TAC KBP 2015 : English Slot Filling Track
Relational Learning with Expert Advice

Dileep Viswanathan
Indiana University, Bloomington
diviswan@indiana.edu

Anurag Wazalwar
Indiana University, Bloomington
anurwaza@indiana.edu

Ameet Soni
Swarthmore College, Swarthmore
soni@cs.swarthmore.edu

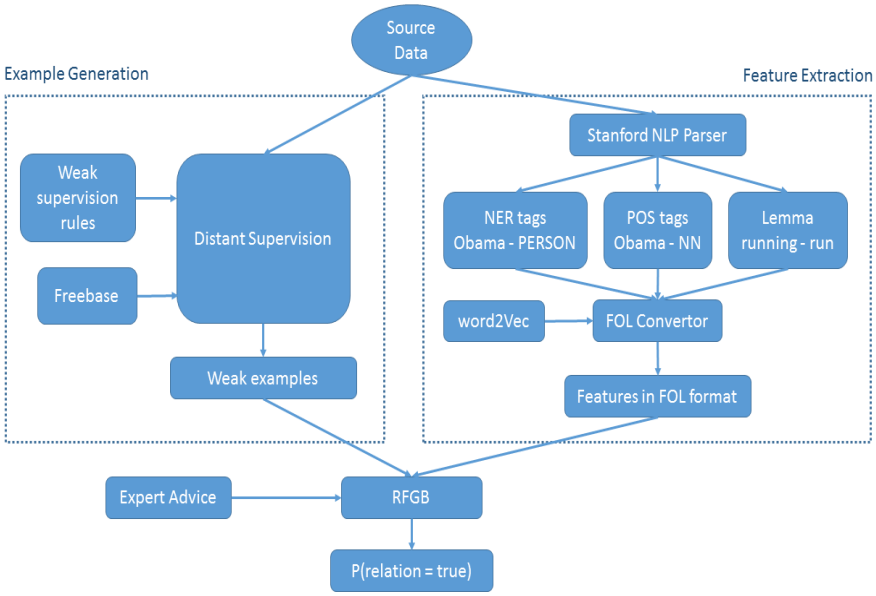
Jude Shavlik
University of Wisconsin-Madison
shavlik@cs.wisc.edu

Sriraam Natarajan
Indiana University, Bloomington
natarasr@indiana.edu

1 Introduction

We present the Wisconsin-Indiana system for the TAC KBP 2015 slot filling task. The proposed approach is a three-staged pipeline. The first stage consists of converting the raw text to a large set of rich relational features (we employ first-order logic for representing the underlying text). During the second stage, weak supervised examples are generated using expert human advice that form weak annotations for the text. In the final stage, the relational features and weak labels are provided as input to an efficient learning system [3] that learns a Relational Dependency Network [6] from the given data.

2 Proposed Pipeline



System overview diagram

2.1 Preprocessing:

Raw text is processed using the Stanford NLP Toolkit (`http://nlp.stanford.edu/software/corenlp.shtml`) to extract parts-of-speech, word lemmas, etc. as well as generate parse trees, dependency graphs and named-entity recognition information. These are then converted into features in prolog format and are given as input to the relation extraction system. In addition to the structured features from the output of Stanford toolkit, we also use deeper features such as word2vec [7] as input to our learning system.

2.2 Human Involvement:

Expert advice is used in our learning system in two ways. In the first step, following our earlier work [4], we took an approach that creates weakly supervised (probabilistic) examples using domain-specific rules. Our insight is that labels are created by "domain experts" who annotate the labels carefully. These domain experts have some inherent rules that they use to create examples. For example, when reading a sports column, there is an inherent bias that we expect "home teams are more likely to win a game" or "in most games, there is a winner and a loser and they are not the same team". We call this *world knowledge* as it describes the domain (or the world) and not specific language constructs. We aim to use such knowledge to create examples for learning from Natural Language (NL) text. More precisely, our hypothesis – which we verify empirically – is that the use of world knowledge will help in learning from NL text. This is particularly true when there is a need to learn a model without any prior structure since the number of examples needed can be large. These weakly supervised examples can augment the gold-standard examples to improve the quality of the learned models.

To this effect, we use the probabilistic logic formalism called *Markov Logic Networks* [2] to perform *weak supervision* [1] to create more examples. Instead of directly obtaining the labels from a different source, we perform inference on *outside knowledge* (i.e., knowledge not explicitly stated in the corpus) to create sets of entities that are "potential" relations. This outside knowledge forms the *context MLN* – CMLN – to reflect that they need not be linguistic models. An example of such knowledge could be that "home teams are more likely to win a game". Note that this approach enables the domain expert to write rules in first-order logic so that the knowledge is not specific to any particular textual wording but is general knowledge about the world (in our example, about the games played). During the information extraction (IE) phase, unlabeled text is parsed through an entity resolution parser to identify potential entities. These entities are used as queries to the CMLN which infers the posterior probability of relations between these entities. These inferred relations become the probabilistic examples for IE. This is in contrast to distant supervision where statistical learning is employed at "system build time" to construct a function from training examples.

The weakly labeled (i.e. probabilistic) examples are combined with gold labeled examples (we labeled about 40 examples each for 18 relations) to create a larger corpus for supervised training.

The second way human expertise is used in our learning system by providing label preferences to the probabilistic learning algorithm [5]. We will explain this in the next section after presenting our learning method.

2.3 Learning:

Given the features and the set of examples, the final step is learning a relational dependency network (RDN) using an algorithm based on functional-gradient boosting [3]. We run both the gold standard and weakly supervised annotated documents through Stanford NLP toolkit to create linguistic features. Once these features are created, we run the RFGB algorithm [3]. This allows us to create a joint model between the target relations. We now briefly describe the adaptation of RFGB to this task.

Our prior work – triggered by the intuition that finding many rough rules of thumb can be faster and more accurate than finding a single, highly accurate local model – turned the problem of learning relational models into a series of relational function approximation problems using functional gradient-

based boosting [8]. The key idea is to represent each relation's conditional distribution as a sum of regression models grown incrementally.

In standard graphical models literature, this is called the pseudo-log-likelihood (PLL) and is defined as:

$$PLL(\mathbf{x}) \equiv \log P(\mathbf{X} = \mathbf{x}) \quad = \quad \log \prod_{x_i \in \mathbf{x}} P(x_i | \mathbf{Ne}(x_i)) = \sum_{x_i \in \mathbf{x}} \log P(x_i | \mathbf{Ne}(x_i))$$

Functional-gradient boosting first computes the functional gradients $\left(\frac{\partial}{\partial \psi(x)}\right)$ of the score that we wish to maximize. In most previous work, our goal was to learn a model that maximizes the PLL of the examples in the training data. Hence, we calculated the functional gradient of PLL for every example.

Recall that, we are employing probabilistic labels that result from the weak supervision phase. Hence, we resort to optimizing a different function, namely, KL-divergence. So instead of optimizing PLL, we optimize the KL-divergence between the observed probabilities of the relations ($P_{obs}(y = \hat{y})$) and the corresponding predicted probabilities ($P_{pred}(y = \hat{y})$). We now present the gradient for this objective function.

$$\begin{aligned} \Delta_m(x) &= \frac{\partial}{\partial \psi_{m-1}} \sum_{\hat{y}} P_{obs}(y = \hat{y}) \log \left(\frac{P_{obs}(y = \hat{y})}{P_{pred}(y = \hat{y} | \psi_{m-1})} \right) \\ &= P_{obs}(y = 1) - P_{pred}(y = 1 | \psi_{m-1}) \end{aligned}$$

The key idea in our work is to use probabilistic examples that we obtain from the weakly supervised phase as input to our structure learning phase along with gold standard examples and their associated documents. Then an RDN is induced by learning to predict the target relations jointly, using features created by the Stanford NLP toolkit. Since we are learning an RDN, we do not have to explicitly check for acyclicity. We chose to employ RDNs as they have been demonstrated to have the state-of-the-art performance in many relational and noisy domains [3].

We recently extended the boosting algorithm to take preferences over labels. The advice consists of preferred and non-preferred target relations - relations that should have higher probability than other relations for any (set of) entity(ies). These advice are defined over spaces of both the gold standard and weakly specified examples. The advice rules are specified as horn clauses. An example advice rule is that "the preferred relations between an husband and wife are *spouse, friends,* etc. and the ones to avoid are *child, enemy,* etc.". Using these relations provide the learning algorithm with an inductive bias that allows for faster convergence. Our algorithm explicitly trades-off between the advice and labeled data when learning the conditional model. For more details of the algorithm, we refer to our knowledge-based probabilistic logic learning work [5].

3 Results

We now present our results on a test set with manually annotated examples. We used about 400 training documents from the TAC KBP 2015 Cold Start Slot Filling evaluation documents. Table 1 presents the list of slots that we evaluated our algorithm on. As can be seen, there are 18 slots that we specifically targeted using our relational advise-based approach.

The results of our algorithm on these different slots are presented in Table 2. We present the area under the ROC and PR curves along with the F1 scores. Since our classifier is a probabilistic one, precision and recall need to be measured at different thresholds. As can be seen from the results, our

Relation	Description
per:age	age of a person
per:origin	origin of a person
per:religion	religion of a person
per:alternate_names	alternate names of a person
per:children	children of a person
per:other_family	other family members of a person
per:parents	parents of a person
per:spouse	spouse of a person
per:siblings	siblings of a person
per:title	titles of a person
org:alternate_names	alternate names of an organization
org:city_of_headquarters	city of headquarters of an organization
org:country_of_headquarters	country of headquarters of an organization
org:date_founded	date in which organization was founded
org:founded_by	person(s)/organization(s) which founded an organization
org:parents	parent organizations of an organization
org:stateorprovince_of_headquarters	state or province of headquarters of an organization
org:subsidiaries	subsidiaries of an organization

Table 1: The query slotes and their brief descriptions.

Relation	AUC ROC	AUC PR	F1
per:age	0.817312	0.363774	0.220551
per:origin	0.426329	0.480946	0.564417
per:religion	0.629187	0.609360	0.500000
per:children	0.873930	0.487770	0.197531
per:other_family	0.851630	0.313805	0.053512
per:parents	0.653824	0.218714	0.290323
per:spouse	0.771928	0.402840	0.416667
per:siblings	0.904845	0.299227	0.353982
per:title	0.532967	0.099058	0.183962
org:city_of_headquarters	0.408140	0.082560	0.171429
org:country_of_headquarters	0.463883	0.157480	0.192547
org:date_founded	0.444076	0.079555	0.123077
org:parents	0.642960	0.506593	0.019108
org:stateorprovince_of_headquarters	0.179487	0.113240	0.222222
org:subsidiaries	0.991379	0.606566	0.200000

Table 2: Results of our algorithm on the different slots. We show the area under the curves and the F1 measure.

algorithm performs reasonably well on several relations such as age, children, spouse, siblings, etc. The key reason is that since it is a joint model, it improves the performance on siblings relation by using predictions from spouse and parents relations. These allow for the model to reason jointly about the family relations from the data.

It must be mentioned that the performance of the system was not quite mirrored in the full KBP evaluation. We hypothesize that this difference in performance is due to the fact that our group focused on the learning side of the KBP task instead of the NLP side. We used off-the-shelf co-reference resolution and entity linking packages. This resulted in the NLP-side errors propagating into the learning side. Nonetheless, on the carefully designed test set, our proposed algorithm appears to perform reasonably well.

Another important thing to mention here is that while our method can potentially use the word2vec features, the current evaluation method did not use these features due to efficiency reasons. The next step is to evaluate our algorithms using these deeper features.

4 Discussion

Our preliminary results indicate that our proposed work is quite successful in learning on 18 relations when evaluated with a hand-constructed tuning set. However, the results are not impressive on the full evaluation. The primary reason appears to be the result of our ignoring co-refs that resulted in lower recall. In addition, we suspect that our method was not effective in document retrieval, which may have resulted in lower recall. Investigating the cause of our results in the final evaluation is an interesting direction of future research.

References

[1] M. Craven and J. Kumlien. Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, 1999.

[2] P. Domingos and D. Lowd. *Markov Logic: An Interface Layer for AI*. Morgan & Claypool, San Rafael, CA, 2009.

[3] S. Natarajan, T. Khot, K. Kersting, B. Guttmann, and J. Shavlik. Gradient-based boosting for statistical relational learning: The relational dependency network case. *Machine Learning*, 86(1):25–56, 2012.

[4] S. Natarajan, J. Picado, T. Khot, K. Kersting, C. Re, and J. Shavlik. Effectively creating weakly labeled training examples via approximate domain knowledge. In *International Conference on Inductive Logic Programming*, 2014.

[5] P. Odom, T. Khot, R. Porter, and S. Natarajan. Knowledge-based probabilistic logic learning. *Association for the Advancement of Artificial Intelligence (AAAI)*, 2015.

[6] J. Neville and D. Jensen. Relational Dependency Networks. *Journal of Machine Learning Research 8: 653-692*, 2007.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *International Conference on Learning Representations Workshop*, 2013.

[8] S. Natarajan, T. Khot, K. Kersting and J. Shavlik. Boosted Statistical Relational Learners: From Benchmarks to Data-Driven Medicine. *SpringerBriefs in Computer Science, ISBN: 978-3-319-13643-1*, 2015.