Unsupervised Ontology Induction from Text

Hoifung Poon and **Pedro Domingos**
Department of Computer Science & Engineering
University of Washington
hoifung,pedrod@cs.washington.edu

Abstract

Extracting knowledge from unstructured text is a long-standing goal of NLP. Although learning approaches to many of its subtasks have been developed (e.g., parsing, taxonomy induction, information extraction), all end-to-end solutions to date require heavy supervision and/or manual engineering, limiting their scope and scalability. We present OntoUSP, a system that induces and populates a probabilistic ontology using only dependency-parsed text as input. OntoUSP builds on the USP unsupervised semantic parser by jointly forming ISA and IS-PART hierarchies of lambda-form clusters. The ISA hierarchy allows more general knowledge to be learned, and the use of smoothing for parameter estimation. We evaluate OntoUSP by using it to extract a knowledge base from biomedical abstracts and answer questions. OntoUSP improves on the recall of USP by 47% and greatly outperforms previous state-of-the-art approaches.

1 Introduction

Knowledge acquisition has been a major goal of NLP since its early days. We would like computers to be able to read text and express the knowledge it contains in a formal representation, suitable for answering questions and solving problems. However, progress has been difficult. The earliest approaches were manual, but the sheer amount of coding and knowledge engineering needed makes them very costly and limits them to well-circumscribed domains. More recently, machine learning approaches to a number of key subproblems have been developed (e.g., Snow et al. (2006)), but to date there is no sufficiently automatic end-to-end solution. Most saliently, supervised learning requires labeled data, which itself is costly and infeasible for large-scale, open-domain knowledge acquisition.

Ideally, we would like to have an end-to-end unsupervised (or lightly supervised) solution to the problem of knowledge acquisition from text. The TextRunner system (Banko et al., 2007) can extract a large number of ground atoms from the Web using only a small number of seed patterns as guidance, but it is unable to extract non-atomic formulas, and the mass of facts it extracts is unstructured and very noisy. The USP system (Poon and Domingos, 2009) can extract formulas and appears to be fairly robust to noise. However, it is still limited to extractions for which there is substantial evidence in the corpus, and in most corpora most pieces of knowledge are stated only once or a few times, making them very difficult to extract without supervision. Also, the knowledge extracted is simply a large set of formulas without ontological structure, and the latter is essential for compact representation and efficient reasoning (Staab and Studer, 2004).

We propose OntoUSP (Ontological USP), a system that learns an ISA hierarchy over clusters of logical expressions, and populates it by translating sentences to logical form. OntoUSP is encoded in a few formulas of higher-order Markov logic (Domingos and Lowd, 2009), and can be viewed as extending USP with the capability to perform hierarchical (as opposed to flat) clustering. This clustering is then used to perform hierarchical smoothing (a.k.a. shrinkage), greatly increasing the system's capability to generalize from

sparse data.

We begin by reviewing the necessary background. We then present the OntoUSP Markov logic network and the inference and learning algorithms used with it. Finally, experiments on a biomedical knowledge acquisition and question answering task show that OntoUSP can greatly outperform USP and previous systems.

2 Background

2.1 Ontology Learning

In general, ontology induction (constructing an ontology) and ontology population (mapping textual expressions to concepts and relations in the ontology) remain difficult open problems (Staab and Studer, 2004). Recently, ontology learning has attracted increasing interest in both NLP and semantic Web communities (Cimiano, 2006; Maedche, 2002), and a number of machine learning approaches have been developed (e.g., Snow et al. (2006), Cimiano (2006), Suchanek et al. (2008,2009), Wu & Weld (2008)). However, they are still limited in several aspects. Most approaches induce and populate a deterministic ontology, which does not capture the inherent uncertainty among the entities and relations. Besides, many of them either bootstrap from heuristic patterns (e.g., Hearst patterns (Hearst, 1992)) or build on existing structured or semi-structured knowledge bases (e.g., WordNet (Fellbaum, 1998) and Wikipedia[1]), thus are limited in coverage. Moreover, they often focus on inducing ontology over individual words rather than arbitrarily large meaning units (e.g., idioms, phrasal verbs, etc.). Most importantly, existing approaches typically separate ontology induction from population and knowledge extraction, and pursue each task in a standalone fashion. While computationally efficient, this is suboptimal. The resulted ontology is disconnected from text and requires additional effort to map between the two (Tsujii, 2004). In addition, this fails to leverage the intimate connections between the three tasks for joint inference and mutual disambiguiation.

Our approach differs from existing ones in two main aspects: we induce a probabilistic ontology from text, and we do so by jointly conducting ontology induction, population, and knowledge extraction. Probabilistic modeling handles uncertainty and noise. A joint approach propagates information among the three tasks, uncovers more implicit information from text, and can potentially work well even in domains not well covered by existing resources like WordNet and Wikipedia. Furthermore, we leverage the ontology for hierarchical smoothing and incorporate this smoothing into the induction process. This facilitates more accurate parameter estimation and better generalization.

Our approach can also leverage existing ontologies and knowledge bases to conduct semi-supervised ontology induction (e.g., by incorporating existing structures as hard constraints or penalizing deviation from them).

2.2 Markov Logic

Combining uncertainty handling and joint inference is the hallmark of the emerging field of statistical relational learning (a.k.a. structured prediction), where a plethora of approaches have been developed (Getoor and Taskar, 2007; Bakir et al., 2007). In this paper, we use Markov logic (Domingos and Lowd, 2009), which is the leading unifying framework, but other approaches can be used as well. Markov logic is a probabilistic extension of first-order logic and can compactly specify probability distributions over complex relational domains. It has been successfully applied to unsupervised learning for various NLP tasks such as coreference resolution (Poon and Domingos, 2008) and semantic parsing (Poon and Domingos, 2009). A *Markov logic network (MLN)* is a set of weighted first-order clauses. Together with a set of constants, it defines a Markov network with one node per ground atom and one feature per ground clause. The weight of a feature is the weight of the first-order clause that originated it. The probability of a state x in such a network is given by the log-linear model $P(x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right)$, where Z is a normalization constant, w_i is the weight of the ith formula, and n_i is the number of satisfied groundings.

2.3 Unsupervised Semantic Parsing

Semantic parsing aims to obtain a complete canonical meaning representation for input sentences. It can be viewed as a structured prediction problem, where a semantic parse is formed by partitioning the input sentence (or a syntactic analysis such as a dependency tree) into meaning units and assigning each unit to the logical form representing an entity or relation (Figure 1). In effect, a semantic

[1] `http://www.wikipedia.org`

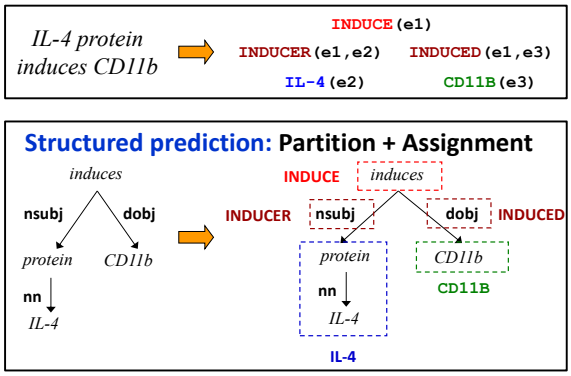

Figure 1: An example of semantic parsing. Top: semantic parsing converts an input sentence into logical form in Davidsonian semantics. Bottom: a semantic parse consists of a partition of the dependency tree and an assignment of its parts.

parser extracts knowledge from input text and converts them into logical form (the semantic parse), which can then be used in logical and probabilistic inference and support end tasks such as question answering.

A major challenge to semantic parsing is syntactic and lexical variations of the same meaning, which abound in natural languages. For example, the fact that IL-4 protein induces CD11b can be expressed in a variety of ways, such as, "Interleukin-4 enhances the expression of CD11b", "CD11b is upregulated by IL-4", etc. Past approaches either manually construct a grammar or require example sentences with meaning annotation, and do not scale beyond restricted domains.

Recently, we developed the USP system (Poon and Domingos, 2009), the first unsupervised approach for semantic parsing.[2] USP inputs dependency trees of sentences and first transforms them into quasi-logical forms (QLFs) by converting each node to a unary atom and each dependency edge to a binary atom (e.g., the node for "induces" becomes $\texttt{induces}(\texttt{e}_1)$ and the subject dependency becomes $\texttt{nsubj}(\texttt{e}_1, \texttt{e}_2)$, where $\texttt{e}_\texttt{i}$'s are Skolem constants indexed by the nodes.).[3] For each sentence, a semantic parse comprises of a partition of its QLF into subexpressions, each of which has a naturally corresponding lambda

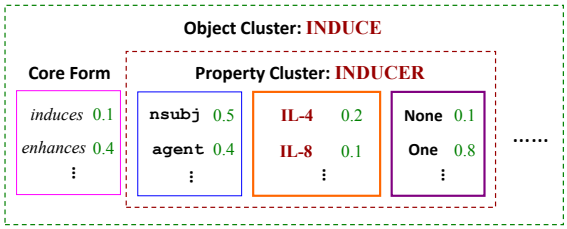

Figure 2: An example of object/property clusters: INDUCE contains the core-form property cluster and others, such as the agent argument INDUCER.

form,[4] and an assignment of each subexpression to a lambda-form cluster.

The lambda-form clusters naturally form an IS-PART hierarchy (Figure 2). An *object cluster* corresponds to semantic concepts or relations such as INDUCE, and contains a variable number of *property clusters*. A special property cluster of *core forms* maintains a distribution over variations in lambda forms for expressing this concept or relation. Other property clusters correspond to modifiers or arguments such as INDUCER (the agent argument of INDUCE), each of which in turn contains three subclusters of property values: the argument-object subcluster maintains a distribution over object clusters that may occur in this argument (e.g., $\texttt{IL} - 4$), the argument-form subcluster maintains a distribution over lambda forms that corresponds to syntactic variations for this argument (e.g., nsubj in active voice and agent in passive voice), and the argument-number subcluster maintains a distribution over total numbers of this argument that may occur in a sentence (e.g., zero if the argument is not mentioned).

Effectively, USP simultaneously discovers the lambda-form clusters and an IS-PART hierarchy among them. It does so by recursively combining subexpressions that are composed with or by similar subexpressions. The partition breaks a sentence into subexpressions that are meaning units, and the clustering abstracts away syntactic and lexical variations for the same meaning. This novel form of relational clustering is governed by a joint probability distribution $P(T, L)$ defined in higher-order[5] Markov logic, where T are the input dependency trees, and L the semantic parses. The

[2] In this paper, we use a slightly different formulation of USP and its MLN to facilitate the exposition of OntoUSP.

[3] We call these QLFs because they are not true logical form (the ambiguities are not yet resolved). This is related to but not identical with the definition in Alshawi (1990).

[4] The lambda form is derived by replacing every Skolem constant $\texttt{e}_\texttt{i}$ that does not appear in any unary atom in the subexpression with a lambda variable $\texttt{x}_\texttt{i}$ that is uniquely indexed by the corresponding node i. For example, the lambda form for $\texttt{nsubj}(\texttt{e}_1, \texttt{e}_2)$ is $\lambda\texttt{x}_1\lambda\texttt{x}_2.\texttt{nsubj}(\texttt{x}_1, \texttt{x}_2)$.

[5] Variables can range over arbitrary lambda forms.

main predicates are:

$\mathtt{e} \in \mathtt{c}$: expression e is assigned to cluster c;

$\mathtt{SubExpr}(\mathtt{s}, \mathtt{e})$: s is a subexpression of e;

$\mathtt{HasValue}(\mathtt{s}, \mathtt{v})$: s is of value v;

$\mathtt{IsPart}(\mathtt{c}, \mathtt{i}, \mathtt{p})$: p is the property cluster in object cluster c uniquely indexed by i.

In USP, property clusters in different object clusters use distinct index i's. As we will see later, in OntoUSP, property clusters with ISA relation share the same index i, which corresponds to a generic semantic frame such as agent and patient.

The probability model of USP can be captured by two formulas:

$$\mathtt{x} \in +\mathtt{p} \wedge \mathtt{HasValue}(\mathtt{x}, +\mathtt{v})$$
$$\mathtt{e} \in \mathtt{c} \wedge \mathtt{SubExpr}(\mathtt{x}, \mathtt{e}) \wedge \mathtt{x} \in \mathtt{p}$$
$$\Rightarrow \exists^1 \mathtt{i}.\mathtt{IsPart}(\mathtt{c}, \mathtt{i}, \mathtt{p}).$$

All free variables are implicitly universally quantified. The "+" notation signifies that the MLN contains an instance of the formula, with a separate weight, for each value combination of the variables with a plus sign. The first formula is the core of the model and represents the mixture of property values given the cluster. The second formula ensures that a property cluster must be a part in the corresponding object cluster; it is a hard constraint, as signified by the period at the end.

To encourage clustering, USP imposes an exponential prior over the number of parameters.

To parse a new sentence, USP starts by partitioning the QLF into atomic forms, and then hill-climbs on the probability using a search operator based on lambda reduction until it finds the maximum a posteriori (MAP) parse. During learning, USP starts with clusters of atomic forms, maintains the optimal semantic parses according to current parameters, and hill-climbs on the log-likelihood of observed QLFs using two search operators:

$\mathtt{MERGE}(\mathtt{c}_1, \mathtt{c}_2)$ merges clusters $\mathtt{c}_1, \mathtt{c}_2$ into a larger cluster c by merging the core-form clusters and argument clusters of $\mathtt{c}_1, \mathtt{c}_2$, respectively. E.g., $\mathtt{c}_1 = \{\text{“}\mathit{induce}\text{”}\}$, $\mathtt{c}_2 = \{\text{“}\mathit{enhance}\text{”}\}$, and $\mathtt{c} = \{\text{“}\mathit{induce}\text{”}, \text{“}\mathit{enhance}\text{”}\}$.

$\mathtt{COMPOSE}(\mathtt{c}_1, \mathtt{c}_2)$ creates a new lambda-form cluster c formed by composing the lambda forms in $\mathtt{c}_1, \mathtt{c}_2$ into larger ones. E.g., $\mathtt{c}_1 = \{\text{“}\mathit{amino}\text{”}\}$, $\mathtt{c}_2 = \{\text{“}\mathit{acid}\text{”}\}$, and $\mathtt{c} = \{\text{“}\mathit{amino\ \ acid}\text{”}\}$.

Each time, USP executes the highest-scored operator and reparses affected sentences using the new parameters. The output contains the optimal lambda-form clusters and parameters, as well as the MAP semantic parses of input sentences.

3 Unsupervised Ontology Induction with Markov Logic

A major limitation of USP is that it either merges two object clusters into one, or leaves them separate. This is suboptimal, because different object clusters may still possess substantial commonalities. Modeling these can help extract more general knowledge and answer many more questions. The best way to capture such commonalities is by forming an ISA hierarchy among the clusters. For example, INDUCE and INHIBIT are both subconcepts of REGULATE. Learning these ISA relations helps answer questions like "What regulates CD11b?", when the text states that "IL-4 induces CD11b" or "AP-1 suppresses CD11b".

For parameter learning, this is also undesirable. Without the hierarchical structure, each cluster estimates its parameters solely based on its own observations, which can be extremely sparse. The better solution is to leverage the hierarchical structure for smoothing (a.k.a. shrinkage (McCallum et al., 1998; Gelman and Hill, 2006)). For example, if we learn that "super-induce" is a verb and that in general verbs have active and passive voices, then even though "super-induce" only shows up once in the corpus as in "AP-1 is super-induced by IL-4", by smoothing we can still infer that this probably means the same as "IL-4 super-induces AP-1", which in turn helps answer questions like "What super-induces AP-1".

OntoUSP overcomes the limitations of USP by replacing the flat clustering process with a hierarchical clustering one, and learns an ISA hierarchy of lambda-form clusters in addition to the IS-PART one. The output of OntoUSP consists of an ontology, a semantic parser, and the MAP parses. In effect, OntoUSP conducts ontology induction, population, and knowledge extraction in a single integrated process. Specifically, given clusters $\mathtt{c}_1, \mathtt{c}_2$, in addition to merge vs. separate, OntoUSP evaluates a third option called *abstraction*, in which a new object cluster c is created, and ISA links are added from $\mathtt{c}_\mathtt{i}$ to c; the argument clusters in c are formed by merging that of $\mathtt{c}_\mathtt{i}$'s.

In the remainder of the section, we describe the

details of OntoUSP. We start by presenting the OntoUSP MLN. We then describe our inference algorithm and how to parse a new sentence using OntoUSP. Finally, we describe the learning algorithm and how OntoUSP induces the ontology while learning the semantic parser.

3.1 The OntoUSP MLN

The OntoUSP MLN can be obtained by modifying the USP MLN with three simple changes. First, we introduce a new predicate $\texttt{IsA}(\texttt{c}_1, \texttt{c}_2)$, which is true if cluster $\texttt{c}_1$ is a subconcept of $\texttt{c}_2$. For convenience, we stipulate that IsA is reflexive (i.e., $\texttt{IsA}(\texttt{c}, \texttt{c})$ is true for any c). Second, we add two formulas to the MLN:

$$\texttt{IsA}(\texttt{c}_1, \texttt{c}_2) \wedge \texttt{IsA}(\texttt{c}_2, \texttt{c}_3) \Rightarrow \texttt{IsA}(\texttt{c}_1, \texttt{c}_3).$$
$$\texttt{IsPart}(\texttt{c}_1, \texttt{i}_1, \texttt{p}_1) \wedge \texttt{IsPart}(\texttt{c}_2, \texttt{i}_2, \texttt{p}_2) \wedge \texttt{IsA}(\texttt{c}_1, \texttt{c}_2) \Rightarrow (\texttt{i}_1 = \texttt{i}_2 \Leftrightarrow \texttt{IsA}(\texttt{p}_1, \texttt{p}_2)).$$

The first formula simply enforces the transitivity of ISA relation. The second formula states that if the ISA relation holds for a pair of object clusters, it also holds between their corresponding property clusters. Both are hard constraints. Third, we introduce hierarchical smoothing into the model by replacing the USP mixture formula

$$\texttt{x} \in +\texttt{p} \wedge \texttt{HasValue}(\texttt{x}, +\texttt{v})$$

with a new formula

$$\texttt{ISA}(\texttt{p}_1, +\texttt{p}_2) \ \wedge \ \texttt{x} \in \texttt{p}_1 \ \wedge \texttt{HasValue}(\texttt{x}, +\texttt{v})$$

Intuitively, for each $\texttt{p}_2$, the weight corresponds to the delta in log-probability of v comparing to the prediction according to all ancestors of $\texttt{p}_2$. The effect of this change is that now the value v of a subexpression x is not solely determined by its property cluster $\texttt{p}_1$, but is also smoothed by statistics of all $\texttt{p}_2$ that are super clusters of $\texttt{p}_1$.

Shrinkage takes place via interaction among the weights of the ISA mixture formula. In particular, if the weights for some property cluster p are all zero, it means that values in p are completely predicted by p's ancestors. In effect, p is backed off to its parent.

3.2 Inference

Given the dependency tree T of a sentence, the conditional probability of a semantic parse L is given by $Pr(L|T) \propto \exp\left(\sum_i w_i n_i(T, L)\right)$. The MAP semantic parse is simply $\arg\max_L \sum_i w_i n_i(T, L)$. Directly enumerating all L's is intractable. OntoUSP uses the same inference algorithm as USP by hill-climbing on the probability of L; in each step, OntoUSP evaluates the alternative semantic parses that can be formed by lambda-reducing a current subexpression with one of its arguments. The only difference is that OntoUSP uses a different MLN and so the probabilities and resulting semantic parses may be different. Algorithm 1 gives pseudo-code for OntoUSP's inference algorithm.

Algorithm 1 OntoUSP-Parse(*MLN, T*)

```
Initialize semantic parse L with individual atoms in the QLF of T
repeat
  for all subexpressions e in L do
    Evaluate all semantic parses that are lambda-reducible from e
  end for
  L ← the new semantic parse with the highest gain in probability
until none of these improve the probability
return L
```

3.3 Learning

OntoUSP uses the same learning objective as USP, i.e., to find parameters θ that maximizes the log-likelihood of observing the dependency trees T, summing out the unobserved semantic parses L:

$$\begin{aligned} L_\theta(T) &= \log P_\theta(L) \\ &= \log \textstyle\sum_L P_\theta(T, L) \end{aligned}$$

However, the learning problem in OntoUSP is distinct in two important aspects. First, OntoUSP learns in addition an ISA hierarchy among the lambda-form clusters. Second and more importantly, OntoUSP leverages this hierarchy during learning to smooth the parameter estimation of individual clusters, as embodied by the new ISA mixture formula in the OntoUSP MLN.

OntoUSP faces several new challenges unseen in previous hierarchical-smoothing approaches. The ISA hierarchy in OntoUSP is not known in advance, but needs to be learned as well. Similarly, OntoUSP has no known examples of populated facts and rules in the ontology, but has to infer that in the same joint learning process. Finally, OntoUSP does not start from well-formed structured input like relational tuples, but rather directly from raw text. In sum, OntoUSP tackles a

Algorithm 2 OntoUSP-Learn(*MLN, T's*)

Initialize with a flat ontology, along with clusters and semantic parses
Merge clusters with the same core form
Agenda $\leftarrow \emptyset$
repeat
  for all candidate operations O **do**
    Score O by log-likelihood improvement
    if score is above a threshold **then**
      Add O to agenda
    end if
  end for
  Execute the highest scoring operation O^* in the agenda
  Regenerate MAP parses for affected trees and update agenda and candidate operations
until agenda is empty
return the learned ontology and MLN, and the semantic parses

very hard problem with exceedingly little aid from user supervision.

To combat these challenges, OntoUSP adopts a novel form of hierarchical smoothing by integrating it with the search process for identifying the hierarchy. Algorithm 2 gives pseudocode for OntoUSP's learning algorithm. Like USP, OntoUSP approximates the sum over all semantic parses with the most probable parse, and searches for both θ and the MAP semantic parses L that maximize $P_\theta(T, L)$. In addition to MERGE and COMPOSE, OntoUSP uses a new operator $\texttt{ABSTRACT}(\texttt{c}_1, \texttt{c}_2)$, which does the following:

1. Create an *abstract* cluster $\texttt{c}$;
2. Create ISA links from $\texttt{c}_1, \texttt{c}_2$ to $\texttt{c}$;
3. Align property clusters of $\texttt{c}_1$ and $\texttt{c}_2$; for each aligned pair $\texttt{p}_1$ and $\texttt{p}_2$, either merge them into a single property cluster, or create an *abstract* property cluster $\texttt{p}$ in $\texttt{c}$ and create ISA links from $\texttt{p}_\texttt{i}$ to $\texttt{p}$, so as to maximize log-likelihood.

Intuitively, $\texttt{c}$ corresponds to a more abstract concept that summarizes similar properties in $\texttt{c}_\texttt{i}$'s.

To add a child cluster $\texttt{c}_2$ to an existing abstract cluster $\texttt{c}_1$, OntoUSP also uses an operator $\texttt{ADDCHILD}(\texttt{c}_1, \texttt{c}_2)$ that does the following:

1. Create an ISA link from $\texttt{c}_2$ to $\texttt{c}_1$;
2. For each property cluster of $\texttt{c}_2$, maximize the log-likelihood by doing one of the following: merge it with a property cluster in an existing child of $\texttt{c}_1$; create ISA link from it to an abstract property cluster in $\texttt{c}$; leave it unchanged.

For efficiency, in both operators, the best option is chosen greedily for each property cluster in $\texttt{c}_2$, in descending order of cluster size.

Notice that once an abstract cluster is created, it could be merged with an existing cluster using MERGE. Thus with the new operators, OntoUSP is capable of inducing any ISA hierarchy among abstract and existing clusters. (Of course, the ISA hierarchy it actually induces depends on the data.)

Learning the shrinkage weights has been approached in a variety of ways; examples include EM and cross-validation (McCallum et al., 1998), hierarchical Bayesian methods (Gelman and Hill, 2006), and maximum entropy with L_1 priors (Dudik et al., 2007). The past methods either only learn parameters with one or two levels (e.g., in hierarchical Bayes), or requires significant amount of computation (e.g., in EM and in L_1-regularized maxent), while also typically assuming a given hierarchy. In contrast, OntoUSP has to both induce the hierarchy and populate it, with potentially many levels in the induced hierarchy, starting from raw text with little user supervision.

Therefore, OntoUSP simplifies the weight learning problem by adopting standard m-estimation for smoothing. Namely, the weights for cluster $\texttt{c}$ are set by counting its observations plus m fractional samples from its parent distribution. When $\texttt{c}$ has few observations, its unreliable statistics can be significantly augmented via the smoothing by its parent (and in turn to a gradually smaller degree by its ancestors). m is a hyperparameter that can be used to trade off bias towards statistics for parent vs oneself.

OntoUSP also needs to balance between two conflicting aspects during learning. On one hand, it should encourage creating abstract clusters to summarize intrinsic commonalities among the children. On the other hand, this needs to be heavily regularized to avoid mistaking noise for the signal. OntoUSP does this by a combination of priors and thresholding. To encourage the induction of higher-level nodes and inheritance, OntoUSP imposes an exponential prior β on the number of *parameter slots*. Each slot corresponds to a distinct property value. A child cluster inherits its parent's slots (and thus avoids the penalty on them). On-

toUSP also stipulates that, in an ABSTRACT operation, a new property cluster can be created either as a *concrete* cluster with full parameterization, or as an *abstract* cluster that merely serves for smoothing purposes. To discourage overproposing clusters and ISA links, OntoUSP imposes a large exponential prior γ on the number of concrete clusters created by ABSTRACT. For *abstract* cluster, it sets a cut-off t_p and only allows storing a probability value no less than t_p. Like USP, it also rejects MERGE and COMPOSE operations that improve log-likelihood by less than t_o. These priors and cut-off values can be tuned to control the granularity of the induced ontology and clusters.

Concretely, given semantic parses L, OntoUSP computes the optimal parameters and evaluates the regularized log-likelihood as follows. Let $w_{p_2,v}$ denote the weight of the ISA mixture formula $\texttt{ISA}(\texttt{p}_1, +\texttt{p}_2) \wedge \texttt{x} \in \texttt{p}_1 \wedge \texttt{HasValue}(\texttt{x}, +\texttt{v})$. For convenience, for each pair of property cluster c and value v, OntoUSP instead computes and stores $w'_{\texttt{c,v}} = \sum_{\texttt{ISA}(\texttt{c},\texttt{a})} w_{\texttt{a,v}}$, which sums over all weights for c and its ancestors. (Thus $w_{\texttt{c,v}} = w'_{\texttt{c,v}} - w'_{\texttt{p,v}}$, where p is the parent of c.) Like USP, OntoUSP imposes local normalization constraints that enable closed-form estimation of the optimal parameters and likelihood. Specifically, using m-estimation, the optimal $w'_{\texttt{c,v}}$ is $\log((m \cdot e^{w'_{\texttt{p,v}}} + n_{\texttt{c,v}})/(m + n_{\texttt{c}}))$, where p is the parent of c and n is the count. The log-likelihood is $\sum_{\texttt{c,v}} w'_{\texttt{c,v}} \cdot n_{\texttt{c,v}}$, which is then augmented by the priors.

4 Experiments

4.1 Methodology

Evaluating unsupervised ontology induction is difficult, because there is no gold ontology for comparison. Moreover, our ultimate goal is to aid knowledge acquisition, rather than just inducing an ontology for its own sake. Therefore, we used the same methodology and dataset as the USP paper to evaluate OntoUSP on its capability in knowledge acquisition. Specifically, we applied OntoUSP to extract knowledge from the GENIA dataset (Kim et al., 2003) and answer questions, and we evaluated it on the number of extracted answers and accuracy. GENIA contains 1999 PubMed abstracts.[6] The question set contains 2000 questions which were created by sampling verbs and entities according to their frequencies in GENIA. Sample questions include "What regulates MIP-1alpha?", "What does anti-STAT 1 inhibit?". These simple question types were used to focus the evaluation on the knowledge extraction aspect, rather than engineering for handling special question types and/or reasoning.

[6] http://www-tsujii.is.s.u-tokyo-.ac.jp/GENIA/home/wiki.cgi.

4.2 Systems

OntoUSP is the first unsupervised approach that synergistically conducts ontology induction, population, and knowledge extraction. The system closest in aim and capability is USP. We thus compared OntoUSP with USP and all other systems evaluated in the USP paper (Poon and Domingos, 2009). Below is a brief description of the systems. (For more details, see Poon & Domingos (2009).)

Keyword is a baseline system based on keyword matching. It directly matches the question substring containing the verb and the available argument with the input text, ignoring case and morphology. Given a match, two ways to derive the answer were considered: KW simply returns the rest of sentence on the other side of the verb, whereas KW-SYN is informed by syntax and extracts the answer from the subject or object of the verb, depending on the question (if the expected argument is absent, the sentence is ignored).

TextRunner (Banko et al., 2007) is the state-of-the-art system for open-domain information extraction. It inputs text and outputs relational triples in the form (R, A_1, A_2), where R is the relation string, and A_1, A_2 the argument strings. To answer questions, each triple-question pair is considered in turn by first matching their relation strings, and then the available argument strings. If both match, the remaining argument string in the triple is returned as an answer. Results were reported when exact match is used (TR-EXACT), or when the triple strings may contain the question ones as substrings (TR-SUB).

RESOLVER (Yates and Etzioni, 2009) inputs TextRunner triples and collectively resolves coreferent relation and argument strings. To answer questions, the only difference from TextRunner is that a question string can match any string in its cluster. As in TextRunner, results were reported for both exact match (RS-EXACT) and substring (RS-SUB).

DIRT (Lin and Pantel, 2001) resolves binary rela-

Table 1: Comparison of question answering results on the GENIA dataset. Results for systems other than OntoUSP are from Poon & Domingos (2009).

	# Total	# Correct	Accuracy
KW	150	67	45%
KW-SYN	87	67	77%
TR-EXACT	29	23	79%
TR-SUB	152	81	53%
RS-EXACT	53	24	45%
RS-SUB	196	81	41%
DIRT	159	94	59%
USP	334	295	88%
OntoUSP	**480**	**435**	**91%**

tions by inputting a dependency path that signifies the relation and returns a set of similar paths. To use DIRT in question answering, it was queried to obtain similar paths for the relation of the question, which were then used to match sentences.

USP (Poon and Domingos, 2009) parses the input text using the Stanford dependency parser (Klein and Manning, 2003; de Marneffe et al., 2006), learns an MLN for semantic parsing from the dependency trees, and outputs this MLN and the MAP semantic parses of the input sentences. These MAP parses formed the knowledge base (KB). To answer questions, USP first parses the questions (with the question slot replaced by a dummy word), and then matches the question parse to parses in the KB by testing subsumption.

OntoUSP uses a similar procedure as USP for extracting knowledge and answering questions, except for two changes. First, USP's learning and parsing algorithms are replaced with OntoUSP-Learn and OntoUSP-Parse, respectively. Second, when OntoUSP matches a question to its KB, it not only considers the lambda-form cluster of the question relation, but also all its sub-clusters.[7]

4.3 Results

Table 1 shows the results comparing OntoUSP with other systems. While USP already greatly outperformed other systems in both precision and recall, OntoUSP further substantially improved on the recall of USP, without any loss in precision. In particular, OntoUSP extracted 140 more correct answers than USP, for a gain of 47% in absolute

[7] Additional details are available at `http://alchemy.cs.washington.edu/papers/poon10`.

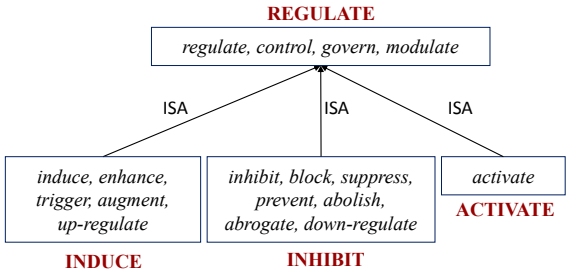

Figure 3: A fragment of the induced ISA hierarchy, showing the core forms for each cluster (the cluster labels are added by the authors for illustration purpose).

recall. Compared to TextRunner (TR-SUB), OntoUSP gained on precision by 38 points and extracted more than five times of correct answers.

Manual inspection shows that the induced ISA hierarchy is the key for the recall gain. Like USP, OntoUSP discovered the following clusters (in core forms) that represent some of the core concepts in biomedical research:

{regulate, control, govern, modulate}

{induce, enhance, trigger, augment, up-regulate}

{inhibit, block, suppress, prevent, abolish, abrogate, down-regulate}

However, USP formed these as separate clusters, whereas OntoUSP in addition induces ISA relations from the `INDUCE` and `INHIBIT` clusters to the `REGULATE` cluster (Figure 3). This allows OntoUSP to answer many more questions that are asked about general regulation events, even though the text states them with specific regulation directions like "induce" or "inhibit". Below is an example question-answer pair output by OntoUSP; neither USP nor any other system were able to extract the necessary knowledge.

Q: What does IL-2 control?

A: The DEX-mediated IkappaBalpha induction.

Sentence: Interestingly, the DEX-mediated IkappaBalpha induction was completely inhibited by IL-2, but not IL-4, in Th1 cells, while the reverse profile was seen in Th2 cells.

OntoUSP also discovered other interesting commonalities among the clusters. For example, both USP and OntoUSP formed a singleton cluster with core form "activate". Although this cluster may appear similar to the `INDUCE` cluster, the data in GENIA does not support merging the two. However, OntoUSP discovered that

the ACTIVATE cluster, while not completely resolvent with INDUCE, shared very similar distributions in their agent arguments. In fact, they are so similar that OntoUSP merges them into a single property cluster. It found that the patient arguments of INDUCE and INHIBIT are very similar and merged them. In turn, OntoUSP formed ISA links from these three object clusters to REGULATE, as well as among their property clusters. Intuitively, this makes sense. The positive- and negative-regulation events, as signified by INDUCE and INHIBIT, often target similar object entities or processes. However, their agents tend to differ since in one case they are inducers, and in the other they are inhibitors. On the other hand, ACTIVATE and INDUCE share similar agents since they both signify positive regulation. However, "activate" tends to be used more often when the patient argument is a concrete entity (e.g., cells, genes, proteins), whereas "induce" and others are also used with processes and events (e.g., expressions, inhibition, pathways).

USP was able to resolve common syntactic differences such as active vs. passive voice. However, it does so on the basis of individual verbs, and there is no generalization beyond their clusters. OntoUSP, on the other hand, formed a high-level cluster with two abstract property clusters, corresponding to general agent argument and patient argument. The active-passive alternation is captured in these clusters, and is inherited by all descendant clusters, including many rare verbs like "super-induce" which only occur once in GENIA and for which there is no way that USP could have learned about their active-passive alternations. This illustrates the importance of discovering ISA relations and performing hierarchical smoothing.

4.4 Discussion

OntoUSP is a first step towards joint ontology induction and knowledge extraction. The experimental results demonstrate the promise in this direction. However, we also notice some limitations in the current system. While OntoUSP induced meaningful ISA relations among relation clusters like REGULATE, INDUCE, etc., it was less successful in inducing ISA relations among entity clusters such as specific genes and proteins. This is probably due to the fact that our model only considers local features such as the parent and arguments. A relation is often manifested as verbs and has several arguments, whereas an entity typically appears as an argument of others and has few arguments of its own. As a result, in average, there is less information available for entities than relations. Presumably, we can address this limitation by modeling longer-ranged dependencies such as grandparents, siblings, etc. This is straightforward to do in Markov logic.

OntoUSP also uses a rather elaborate scheme for regularization. We hypothesize that this can be much simplified and improved by adopting a principled framework such as Dudik et al. (2007).

5 Conclusion

This paper introduced OntoUSP, the first unsupervised end-to-end system for ontology induction and knowledge extraction from text. OntoUSP builds on the USP semantic parser by adding the capability to form hierarchical clusterings of logical expressions, linked by ISA relations, and using them for hierarchical smoothing. OntoUSP greatly outperformed USP and other state-of-the-art systems in a biomedical knowledge acquisition task.

Directions for future work include: exploiting the ontological structure for principled handling of antonyms and (more generally) expressions with opposite meanings; developing and testing alternate methods for hierarchical modeling in OntoUSP; scaling up learning and inference to larger corpora; investigating the theoretical properties of OntoUSP's learning approach and generalizing it to other tasks; answering questions that require inference over multiple extractions; etc.

6 Acknowledgements

We give warm thanks to the anonymous reviewers for their comments. This research was partly funded by ARO grant W911NF-08-1-0242, AFRL contract FA8750-09-C-0181, DARPA contracts FA8750-05-2-0283, FA8750-07-D-0185, HR0011-06-C-0025, HR0011-07-C-0060 and NBCH-D030010, NSF grants IIS-0534881 and IIS-0803481, and ONR grant N00014-08-1-0670. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, DARPA, NSF, ONR, AFRL, or the United States Government.

References

Hiyan Alshawi. 1990. Resolving quasi logical forms. *Computational Linguistics*, 16:133–144.

G. Bakir, T. Hofmann, B. B. Schölkopf, A. Smola, B. Taskar,

S. Vishwanathan, and (eds.). 2007. *Predicting Structured Data*. MIT Press, Cambridge, MA.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 2670–2676, Hyderabad, India. AAAI Press.

Philipp Cimiano. 2006. *Ontology learning and population from text*. Springer.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 449–454, Genoa, Italy. ELRA.

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

Miroslav Dudik, David Blei, and Robert Schapire. 2007. Hierarchical maximum entropy density estimation. In *Proceedings of the Twenty Fourth International Conference on Machine Learning*.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Andrew Gelman and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

Lise Getoor and Ben Taskar, editors. 2007. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. GENIA corpus - a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19:180–82.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the Forty First Annual Meeting of the Association for Computational Linguistics*, pages 423–430.

Dekang Lin and Patrick Pantel. 2001. DIRT - discovery of inference rules from text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 323–328, San Francisco, CA. ACM Press.

Alexander Maedche. 2002. *Ontology learning for the semantic Web*. Kluwer Academic Publishers, Boston, Massachusetts.

Andrew McCallum, Ronald Rosenfeld, Tom Mitchell, and Andrew Ng. 1998. Improving text classification by shrinkage in a hierarchy of classes. In *Proceedings of the Fifteenth International Conference on Machine Learning*.

Hoifung Poon and Pedro Domingos. 2008. Joint unsupervised coreference resolution with Markov logic. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 649–658, Honolulu, HI. ACL.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Singapore. ACL.

Rion Snow, Daniel Jurafsky, and Andrew Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of COLING/ACL 2006*.

S. Staab and R. Studer. 2004. *Handbook on ontologies*. Springer.

Fabian Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2008. Yago - a large ontology from Wikipedia and WordNet. *Journal of Web Semantics*.

Fabian Suchanek, Mauro Sozio, and Gerhard Weikum. 2009. Sofie: A self-organizing framework for information extraction. In *Proceedings of the Eighteenth International Conference on World Wide Web*.

Jun-ichi Tsujii. 2004. Thesaurus or logical ontology, which do we need for mining text? In *Proceedings of the Language Resources and Evaluation Conference*.

Fei Wu and Daniel S. Weld. 2008. Automatically refining the wikipedia infobox ontology. In *Proceedings of the Seventeenth International Conference on World Wide Web*, Beijing, China.

Alexander Yates and Oren Etzioni. 2009. Unsupervised methods for determining object and relation synonyms on the web. *Journal of Artificial Intelligence Research*, 34:255–296.