Using Markov Logic to Refine an Automatically Extracted Knowledge Base

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

Abstract

A number of information extraction (IE) projects such as NELL and TextRunner seek to build a usable knowledge base from the rapidly growing amount of information on the web. However, these solutions use heuristic approaches to reasoning rather than sound probabilistic inference. In this paper, we present a method based on Markov logic for cleaning an automatically extracted knowledge base using only the confidence values and ontological constraints of the original system. Our approach works by reasoning jointly over all candidate facts. 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, our method improves both F1 and AUC.

1 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 challenging task due to both the large scale of the web and the large degree of uncertainty in the extracted knowledge. A number of information extraction systems, such as NELL [3] and TextRunner [1], have been developed for this purpose. TextRunner uses 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 also uses a bootstrapping approach, but organizes the extracted information in an ontology. In addition to serving as useful organization, the ontological structure can improve the quality of the knowledge base by enforcing consistency constraints. In this way, NELL can take advantage of logical structure.

However, NELL's handling of uncertainty is relatively limited. It combines the confidences of multiple uncertain information extraction components using heuristics, excludes any 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. NELL also ignores the relationship between the uncertainty of different candidate facts. If two related facts have a modest amount of support, then both are likely to be true. On the other hand, if a contradictory fact has some support, that should decrease the probability that a given fact is true.

In this paper, we present an application of statistical relational AI to the problem of automatically cleaning a noisy knowledge base, such as those extracted from the web. Specifically, we construct a Markov logic network (MLN) [7] that reasons jointly about uncertain knowledge while enforcing hard ontological constraints. Rather than repeat the entire knowledge extraction procedure from scratch using an expensive statistical relational model, we work on the extracted facts and confidence values of an existing information extraction system. This allows our approach to take full advantage of the scalability of the underlying system, while improving its results with sound probabilistic reasoning.

Since knowledge bases are often too large to reason about all at once, we introduce a novel neighborhood-based grounding procedure which selects a tractable subset of the knowledge base to reason about. By rotating through different subsets, we can clean a very large knowledge base without running out of memory. Different subsets can also be run in parallel.

To evaluate this method, we apply two 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 2 gives brief introductions to MLNs, NELL, and other related work. Section 3 describes our MLN-based approach in detail. Section 4 shows the experiments and analyzes the results. Section 5 concludes and discusses some directions of future work.

2 Background and Related Work

2.1 Markov Logic Networks

First-order logic (FOL) is an expressive language that is often used to compactly represent complex relationships among entities, including knowledge bases and ontologies. In first-order logic, a *constant* is a symbol representing an object or concept of interest, such as `Basketball` or "`Tiger Woods`". Logical variables range over objects in the domain. A *predicate* or *relation* is a mapping from tuples of constants to Boolean values. For example, `TeamPlaysSport(t, s)` is a predicate that is true if team `t` plays sport `s`. An atomic formula or *atom* is the application of a predicate to a tuple of variables and/or constants, e.g., `TeamPlaysSport(t, Basketball)`. Formulas are recursively constructed from quantifies ($\exists, \forall$), logical connectives ($\wedge$, $\vee$, $\Rightarrow$, $\neg$, etc.), and atoms. A knowledge base is often represented as a set of formulas, $\{(F_i)\}$. A ground formula or ground atom is one where all logical variables have been replaced by constants.

Inference in first-order logic is semi-decidable in general, but becomes decidable when all functions have known values and the set of constants is finite. One key weakness of first-order logic is that it is very brittle: a single inconsistency renders the entire knowledge base false. In the real world, our knowledge is often uncertain. Even with perfect knowledge of the world, many events are inherently stochastic.

A Markov logic network [7] softens a first-order knowledge base by attaching a real-valued weight w_i to each formula F_i. If formulas are viewed as hard constraints on the set of possible worlds, Markov logic turns these hard constraints into soft constraints, where larger weights intuitively represent stronger constraints and smaller weights represent weaker constraints. Together with a finite set of constants, a Markov logic network defines a probability distribution over possible worlds or Herbrand interpretations as a log-linear model, where the features are the number of times each formula is satisfied:

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right) = \frac{1}{Z} \prod_i \phi_i(x_{\{i\}})^{n_i(x)}$$

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, $x_{\{i\}}$ is the state (truth values) of the predicates appearing in F_i and $\phi(x_i) = e^{w_i}$.

One of the strengths of MLNs is in performing joint inference over a set of related, uncertain facts. For example, Singla and Domingos [20] perform entity resolution based on Markov logic by jointly inferring which pairs of bibliographic entries refer to the same paper. Their approach simultaneously infers equivalences among paper authors, titles, and venues. Poon and Domingos [17] used Markov logic to extract and match database records from CiteSeer text, using joint inference to perform simultaneous segmentation and entity resolution.

For more background on Markov logic, see Domingos and Lowd [7].

Inference Inference helps us reason probabilistically about complex relations in Markov logic networks. There are two basic types of inference: maximum a posteriori (MAP)/most probable explanation (MPE) inference that finds the most probable state of the world consistent with some evidence, as well as conditional/marginal probability inference that finds the conditional/marginal distribution of a formula or a predicate. Any Markov network inference algorithm can be applied to Markov networks, but specialized algorithms that exploit the structure of MLNs often give better performance.

In our experiments, we used the MC-SAT algorithm [16], a Markov chain Monte Carlo algorithm for computing marginal and conditional probabilities in MLNs. Given a current state, MC-SAT selects a random satisfying assignment of a random subset of the currently satisfied clauses. This allows MC-SAT to handle the mix of hard and soft constraints which are often present in MLNs. It can be shown that the set of samples from MC-SAT converges to the correct distribution as long as the satisfying assignments are selected uniformly at random.

Traditional inference algorithm first fully instantiate all the FOL formulas by grounding and then proceed on a propositional level. This requires memory on the order of the number of constants raised to the arity of the clause. Lazy inference [20, 18] takes advantage of the sparseness in typical relational domains (most ground predicates are false, and most clauses are trivially satisfied), by only putting into memory the non-default value ground predicates and clauses. With lazy inference, the memory cost does not scale with total number of groundings, but only with the number of non-default value groundings. Lazy variants have been developed for both MaxWalkSAT and MC-SAT.

2.2 Never Ending Language Learner

In this paper, we use the Never-Ending Language Learner (NELL) system [13, 3, 12] as a case study to explore methods for automatically refining extracted knowledge bases. NELL is an information extraction system proposed and implemented by a group of researchers at Carnegie Mellon University. The final goal of NELL is to create an AI system that runs 24 hours per day, 7 days per week, forever, performing two tasks each day:

- Reading task: extract information from web and populate a knowledge base containing structured facts.
- Learning task: improve its reading ability so that it can extract more facts from the web, more accurately.

NELL starts from a small number of "seed instances" of each category and relation in the seed ontology. 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. The four subcomponents that extract candidates are *Pattern Learner*, *SEAL*, *Morphological Classifier*, and *Rule Learner*, where most candidates are extracted from the first two subcomponents. The Pattern Learner is a free-text extractor which learns and uses contextual patterns such as "mayor of X" and "X plays for Y" to extract instances of categories and relations. The extraction patterns are learned using the co-occurrence statistics between noun phrases and contextual patterns. SEAL is a semi-structured extractor which queries the webpages with instances, and mines lists and tables to learn new instances of the corresponding predicate. It is based on the assumption that the entities showing up in the same list or table tend to belong to the same category or have the same relation. Morphological Classifier uses a set of binary L_2-regularized logistic regression models to classify noun phases based on various morphological features. Rule Learner uses the FOIL similar algorithm to learn probabilistic Horn clauses. The learned rules are used to infer new relation instances from the current KB.

After extracting candidates, NELL's Knowledge Integrator (KI) promotes candidate facts to the beliefs using the following strategy: candidates that have high confidence (e.g, posterior > 0.9) from a single source (i.e., extraction subcomponent) are promoted or candidates with lower-confidence are promoted if they have been proposed by multiple sources. However, candidate category instances are not promoted if they already belong to a mutually exclusive category, and relation instances are not promoted unless their arguments are at least candidates for the appropriate category types. NELL heuristically promotes the most likely instances, updates its information extraction systems, and repeats the process, continually expanding its knowledge base and refining its extraction sub-systems. This *bootstrap learning* method takes advantage of the tremendous redundancy in the web corpus. It does not need perfect extraction rules, because multiple pieces of evidence for a new instance can be used to support its correctness.

A major problem of NELL is that the accuracy of the knowledge it acquires gradually decreases as it continues to operate. 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 false instances are extracted sometimes. The false instances will be used to extract more and more unreliable extraction patterns and false instances, and finally, dominate the knowledge base. Error propagation is a common problem of bootstrap learning systems.

Coupled training [2, 4] was proposed to alleviate the problem of error propagation. These constraints can identify the false candidate instances by reasoning on the subsumption, mutual exclusion, and type checking constraints of the concepts in the ontology. For instance, candidate category instances are not promoted if they already belong to a mutually exclusive category, and relation instances are not promoted unless both arguments belong to appropriate categories.

Periodic human supervision is also used in NELL. Coupled training and human supervision can both slow down the process of error propagation to some extent. However, human supervision is very expensive, and both of them cannot prevent error propagation entirely. Recently, Lao et al. [9] presented an approach to combine constrained, weighted, and random walks through the NELL knowledge base graph to reliably infer new facts for NELL. This approach can lean to infer different target relations by tuning the weights associated with random walks that follow different paths through the knowledge base graph.

2.3 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 [22]. Many OBIE systems only extract instances for classes and property values for properties. Such OBIE systems include PANKOW [5], OntoSyphon [11], and KIM [19]. The Kylin system [23] constructs an ontology based on the structure of Wikipedia infoboxes. It is interesting to note that constructing an ontology from text and making extractions with respect to that ontology (in the form of individuals and property values) is similar in principle to open information extraction, where relations of interest are automatically discovered from text. Banko *et al.* have developed the "TextRunner" IE system, which discovers relations from text using machine learning techniques [1]. In addition, Weld *et al.* consider their

Kylin system to be an open information extraction system because it discovers relations from infobox classes of Wikipedia, allowing it to discover about 50,000 relations each having around 10 attributes [21]. Other potentials of OBIE include its ability to create semantic contents for the Semantic web [5, 23] and the ability to use it as a mechanism of improving ontologies [8, 10].

3 Methodology

In this section, we describe the Markov logic networks we use for refining an extracted knowledge base, as well as our neighborhood-based inference method. We begin with the representation of a knowledge base and associated ontology in Markov logic, and then discuss how to extend this approach to reason more intelligently around the extracted knowledge. We conclude the section by describing how we make inference in this model tractable.

Our listings of Markov logic formulas will be in a monospace font, following the syntax used by the Alchemy system [6]: conjunction is represented by `^`, disjunction by `v`, implication by `=>`, and negation by `!`. Symbols for predicates and constants begin with an uppercase letter while logical variable are lowercase. Formula weights are shown to the left of the formula. Hard formulas are represented by placing a period (`.`) at the end of the formula.

3.1 Markov Logic Representation of the Knowledge Base and Ontology

As introduced previously, the NELL knowledge base has two types of predicates: *category*, such as `Athlete(Tiger Woods)`, and *relation*, such as `TeamPlaysSports(Lakers, Basketball)`. The ontology hierarchy and other 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) => Person(x)`. Similarly, since every bird is an animal, `Bird(x) => Animal(x)`, and so on.

However, rather than creating predicates in our MLNs for every category and relation in the ontology, we use a more 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`).

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.

Ontological constraints We represent four types of ontological constraints: 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 the presence of these constraints using the following predicates: `Sub` and `RSub` for the subclass relationships for categories and relations; `Mut` and `RMut` are the mutual exclusion relationships for categories and relations; `Inv` is inversion; and `Dom` and `Ran` are the domain and range relationships. Symbol `!` means negation.

The MLN formulas to enforce these constraints are as follows:

```
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).
```

All of these formulas are maintained as hard constraints, which is equivalent to having an infinitely large weight.

Prior confidence of instances Different 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 allows to reason *jointly* over all facts in order to accept an entire set of facts that is mutually consistent and well-supported by evidence.

In our MLN, we use the predicates `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`. The confidences are real numbers provided by the base IE system used to extract the candidate categories and relations. 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`.

We can incorporate the IE system's confidence by using it as a weight for the corresponding ground fact:

```
conf CandCat(x,c,conf) => Cat(x,c)
conf CandRel(x,y,r,conf) => Rel(x,y,r)
conf PromCat(x,c,conf) => Cat(x,c)
conf PromRel(x,y,r,conf) => Rel(x,y,r)
```

We can assign a weight to these formulas as well, which would effectively scale all of the confidences by a constant value. For our experiment, we simply use the original confidences, which range from 0 to 1. If the base IE system doesn't have a confidence measure, we can just use a constant instead.

Facts not extracted by the system have no assigned confidence, and are assumed to be less likely. We represent this with the following two formulas:

```
0.2 (!EXISTS conf: CandCat(x,c,conf))
        => !Cat(x,c)
0.2 (!EXISTS conf: CandRel(x,y,r,conf))
        => !Rel(x,y,r)
```

In our experiments, we assigned a weight of 0.2 to both weights, as shown. In future work, we hope to learn weights such as these automatically.

Seed instances Seed instances used to initialize the information extraction system are known to be true. We denote these with the `SeedCat` and `SeedRel` predicates, for category and relation facts, respectively. For some categories and relations, there may be negative seed examples, which are denoted as `NSeedCat` and `NSeedRel`.

We handle seed instances with these hard formulas:

```
SeedCat(x,c) => Cat(x,c).
NSeedCat(x,c) => !Cat(x,c).
SeedRel(x,y,r) => Rel(x,y,r).
NSeedRel(x,y,r) => !Rel(x,y,r).
```

3.2 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. Many ontologies are well designed and properly reflects the necessary knowledge of specific domains, and all the knowledge or constraints are in the form of first-order logic. This suggests that our approach has very good extensibility.

For example, while the current ontology used in NELL is simple, in some ontologies, we may have more complex rules such as:

```
Rel(city,country,Citycapitalofcountry)
^ city != city'
=> !Rel(city',country,Citycapitalofcountry).
```

which means there is only one capital for each country. Such formulas can easily be added into the model.

Some current extensions of NELL and similar IE systems can also be straightforwardly applied to our model. For instance, [9] proposes an approach to learn the chain rules in NELL such as:

```
AthletePlaysForTeam(x,y)
^ TeamPlaysInLeague(y,z)
    => AthletePlaysInLeague(x,z)
```

These rules can be used to facilitate the system through inference by graph random walks. In Markov logic, this procedure can be viewed as a typical structural learning and MAP inference procedure. The formulas can be put into our model as:

```
ChainRule(AthletePlaysForTeam,
    TeamPlaysInLeague,
    AthletePlaysInLeague)

ChainRule(r1,r2,r3) ^ Rel(x,y,r1) ^ Rel(y,z,r2)
    => Rel(x,z,r3)
```

3.3 Inference

We used MC-SAT [16] to compute the marginal probability of each candidate category and relation fact. However, we needed to modify our inference task in order to make it tractable, as we describe below.

The major problem we face in inference is that the scale of an information extraction system is usually extremely large. For example, NELL extracted more than 943,000 candidate instances by the 165th iteration. These numbers are even larger for the later iterations since the system keeps running and generating more and more candidates. Lazy inference [18] is a general approach to reduce complexity for relational inference algorithms. In Markov logic, it assumes that most atoms are false by default and most formulas true, so that it only has to instantiate a few number of necessary 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. 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. 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 well as the atoms in the initial unsatisfied clauses as the center of the network. Their close neighbors are also added in to enable the joint inference, but the distant atoms and formulas are discarded. In practice, we include the 2-hop neighborhood of the center atoms. We can safely adopt these two reductions without sacrificing too much accuracy since most discarded groundings are irrelevant to our query.

Other inference methods, such as box propagation [14] and expanding frontier belief propagation (EFBP) [15] have used similar ideas about running inference in a partial network to obtain increased efficiency. Our method is potentially more efficient, since it selects the neighborhood before grounding the model or considering evidence, but it lacks the formal guarantees of the other methods.

4 Experiment

4.1 Methodology

We evaluated our approach by applying it to the knowledge base extracted by NELL. We used the Markov logic formulas introduced in the previous sections. NELL's candidate instances, candidate extraction patterns, and seed instances were treated as evidence. Since NELL is a continuously running system, we took a snapshot for test. We used the 165th iteration as our dataset.

The instances that we chose for comparison spread over multiple predicates on several domains. Most predicates are from the sports domain, since this domain is widely used in NELL-related works for testing. For the comparison, we chose 6 relations (`TeamPlaysSport`, `TeamWonTrophy`, `TeamPlaysInLeague`, `ProducesProduct`, `CityCapitalOfCountry`, `ActorStarredInMovie`) and 4 categories (`Sport`, `Country`, `Movie`, `Vegetable`). Each relation has about 2000 instances and each category has about 5000-10000 instances. For each relation or category, we sampled about 500-1000 instances for testing.

Our system produces a list of all instances, ordered by marginal probability as computed by MC-SAT. We computed the precision, recall, and F1 score of our predictions by thresholding these probabilities, so that all atoms with a probability of at least 0.5 were considered true, and all atoms with a smaller probability were considered false. (We also explored using MaxWalkSAT for MAP inference, but found that it produced worse results.) For NELL, we evaluated precision, recall, and F1 score on its set of promoted facts.

Since NELL uses a semi-supervised bootstrap learning method, at each iteration it only promotes a limited number of high confidence instances into the KB in order to maintain high precision at the possible cost of lower recall. Therefore we also compared the two methods using AUC (area under the precision-recall curve). 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 help the joint inference, we used two models to compare with NELL:

- MLN-0: Markov logic with the information of candidate and promoted facts, but without the ontological constraints;

Figure 1: Comparison of F1-score, overall and by predicate

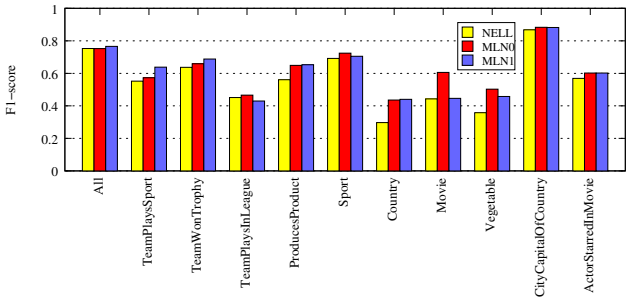
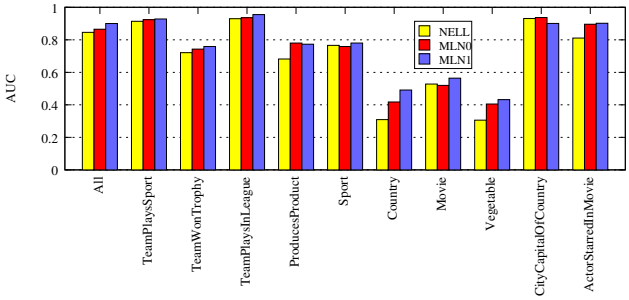


Figure 2: Comparison of AUC, overall and by predicate



- MLN-1: Markov logic with the ontological constraints.

4.2 Results and Analysis

Table 1 shows that our approach (MLN-1) outperforms NELL in both AUC value and F1 score for the complete test set. For the per predicate performance, in 8 out of the 10 predicates our approach outperforms NELL in both AUC value and F1 score.

MLN-0's performance is between NELL's and MLN-1's. This model basically assigns the probability of each instance based on the weighted average of two confidence measures. As we can see, this simple combination is better than simply trusting NELL's promoted facts all the time, but the ontological constraints really play an important role in our approach.

Although our 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 here.

4.3 Discussion

We may further look at some examples to see how exactly our approach refines the knowledge base and cleans the po-

Table 1: Comparison of knowledge instance results by predicate

Measure	NELL	MLN-0	MLN-1
All			
Precision	0.880	0.849	**0.906**
Recall	0.658	**0.675**	0.664
F1	0.753	0.752	**0.766**
AUC	0.846	0.865	**0.900**
TeamPlaysSport			
Precision	**0.987**	0.965	0.942
Recall	0.383	0.408	**0.483**
F1	0.552	0.573	**0.638**
AUC	0.914	0.924	**0.928**
TeamWonTrophy			
Precision	0.684	0.695	**0.733**
Recall	0.595	0.626	**0.649**
F1	0.637	0.659	**0.688**
AUC	0.721	0.743	**0.759**
TeamPlaysInLeague			
Precision	**0.963**	0.961	0.961
Recall	0.295	**0.307**	0.277
F1	0.451	**0.466**	0.430
AUC	0.930	0.936	**0.955**
ProducesProduct			
Precision	0.611	0.762	**0.774**
Recall	0.518	**0.565**	**0.565**
F1	0.561	0.649	**0.653**
AUC	0.682	**0.780**	0.773
Sport			
Precision	0.642	**0.662**	0.661
Recall	0.750	**0.800**	0.755
F1	0.692	**0.724**	0.705
AUC	0.766	0.759	**0.780**
Country			
Precision	0.223	0.327	**0.513**
Recall	0.442	**0.654**	0.385
F1	0.297	0.436	**0.440**
AUC	0.309	0.418	**0.491**
Movie			
Precision	0.554	**0.582**	0.564
Recall	0.369	**0.631**	0.369
F1	0.443	**0.606**	0.446
AUC	0.528	0.520	**0.564**
Vegetable			
Precision	0.264	**0.415**	0.398
Recall	0.557	**0.639**	0.541
F1	0.358	**0.503**	0.458
AUC	0.306	0.405	**0.432**
CityCapitalOfCountry			
Precision	0.958	0.947	**0.959**
Recall	0.793	**0.828**	0.816
F1	0.868	**0.883**	0.882
AUC	0.931	**0.937**	0.901
ActorStarredInMovie			
Precision	0.883	**0.925**	**0.925**
Recall	0.432	**0.446**	**0.446**
F1	0.569	**0.602**	**0.602**
AUC	0.811	0.896	**0.902**

tential errors.

In the first example, `ProducesProduct` is a relation (predicate) 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 noticed that `Acrobat reader software` has a higher confidence value (thus higher probability) than `Acrobat reader version` to be an instance of `product`. Therefore it assigned a higher probability to the former relation instance than the latter one. NELL also uses type checking constraints, but only to maintain consistency with already promoted facts. Thus, it ignores this additional information.

Another example is that the entity `Los Angeles county` is extracted as an instance for both `City` and `County`. Although the former is wrong, it was extracted before the latter and got promoted by NELL since it had strong supporting evidence at that time. The latter also has supporting evidence, but it was not promoted because it violated the mutual exclusion rule of the two categories (i.e., a `City` cannot be a `County`, and vice versa). In this case, NELL's bootstrapping method tries to use the ontological constraints to rule out the wrong instances, but it fails when the wrong instances are promoted first. On the other hand, our joint inference framework is able to smartly reason about contradictory instances using all available information, rather than stubbornly enforcing earlier decisions.

One weakness of our approach is that the weights of the formulas are manually assigned and hard to tune. In future work, we will learn the weights automatically from labelled data or NELL's promoted instances.

5 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.

In future work, we plan to extend our method to include additional information, such as the specific pattern rules matched by different instances. By learning weights for different matched patterns, we may be able to create a confidence measure that is better calibrated than NELL's. 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.

6 Acknowledgments

We thank the anonymous reviews for helpful comments. This research was partly funded by NSF grant IIS-1118050.

References

[1] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open information extraction from the web. In *IJCAI-2007*, pages 2670–2676, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[2] 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.

[3] 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 *AAAI-2010*, 2010.

[4] A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka, Jr., and T. M. Mitchell. Coupled semi-supervised learning for information extraction. In *WSDM-2010*, 2010.

[5] P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *WWW*, pages 462–471, 2004.

[6] P. Domingos, D. Jain, S. Kok, D. Lowd, L. Mihalkova, H. Poon, M. Richardson, P. Singla, M. Sumner, and J. Wang. Alchemy - Open source AI. `http://alchemy.cs.washington.edu/`.

[7] P. Domingos and D. Lowd. Markov logic: An interface layer for artificial intelligence. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 3(1):1–155, 2009.

[8] 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.

[9] N. Lao, T. Mitchell, and W. W. Cohen. Random walk inference and learning in a large scale knowledge base. In *EMNLP-2011*, pages 529–539, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

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

[11] L. McDowell and M. J. Cafarella. Ontology-driven information extraction with ontosyphon. In *International Semantic Web Conference*, pages 428–444, 2006.

[12] 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/`.

[13] T. M. Mitchell, J. Betteridge, A. Carlson, E. R. Hruschka, Jr., and R. C. Wang. Populating the semantic web by macro-reading Internet text. In *ISWC-2009*, 2009.

[14] J. Mooij and B. Kappen. Bounds on marginal probability distributions. *Advances in Neural Information Processing Systems*, 21:1105–1112, 2009.

[15] A. Nath and P. Domingos. Efficient belief propagation for utility maximization and repeated inference. In *AAAI-2010*, 2010.

[16] H. Poon and P. Domingos. Sound and efficient inference with probabilistic and deterministic dependencies. In *AAAI-2006*, pages 458–463. AAAI Press, 2006.

[17] H. Poon and P. Domingos. Joint inference in information extraction. In *AAAI-2007*, pages 913–918, 2007.

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

[19] B. Popov, A. Kiryakov, D. Ognyanoff, D. Manov, and A. Kirilov. KIM – a semantic platform for information extraction and retrieval. *Nat. Lang. Eng.*, 10(3-4):375–392, 2004.

[20] P. Singla and P. Domingos. Memory-efficient inference in relational domains. In *AAAI-2006*, pages 488–493. AAAI Press, 2006.

[21] D. S. Weld, R. Hoffmann, and F. Wu. Using Wikipedia to bootstrap open information extraction. *SIGMOD Record*, 37(4):62–68, 2008.

[22] D. C. Wimalasuriya and D. Dou. Ontology-Based Information Extraction: An Introduction and a Survey of Current Approaches. *Journal of Information Science*, 36(3):306–323, 2010.

[23] F. Wu and D. S. Weld. Autonomously semantifying wikipedia. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 41–50, New York, NY, USA, 2007. ACM.