2012 IEEE 12th International Conference on Data Mining

Learning to Refine an Automatically Extracted Knowledge Base using Markov Logic

Shangpu Jiang Daniel Lowd Dejing Dou
Department of Computer and Information Science
University of Oregon
Eugene, OR 97403
{*shangpu,lowd,dou*}*@cs.uoregon.edu*

***Abstract*—A number of text mining and information extraction projects such as TextRunner and NELL seek to automatically build knowledge bases from the rapidly growing amount of information on the web. In order to scale to the size of the web, these projects often employ ad hoc heuristics to reason about uncertain and contradictory information rather than reasoning jointly about all candidate facts. In this paper, we present a Markov logic-based system for cleaning an extracted knowledge base. This allows a scalable system such as NELL to take advantage of joint probabilistic inference, or, conversely, allows Markov logic to be applied to a web scale problem. Our system uses only the ontological constraints and confidence values of the original system, along with human-labeled data if available. The labeled data can be used to calibrate the confidence scores from the original system or learn the effectiveness of individual extraction patterns. To achieve scalability, we introduce a neighborhood grounding method that only instantiates the part of the network most relevant to the given query. This allows us to partition the knowledge cleaning task into tractable pieces that can be solved individually. In experiments on NELL's knowledge base, we evaluate several variants of our approach and find that they improve both F1 and area under the precision-recall curve.**

***Keywords*-Information extraction; text mining; ontology; Markov logic; knowledge base**

I. Introduction

There is a vast amount of unstructured or semi-structured information on the web in the form of natural language. Automatically acquiring and integrating this information into a structured *knowledge base* (KB) is a crucial *text mining* task. Text mining from the web presents particularly large challenges and opportunities due to the large amount of knowledge, the wide variety of concepts, and the variable quality of information.

Several information extraction systems, such as NELL [1] and TextRunner [2], have been developed for this purpose. NELL and TextRunner both use a bootstrapping approach, starting with some seed knowledge, using it to extract more knowledge, and using the additional knowledge to construct more extraction rules automatically. NELL, in addition, organizes the extracted information in an ontology to enforce consistency and improve the quality of the knowledge base.

However, NELL's handling of uncertainty is relatively limited. It simply filters out any candidate facts that disagree with its existing knowledge, and promotes the highest-confidence facts that remain. When NELL incorporates incorrect facts in its knowledge base, those facts could lead it to exclude correct but contradictory facts from being added later on, even if they were supported by overwhelming evidence. Moreover, NELL also ignores the relationship between confidence values of related (supporting or contradictory) candidate facts.

In order to handle both the large scale and uncertainty in the web, in this paper, we present a new method for automatically cleaning a noisy knowledge base using a *Markov Logic Network* (MLN). Our method performs *joint probabilistic inference* over candidate facts. Ontological constraints from the original information extraction (IE) system serve as hard constraints in the MLN, while confidence values on individual facts serve as soft constraints. We use human labeled facts to learn the weights of the soft constraints, which effectively calibrate the confidence values provided by the IE system. Our method achieves scalability by working on an extracted knowledge base rather than the original text corpus, which could contain millions or billions of web pages. Since the extracted knowledge base could still be very large, we introduce a novel neighborhood-based grounding procedure which divide the knowledge bases into tractable subsets. To evaluate this method, we apply several versions of our MLN and grounding procedure to NELL and show that running joint inference usually leads to higher accuracy, as measured by area under the precision-recall curve (AUC) and F1. Furthermore, we look at examples of specific facts and investigate how joint reasoning helps to predict their correct values.

The rest of the paper is organized as follows. Section II gives brief introductions to MLNs, NELL, and other related work. Section III describes our MLN-based approach in detail. Section IV shows the experiments and analyzes the results. Section V concludes and discusses some directions of future work.

II. Background and Related Work

A. Markov Logic Networks

A Markov logic network [3] consists of a set of weighted formulas in first-order logic, $\{w_i, F_i\}$. Together with a finite set of constants, a Markov logic network defines a probability distribution over possible worlds or Herbrand

1550-4786/12 $26.00 © 2012 IEEE
DOI 10.1109/ICDM.2012.156

interpretations as a log-linear model, where the features are the number of times each formula is satisfied:

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right)$$

Here, x represents a possible world, specified by assigning truth values to all ground atoms. $n_i(x)$ is the number of true groundings of F_i in x. Intuitively, this means that a world in which the ith formula has one more true grounding than another is e^{w_i} times as likely, all other things being equal. Thus, larger weights correspond to stronger constraints. Hard constraints are represented by infinite weights. For more background on Markov logic, see Domingos and Lowd [4].

B. Never Ending Language Learner

In this paper, we use the Never-Ending Language Learner (NELL) system [1], [5] as a case study to explore methods for automatically refining extracted knowledge bases.

There are two types of knowledge that NELL would explore and import into the knowledge base, namely *categories* and *relations*. They can be represented as unary and binary predicates respectively. For example, `Athlete(Tiger Woods)` means that `Tiger Woods` has the category of `Athlete`, and `TeamPlaysSports(Lakers, Basketball)` means that `Lakers` are related to `Basketball` by the relation `TeamPlaysSports`.

NELL starts from a few "seed instances" of each category and relation. It uses natural language processing and information extraction techniques to extract candidate instances from a large web corpus, using the current facts in the knowledge base as training examples. NELL has four subcomponents that extract candidates, namely *Pattern Learner*, *SEAL*, *Morphological Classifier*, and *Rule Learner*, where most candidates are extracted from the first two.

After extracting candidates, NELL's Knowledge Integrator (KI) promotes candidate facts to beliefs when they have support from multiple extraction components or a very high confidence from a single component. Additionally, NELL uses *coupled training* [6], [7] to leverage the ontology hierarchy or other constraints to filter out or share instances among predicates. The ontological constraints can be seen as axioms or rules in first-order logic. For example, we can represent the ontological constraint that every `Athlete` is a `Person` with the rule: `Athlete(x)` $\Rightarrow$ `Person(x)`. Similarly, since a `City` is always not a `Person`, `City(x)` $\Rightarrow \neg$ `Person(x)`, and so on. The KI will not promote an instance if it violates a constraint with another instance that is already in the KB.

A major problem of NELL is that the accuracy of the knowledge it acquires gradually decreases. After the first month, NELL had an estimated precision of 0.9; after two more months, precision had fallen to 0.71, nearly tripling the fraction of incorrect extractions. The underlying reason is that the extraction patterns are not perfectly reliable, so some false instances are extracted as well. The false instances will be used to extract more and more unreliable extraction patterns and false instances, eventually dominating the knowledge base. This kind of error propagation is a common problem of bootstrap learning systems. Coupled training slows the degradation, but does not entirely prevent it. NELL also uses periodic human supervision to remove incorrect facts. However, human labels are often expensive.

C. Other Related Work

Our research is closely related to ontology-based information extraction (OBIE) which combines information extraction with knowledge representation by using ontologies to guide information extraction [8]. Many OBIE systems only extract instances for classes and property values for properties. Such OBIE systems include PANKOW [9], [10]), SOBA [11], OntoSyphon [12], Vulcain [13], and KIM [14]. The Kylin system [15] constructs an ontology based on the structure of Wikipedia infoboxes. Other potentials of OBIE include its ability to create semantic contents for the Semantic web [9] and the ability to use it as a mechanism of improving ontologies [16], [17].

III. Methodology

In this section, we begin with the representation of a knowledge base and associated ontology in Markov logic. Then we discuss how to extend this model to reason more intelligently about the extracted knowledge. We conclude by describing how we make inference and weight learning in this model tractable.

A. Markov Logic Representation of the Knowledge Base and Ontology

For simplicity of our MLN, we use a compact representation in which the names of categories and relations (such as `Bird`, `Animal`, etc.) are viewed as constants of type "category" or "relation" in the second-order predicates `Cat(x,c)` (`x` is an entity of category `c`) or `Rel(x,y,r)` (`x` and `y` have relation `r`). For example, our sample facts from the previous paragraph would be represented as `Cat(Tiger Words, Athlete)` and `Rel(Lakers, Basketball, TeamPlaysSports)`.

In our task, we want to infer the values of `Cat(x,c)` and `Rel(x,y,r)`. The formulas we use to capture the joint distribution of all the ground predicates are as follows.

1) Ontological constraints: The four types of ontological rules used in NELL are: subsumption among categories and relations (e.g., every bird is an animal); mutually exclusive categories and relations (e.g., no person is a location); inversion (for mirrored relations like `TeamHasPlayer` and `PlaysForTeam`); and the type of the domain and range of each predicate (e.g., the mayor of a city must be a person).

We represent these constraints using the following predicates: `Sub` and `RSub` for the subclass relationships; `Mut` and `RMut` are the mutual exclusion relationships; `Inv` is inversion; and `Dom` and `Ran` are the domain and range relationships. These constraints are shown as follows, where the period (.) at the end represents a hard formula:

```
Sub(c1,c2) ∧ Cat(x,c1) ⇒ Cat(x,c2).
RSub(r1,r2) ∧ Rel(x,y,r1) ⇒ Rel(x,y,r2).
Mut(c1,c2) ∧ Cat(x,c1) ⇒ ¬ Cat(x,c2).
RMut(r1,r2) ∧ Rel(x,y,r1) ⇒ ¬ Rel(x,y,r2).
Inv(r1,r2) ∧ Rel(x,y,r1) ⇒ Rel(y,x,r2).
Domain(r,c) ∧ Rel(x,y,r) ⇒ Cat(x,c).
Range(r,c) ∧ Rel(x,y,r) ⇒ Cat(y,c).
```

2) Prior confidence of instances: Facts extracted by an IE system often have different degrees of confidence, based on the amount of supporting evidence available. Rather than simply thresholding or taking the highest-confidence facts consistent with the current knowledge base, Markov logic enables us to reason *jointly* over all facts in order to select an entire set of facts that is mutually consistent and well-supported by evidence.

We use `CandCat(x,c,conf)` and `CandRel(x,y,r,conf)` to represent that `x` has category `c` with confidence `conf`, and `x` and `y` have relation `r` with confidence `conf`. Similarly, we use `PromCat(x,c,conf)` and `PromRel(x,y,r,conf)` to represent the instances actually promoted to the knowledge base, with confidence `conf`. Note that in NELL, the two confidence values may be different, because they are evaluated by different components.

We can incorporate this information using the following formulas:

```
w1 · conf CandCat(x,c,conf) ⇒ Cat(x,c)
w2 · conf CandRel(x,y,r,conf) ⇒ Rel(x,y,r)
w3 · conf PromCat(x,c,conf) ⇒ Cat(x,c)
w4 · conf PromRel(x,y,r,conf) ⇒ Rel(x,y,r)
```

Here $w_i \cdot$ `conf` is the formula weight, which depends on the IE system's original confidence. w_i can be either set to 1 or learned from data.

3) Seed instances: Positive and negative seed instances are fixed to be true and false.

B. Confidences Evaluated by Extraction Patterns

Many IE systems use extraction patterns or rules as a primary means to generate knowledge [2]. Extraction patterns are manually or automatically created and may vary considerably in their effectiveness. If the patterns used to extract each candidate fact and the reliabilities of them are provided, this extra information can help us better determine the truth of the candidate fact.

We use a simple logistic regression model for each category or relation to predict the truth of candidate instances. The features are whether each pattern co-occurs with the instance in the text, and the coefficients reflect the reliability of patterns in extracting facts. If the human labels are available, we can use them to learn the logistic regression model. When labels are not available, we can still use the promoted facts in the knowledge base as labels for learning.

Finally the probabilities of candidate facts are incorporated into the Markov logic by the formulas:

```
w5· conf PattCat(x,c,conf) ⇒ Cat(x,c)
w6· conf PattRel(x,y,r,conf) ⇒ Rel(x,y,r)
```

where `PattCat(x,c,conf)` and `PattRel(x,y,r,conf)` represent a category fact `Cat(x,c)` or a relation fact `Rel(x,y,r)` with probability `conf` learned from logistic regression models.

C. Weight Learning of Formulas

Ideally, we can adopt standard weight learning algorithms [18] to learn weights of the formulas in Markov logic. However, these algorithms tend to be slow, especially in the presence of hard constraints. In this particular problem though, we notice that all facts are independent when hard constraints are absent. In this case, the MLN is equivalent to a logistic regression model. Therefore, we use logistic regression to approximate the weight learning. There are only six soft formulas, whose weights are denoted by $w_i, i = 1, \cdots, 6$ in the previous sections. The weight learning leverages the training labels to automatically determine how good each measure is for the specific knowledge base. Since we do not know how a base IE system calculates the confidence measures, we can also add some simple transformations of the original measures (e.g., log-odds) as additional features.

Huynh and Mooney [19] used a similar approach of learning weights for independent predicates and adding in a hard transitivity constraint at inference time. Using logistic regression may sacrifice some accuracy, but it is much faster than standard Markov logic weight learning algorithms.

D. Inference

Finding the truth value of each fact is a typical MPE inference task. However, due to the large scale and existence of hard constraints, MaxWalkSAT [3] for MPE inference did not produce reasonably good results. As an alternative, we used MC-SAT [20] to compute the marginal probability of each candidate category and relation fact.

The major problem we face in inference is that the scale of an IE system is usually extremely large. For example, NELL extracted more than 943,000 candidate instances by the 165th iteration. This number is even larger for later iterations since the system keeps running and generating more and more candidates. Lazy inference [21] is a general approach to reduce complexity for relational inference algorithms by only instantiating non-default value atoms and formulas. However, when the whole ground network is densely connected and many atoms are supported by weak evidence, lazy inference still tends to instantiate all those atoms and therefore becomes very inefficient.

We developed an alternate approach for making these particular MLN inference problems tractable. First, we notice that the whole network usually forms several clusters, each of which represents a domain, such as facts related to sports or business. Most connections between atoms are between atoms in the same cluster. Second, for each cluster, we are mainly concerned with the values of the query atoms, which for this task consist of the candidate categories and relations in the domain. Other unknown atoms are only useful for their role in correctly inferring the query atoms, and therefore tolerate more error. We treat the query atoms as the center of the network. The close neighbors of them are added in to enable the joint inference, but the distant ones are discarded. In practice, we include 2-hop neighborhood of the central atoms. We can safely adopt these reductions without sacrificing too much accuracy.

The idea of this grounding strategy is similar to lazy inference or cutting plane inference [22]. Compared to lazy inference, our approach further reduces the complexity for large scale problems by explicitly controlling the size of the grounded network. However, unlike lazy inference, it is not guaranteed to produce the same result, but merely approximates it. Our method is also similar to expanding frontier belief propagation (EFBP) [23]. But instead of dynamically selecting a set of atoms affected by updated evidence, we generate the set in advance of the inference phase, which is more efficient and specific for the task.

E. Extensibility of Our Approach

A big advantage of our proposed model compared to other models is that it provides a general framework to combine information from different sources, as long as the information can be represented in first-order logic. This includes many ontologies, which typically encode a large amount of domain-specific knowledge and constraints in first-order logic.

For example, we can easily add extra ontological rules, such as "There is only one capital for each country". Some extensions of NELL and similar IE systems can also be straightforwardly applied to our model. For instance, Lao et al. [24] proposed an approach to learn chain rules in NELL such as "If an athlete x plays for a team y, and team y plays in league z, then x plays in league z." The chain rules can be used to facilitate the system through inference by graph random walks. In Markov logic, this can be viewed as a typical structural learning and MPE inference procedure.

IV. Experiment

A. Methodology

Since NELL is a continuously running system, we used NELL's KB by the 165th iteration as a snapshot for our test. We chose 13 relations and 10 categories, mostly from the sports domain since this domain is widely used in NELL-related research for testing. Each relation has about 1,000 to 2,000 candidate facts and each category has about 5,000 to 10,000 candidate facts. We randomly sampled about 200 facts for each category and relation, 4,511 in total, as the test set. We labeled another 9,887 instances from 5 of the relations and 6 of the categories as the training set.

Table I
Comparison of all 8 methods

Method	AUC	Prec	Recall	F1
NELL	0.765	0.801	0.580	0.673
MLN	0.804	0.726	0.939	0.819
MLN-P	0.817	0.719	0.937	0.814
MLN-PL	0.823	0.833	0.809	0.821
MLN-O	0.874	0.736	**0.946**	0.828
MLN-PO	0.881	0.739	0.927	0.822
MLN-PLO	**0.899**	**0.836**	0.837	**0.836**
MLN-PO*	0.840	0.694	0.751	0.721

Our system produces a list of all test instances, ordered by marginal probability as computed by MC-SAT. We considered all facts with a probability of at least 0.5 as true, and all others as false. For NELL, we took the promoted facts as its result. In addition to precision, recall and F1, we also compared the two methods using area under the precision-recall curve (AUC). Our instances were ordered by their marginal probabilities. For NELL's result, we ordered promoted facts by the associated confidence values, followed by the rest of the candidate facts ordered by their associated confidences as well. This was necessary because NELL's confidence values for promoted and non-promoted facts are not comparable: some promoted facts have lower confidence than some non-promoted candidates. Naively ordering all facts by confidence value led to lower AUCs for NELL.

In order to see how the ontological constraints and pattern information help the joint inference, we experimented on several Markov logic networks to compare with NELL:

- MLN: Uses only the candidate and promoted facts (No extraction patterns or ontological constraints are used);
- MLN-O: MLN adding the ontological constraints;
- MLN-P: MLN adding the extraction patterns, where confidences evaluated by extraction patterns are trained using NELL's promoted facts;
- MLN-PO: MLN-P adding the ontological constraints;
- MLN-PO*: MLN-PO with all w_i set to 1;
- MLN-PL: same as MLN-P, except that confidences evaluated through extraction patterns are trained using human labeling instead;
- MLN-PLO: MLN-PL adding the ontological constraints.

B. Results and Analysis

First, we show a brief comparison of the overall performance of all the 8 methods in Table I. Without the ontological constraints, the MLNs are equivalent to logistic regression models of individual training instances. Table I

shows that MLN, MLN-P and MLN-PL achieve better AUCs and F1 than naively trusting NELL's promoted facts.

The MLNs with ontological constraints, on the other hand, leverage the dependencies between the instances in the joint inference. All the three models with ontological constraints outperform their counterparts without ontological constraints.

The comparison of MLN-O, MLN-PO and MLN-PLO's results show that adding pattern information as an extra feature improves the overall performance. When the labeled training data are available, the results are even better than using NELL's promoted facts as the training data. However, the latter approach can be extended to any new categories or predicates without extra labels, while the former one needs labels in all the categories and predicates to train the pattern's logistic regression model.

It would also be interesting to look into the performances of individual predicates. Due to the limitation of space, we show the detailed overall and per-predicate performance only for NELL, MLN-PO* and MLN-PLO in Table II. MLN-PO* does not use any labeled training data so it is perfectly fair to be compared with NELL, while MLN-PLO is the best MLN with the training data.

As we can see from the table, MLN-PLO has better F1 than NELL in 19 out of 23 predicates, and better AUCs in 16 out of 23 predicates. For the 8 relations and 4 categories that have no labeled training data, MLN-PLO outperforms in 5 relations and 4 categories for F1, and in 3 relations and 3 categories for AUC. MLN-PO* does somewhat better on the relations and categories with no labeled data, obtaining a higher AUC than NELL for 3 out of 4 categories and 6 out of 8 relations, and a higher F1 for all 4 categories and 7 out of 8 relations. Therefore, while both methods are effective, MLN-PO* appears to better generalize to new categories and relations since it does not rely on any training data.

Although the increases in precision and recall are modest, we are able to obtain them using only the information that NELL is already using. These gains are realized by replacing NELL's heuristic logical inference with a sound statistical relational approach that considers the joint uncertainty of many facts. The results show that our use of joint probabilistic inference is effective.

Table II
COMPARISON OF KNOWLEDGE INSTANCE RESULTS BY PREDICATE

	F1			AUC		
Predicate	NELL	PO*	PLO	NELL	PO*	PLO
All	0.673	0.719	**0.836**	0.765	0.840	**0.899**
Relations with training data						
AthletePlaysForLeague	0.719	0.723	**0.973**	0.948	0.917	**0.982**
AthletePlaysSport	0.708	0.961	**0.972**	0.939	0.947	**0.983**
StadiumLocatedInCity	0.413	0.413	**0.433**	0.668	0.610	**0.669**
TeamHomeStadium	0.423	0.430	**0.876**	**0.941**	0.927	0.918
TeamPlaysInLeague	0.419	0.466	**0.939**	0.979	**0.997**	0.996
Relations without training data						
TeamPlaysSport	0.571	0.803	**0.916**	0.916	**0.925**	0.866
TeamWonTrophy	0.640	0.687	**0.742**	0.721	0.733	**0.826**
ProducesProduct	0.560	0.580	**0.589**	0.683	0.687	**0.742**
Acquired	0.605	**0.617**	0.591	0.650	**0.740**	0.639
CityCapitalOfCountry	**0.869**	0.800	0.797	**0.936**	0.928	0.814
ActorStarredInMovie	0.570	**0.764**	0.557	0.814	0.828	**0.837**
AthletePlaysForTeam	0.395	0.402	**0.992**	0.986	**0.988**	0.967
TeamPlaysInCity	0.410	0.435	**0.583**	**0.718**	0.654	0.467
Categories with training data						
SportsTeam	0.962	**0.977**	0.969	0.979	**0.997**	0.996
Athlete	0.973	**0.984**	**0.984**	0.954	0.993	**0.999**
SportsLeague	0.467	0.516	**0.812**	0.597	0.541	**0.843**
StadiumOrEventVenue	0.940	0.958	**0.960**	0.946	0.953	**0.964**
AwardTrophyTournament	0.430	0.189	**0.549**	0.396	**0.815**	0.649
City	0.960	**0.982**	0.956	0.988	**0.999**	0.998
Categories without training data						
Sport	0.691	**0.746**	0.723	**0.717**	0.697	0.708
Country	0.326	0.379	**0.500**	0.346	0.462	**0.614**
Movie	0.448	**0.730**	0.465	0.534	0.670	**0.690**
Vegetable	0.353	0.417	**0.626**	0.332	0.406	**0.572**

C. Discussion

We may further look at some examples to see how exactly our approach refines the knowledge base and cleans the potential errors.

In the first example, `ProducesProduct` is a relation whose domain is `Company` and range is `Product`. (`Adobe`, `Acrobat reader software`) and (`Adobe`, `Acrobat reader version`) are both candidate instances of `ProducesProduct` and have the same initial confidence. Our approach notices that `Acrobat reader software` has a higher confidence value (and thus higher probability) than `Acrobat reader version` to be an instance of `product`. Therefore it assigns a higher probability to the former relation instance than the latter one. NELL also uses type checking constraints, but its logical approach only allows the true relation instance to identify the true category instance, not vice versa.

Another example is that the entity `Los Angeles county` is extracted as a candidate for two disjoint categories `City` and `County`. Although the former is wrong, it was extracted first and got promoted since it had strong supporting evidence at that time. The latter was not promoted by NELL because it violated the mutual exclusion rule with an existing fact, even though it has stronger evidence. In this case, our joint inference framework is able to smartly reason about contradictory instances using all available information, rather than stubbornly enforcing earlier decisions.

V. CONCLUSION AND FUTURE WORK

We have proposed a method for cleaning an automatically extracted knowledge base using Markov logic. Our method uses probabilistic inference to simultaneously reason about the truth values of many related facts. This is an

improvement on systems such as NELL, which uses logical inference and heuristics to update its knowledge base. Our proposed model is also a generic approach that can be extended with other sources of knowledge and constraints in first-order logic. Preliminary experiments show that our method achieves better F1 score and AUC than NELL's knowledge base. We also developed a custom local grounding method to make inference in this problem tractable. By learning weights with logistic regression for different matched patterns, we are able to create a confidence measure that is better calibrated than NELL's. In the future work, we would also like to explore doing unsupervised or semi-supervised learning, to automatically learn the strength of these relationships without requiring many human labels.

VI. Acknowledgments

We thank Ameneh Sarbaziazad for data labeling. This research is funded by NSF grant IIS-1118050.

References

[1] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka, Jr., and T. M. Mitchell, "Toward an architecture for never-ending language learning," in *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*, 2010.

[2] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the web," in *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, San Francisco, CA, USA, 2007, pp. 2670–2676.

[3] M. Richardson and P. Domingos, "Markov logic networks," *Mach. Learn.*, vol. 62, pp. 107–136, February 2006.

[4] P. Domingos and D. Lowd, "Markov logic: An interface layer for artificial intelligence," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 3, no. 1, pp. 1–155, 2009.

[5] T. Mitchell, W. Cohen, J. Estevam Hruschka, B. Settles, D. Wijaya, E. Law, J. Betteridge, J. Krishnamurthy, and B. Kisiel, "Read the web," `http://rtw.ml.cmu.edu/rtw/`.

[6] A. Carlson, J. Betteridge, E. R. Hruschka, Jr., and T. M. Mitchell, "Coupling semi-supervised learning of categories and relations," in *Proceedings of the NAACL HLT 2009 Workskop on Semi-supervised Learning for Natural Language Processing*, 2009.

[7] A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka, Jr., and T. M. Mitchell, "Coupled semi-supervised learning for information extraction," in *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM 2010)*, 2010.

[8] D. C. Wimalasuriya and D. Dou, "Ontology-Based Information Extraction: An Introduction and a Survey of Current Approaches," *Journal of Information Science*, vol. 36, no. 3, pp. 306–323, 2010.

[9] P. Cimiano, S. Handschuh, and S. Staab, "Towards the self-annotating web," in *WWW*, 2004, pp. 462–471.

[10] P. Cimiano, G. Ladwig, and S. Staab, "Gimme' the context: context-driven automatic semantic annotation with C-PANKOW," in *WWW*, 2005, pp. 332–341.

[11] P. Buitelaar and M. Siegel, "Ontology-based information extraction with SOBA," in *LREC*, 2006, pp. 2321–2324.

[12] L. McDowell and M. J. Cafarella, "Ontology-driven information extraction with ontosyphon," in *International Semantic Web Conference*, 2006, pp. 428–444.

[13] A. Todirascu, L. Romary, and D. Bekhouche, "Vulcain - an ontology-based information extraction system," in *NLDB*, 2002, pp. 64–75.

[14] B. Popov, A. Kiryakov, D. Ognyanoff, D. Manov, and A. Kirilov, "KIM – a semantic platform for information extraction and retrieval," *Nat. Lang. Eng.*, vol. 10, no. 3-4, pp. 375–392, 2004.

[15] F. Wu, R. Hoffmann, and D. S. Weld, "Information extraction from wikipedia: moving down the long tail," in *KDD '08: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2008, pp. 731–739.

[16] J. Kietz, A. Maedche, and R. Volz, "A method for semi-automatic ontology acquisition from a corporate intranet," *EKAW-2000 Workshop "Ontologies and Text"*, 2000.

[17] D. Maynard, "Metrics for evaluation of ontology-based information extraction," in *In WWW 2006 Workshop on Evaluation of Ontologies for the Web*, 2006.

[18] D. Lowd and P. Domingos, "Efficient weight learning for Markov logic networks," in *PKDD*, 2007, pp. 200–211.

[19] T. Huynh and R. Mooney, "Discriminative structure and parameter learning for Markov logic networks," in *Proceedings of the 25th International Conference on Machine Learning*. ACM, 2008, pp. 416–423.

[20] H. Poon and P. Domingos, "Sound and efficient inference with probabilistic and deterministic dependencies," in *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, ser. AAAI'06, 2006, pp. 458–463.

[21] H. Poon, P. Domingos, and M. Sumner, "A general method for reducing the complexity of relational inference and its application to MCMC." in *AAAI*, D. Fox and C. P. Gomes, Eds. AAAI Press, 2008, pp. 1075–1080.

[22] S. Riedel, "Improving the accuracy and efficiency of MAP inference for Markov logic," in *Proceedings of the Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-08)*, Corvallis, Oregon, 2008, pp. 468–475.

[23] A. Nath and P. Domingos, "Efficient belief propagation for utility maximization and repeated inference," in *AAAI*, 2010.

[24] N. Lao, T. Mitchell, and W. W. Cohen, "Random walk inference and learning in a large scale knowledge base," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, UK., July 2011, pp. 529–539.