghavan and Mooney (2013) for additional details of the algorithm and an empirical evaluation.

5.3 Rule Weight Learning

To assign weights to the learned rules, we compute maximum likelihood estimates from the training data using a modified closed world assumption. While we do not assume that our training set is complete, we do assume that if the training set contains facts with a particular entity occurring in the subject position, then it contains all relevant facts about that entity. (Here, by subject position we denote the first argument of the predicate.) This assumption is motivated by the observation that in linked open data resources, such as DBpedia, relation instances are often only present when a notable entity is the subject of the relation. For example, the relation $per{:}children(Barack\_Obama, Sasha\_Obama)$ is more likely to appear than $per{:}parents(Sasha\_Obama, Barack\_Obama)$, since Sasha Obama may not have her own corresponding Wikipedia page. Because of this, maximum likelihood estimates computed using the standard closed world assumption were qualitatively bad, with definitional rules (like $per{:}children(x, y) \rightarrow per{:}parents(y, x)$) assigned weights far from 1. This problem is alleviated by using the modified assumption.

Put more formally, we call a substitution *good* with respect to a rule if when that substitution is applied to the head of the rule the fact generated matches at least one fact in the training set, in both the predicate and the first argument. For each rule, we set the corresponding CPT parameter in the associated *noisy-or* combiner to be the percentage of good substitutions for that rule for which the fact generated is present in the training set.

5.4 Inference

In testing, we use the output of the `BLENDER` KBP system on test queries. In a second pass, we use all the slot fillers found for the original queries as secondary queries to the system, generating additional extractions that may satisfy the body of some of our BLP rules.

We perform BLP inference as described in (Kersting and De Raedt, 2007) and (Raghavan et al., 2012), using each KBP slot as a query on which we perform backwards chaining using SLD resolution to construct a ground Bayesian network, as described in Section 3. We then use SampleSearch (Gogate and Dechter, 2007), an approximate sampling method for Bayesian networks, to estimate the marginal probability of each inferred slot filler.

5.5 Post Processing

We combine extracted and inferred facts to form our KBP submission. Our confidence values correspond to the probability of each inference from the BLP times the confidence value of each extraction used to make that inference, as reported by the BLENDER system. In cases where an inference was used to fill a slot, we propagate the query entity, slot filler, and justification offsets from the extraction

6 Results

We submitted two runs of our KBP system. The first, labeled UTAUSTIN1, combined inferred relations and extracted relations, always preferring an inferred relation when an inference has been made. The second system, UTAUSTIN2, provides only the output of the relation extractor. We submitted our system in this way so that we may evaluate the accuracy of all facts inferred by the BLP.

6.1 Rules

We applied our rule learner to a set of 912,375 facts from DBpedia mapped to the KBP ontology using a deterministic mapping and a filter for type consistency. We divided this set into 913 segments, lexicographically sorted by the first argument, and presented each segment to the rule learner in turn. We were able to map DBPedia relations to 26 of the 41 KBP predicates.

Our rule learner produced 591 rules, including rules with up to 3 predicates in the body. We filtered rules with little empirical support (low counts in the training data) and those whose learned weights were very low. Table 1 shows some of the sample rules learned by the rule learner. We observed that many of the definitional rules had weights close to 1.0, while other rules and weights seemed qualitatively plausible. Our methods did not yield rules with more than one predicate in the body using our automated rule learner, but we supplemented the learned rules with a small set of hand-written rules.

Number of filled slots in responses	603
Number Correct (not in reference KB)	97
Number Redundant with reference KB	15
Number redundant with another response:	0
Number inexact	17
Number incorrect / spurious	474
Recall	0.076
Precision	0.186
F1	0.108

Table 2: Results of UTAUSTIN1 run

Number of filled slots in responses	473
Number Correct (not in reference KB)	99
Number Redundant with reference KB	20
Number redundant with another response:	0
Number inexact	14
Number incorrect / spurious	340
Recall	0.079
Precision	0.225
F1	0.123

Table 3: Results of UTAUSTIN2 run

6.2 KBP Results

The results of our two KBP submissions are presented in Tables 2 and 3.

Additionally, we performed a post-submission run in which we preferred higher confidence extractions when we had a lower confidence inference available. We ran the KBP scorer[2] on this configuration and obtained the results shown in Table 4. In this configuration, our system was able to infer 7 additional correct facts that were not extracted by the BLENDER system, at the expense of precision.

[2]http://surdeanu.info/kbp2013/software.php

Number of filled slots in responses	603
Number Correct (not in reference KB)	105
Number Redundant with reference KB	20
Number redundant with another response:	0
Number inexact	15
Number incorrect / spurious	463
Recall	0.085
Precision	0.207
F1	0.121

Table 4: Results of unsubmitted run that prefers higher confidence extractions to inferences when both are available

per:country_of_birth(A,B) $\rightarrow$ per:countries_of_residence(A,B) [0.77]
If person A was born in country B, he or she likely resided in country B
org:top_members_employees(A,B) $\rightarrow$ org:shareholders(A,B) [0.13]
If person B is a key employee of organization A, then B may be a shareholder in A
per:country_of_birth(A,B) $\rightarrow$ per:country_of_death(A,B) [0.64]
If person A was born in country B, then he or she died in that country
per:country_of_death(A,B) $\rightarrow$ per:country_of_birth(A,B) [0.66]
If person A died in country B, then he or she was likely born in that country
per:children(A,B) $\rightarrow$ per:parents(B,A) [0.96]
If A is a child of B, B is the parent of A

Table 1: Sample learned probabilistic rules. Associated learned weights are in brackets.

7 Discussion and Future Work

Analyzing our results shows several points of potential improvement and some fundamental hurdles:

- The low precision of the input relation extraction system made it difficult to infer unseen relations, especially using rules with multiple predicates in the body. The performance of the base system is described in Table 3.
- Some relations are inherently difficult to infer from other relations, such as org:website or per:charges.
- Some relations can be inferred, but may be helped by an extractor that can provide extractions outside of the target ontology. For example, an extractor that targets the entire DBpedia ontology may extract relations that do not have a KBP mapping but nonetheless can be used in the body of a inference rule whose head is a KBP-mappable relation.
- Because we did not target all of the phenomena that KBP attempts to address, our team did not expend engineering effort on various normalizing and filtering components that may have increased our scores. We also did not build a filter to check if a given inference was already present in the reference KB.
- Our system was not able to distinguish when the response `NIL` was a correct response, such as the case of per:city_of_death for a person who is still alive, from a case where a relation extractor did not find a filler for the slot. Many of the mistakes our system made resulted from rules such as per:city_of_birth(X,Y) $\rightarrow$ per:city_of_death(X,Y), which is a reasonable rule assuming that X is dead. As a result, our system produced many spurious responses when `NIL` was a correct response.
- Our rule learner may perform better if we provide more related relation instances in each batch. We have developed a graph-splitting method that attempts to present highly connected areas in the knowledge graph as a single training batch to the online rule learner but have not yet evaluated it on this task.
- The structure of the justification offsets and single-document sourcing required by the KBP task description limits the judges to accepting as correct only relations that are explicitly stated in the text or that can be inferred directly from what is stated in a short span of text. Our system could in principle combine multiple facts from different documents to infer a new fact, or it could correctly guess a fact from existing facts but lacking sufficient justification to be counted as correct.

In prior work, BLPs have shown promise for inferring relations in similar domains (Raghavan et al., 2012). While we hoped to replicate this in the KBP domain, our results are currently limited. However, our analysis points to several areas for improvement in future work, and several directions that can provide a better evaluation for relation inference tasks.

Acknowledgments

This research was supported in part by the DARPA DEFT program under AFRL grant FA8750-13-2-

0026 and by MURI ARO grant W911NF-08-1-0242. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, AFRL, ARO, DoD or the US government. The authors would also like to acknowledge the contributions of Heath Vinicomb, Karl Pichotta, and Katrin Erk.

References

Zheng Chen, Suzanne Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Matthew Snover, Javier Artiles, Marissa Passantino, and Heng Ji. 2010. CUNY-BLENDER TAC-KBP2010: Entity Linking and Slot Filling System Description. In *Text Analysis Conference (TAC)*. NIST.

P. Domingos and D. Lowd. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan & Claypool, San Rafael, CA.

Vibhav Gogate and Rina Dechter. 2007. SampleSearch: A scheme that searches for consistent samples. In *Proceedings of AISTATS 2007*.

Ralph Grishman, David Westbrook, and Adam Meyers. 2005. NYU's English ACE 2005 system description. Technical report, Department of Computer Science, New York University.

Heng Ji and Ralph Grishman. 2008. Refining Event Extraction through Cross-Document Inference. In *Proceedings of ACL-08: HLT*, pages 254–262. Association for Computational Linguistics.

Heng Ji and Ralph Grishman. 2011. Knowledge Base Population: Successful approaches and challenges. In *ACL*, pages 1148–1158. The Association for Computer Linguistics.

K. Kersting and L. De Raedt. 2007. Bayesian Logic Programming: Theory and tool. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2013. DBpedia - a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal*. (Under review.).

Eric Mccreath and Arun Sharma. 1998. Lime: A system for learning relations. In *Ninth International Workshop on Algorithmic Learning Theory*, pages 336–374. Springer-Verlag.

Sindhu Raghavan and Raymond J. Mooney. 2013. Online inference-rule learning from natural-language extractions. In *Proceedings of the 3rd Statistical Relational AI (StaRAI-13) workshop at AAAI '13*.

Sindhu Raghavan, Raymond J. Mooney, and Hyeonseo Ku. 2012. Learning to "read between the lines" using Bayesian logic programs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL-2012)*, pages 349–358.