Unsupervised learning of an IS-A taxonomy from a limited domain-specific corpus

Daniele Alfarone
Jesse Davis

Report CW 664, May 2014

KU Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Unsupervised learning of an IS-A taxonomy from a limited domain-specific corpus

Daniele Alfarone
Jesse Davis

Report CW 664, May 2014

Department of Computer Science, KU Leuven

Abstract

This report addresses the problem of learning a taxonomy from a given domain-specific text corpus. We propose a novel unsupervised algorithm for this problem. Its key contributions include a clustering-based inference approach that increases recall over surface patterns and a graph-based algorithm for detecting incorrect edges that improves precision. Our system induces the taxonomy simply by analyzing the provided corpus. Thus, the learned taxonomy is focused on the concepts that are relevant for the specific corpus. An empirical evaluation on five corpora demonstrates the utility of the system.

CR Subject Classification : I.2.6, I.2.7

Unsupervised learning of an IS-A taxonomy from a limited domain-specific corpus

Daniele Alfarone and **Jesse Davis**
Department of Computer Science, KU Leuven
Celestijnenlaan 200A - box 2402, 3001 Leuven, Belgium
{daniele.alfarone,jesse.davis}@cs.kuleuven.be

Abstract

This report addresses the problem of learning a taxonomy from a given domain-specific text corpus. We propose a novel unsupervised algorithm for this problem. Its key contributions include a clustering-based inference approach that increases recall over surface patterns and a graph-based algorithm for detecting incorrect edges that improves precision. Our system induces the taxonomy simply by analyzing the provided corpus. Thus, the learned taxonomy is focused on the concepts that are relevant for the specific corpus. An empirical evaluation on five corpora demonstrates the utility of the system.

1 Introduction

Hard NLP tasks that require deep text understanding, such as Question Answering (Harabagiu et al., 2000) and Textual Entailment (Geffet and Dagan, 2005), typically rely on *domain ontologies* to extract facts and reason over them. Taxonomies are considered "the backbone" of ontologies, as they organize all domain concepts hierarchically through *is-a* relations,[1] which enables sharing of information among related concepts.

Many handcrafted taxonomies have been built to capture both open-domain (e.g., WordNet) and domain-specific (e.g., MeSH, for the medical domain) knowledge. Yet, our knowledge is constantly evolving and expanding. Therefore, even domain-specific handcrafted taxonomies inevitably lack coverage and are expensive and time-consuming to keep up-to-date. This has motivated the interest in automatically learning taxonomies from text. Initially, systems focused on modifying existing taxonomies to incorporate new concepts (Widdows, 2003; Snow et al., 2006; Yang and Callan, 2009). Recently, there has been growing interest in automatically building entire taxonomies *from scratch*. Algorithms construct taxonomies by using either the Web (Kozareva and Hovy, 2010; Wu et al., 2012) or a combination of a domain-specific corpus and the Web (Navigli et al., 2011; Velardi et al., 2012; Yang, 2012).

At a high-level, these approaches apply variants of the following strategy. First, an initial set of concepts are identified either via (a variant of) Hearst patterns (Hearst, 1992) or by clustering similar terms. Second, precision is increased. This can be done by discarding facts with low redundancy on the Web, or alternatively by learning from resources such as WordNet. Finally, either they build a graph from all facts and prune it to maximize some quality measure, or they iteratively add facts to the taxonomy to maximize the likelihood of the current taxonomy.

In this report we present TAXIFY, a novel unsupervised approach that learns a taxonomy from a domain-specific corpus. This has the potential advantage of focusing the learned taxonomy on the most important concepts that appear in a specific corpus, while minimizing the risk that unrelated concepts extracted from irrelevant documents are included. TAXIFY first uses Hearst patterns to collect its initial set of *is-a* relations. Second, it improves recall through a clustering-based inference procedure for identifying additional *is-a* relations. This helps overcome the limitation that Hearst patterns occur infrequently, especially in small corpora, and fail to identify relevant concepts. Third, TAXIFY attempts to improve the taxonomy's precision through a novel approach for detecting in-

[1] Alternatively called "hyponym-hypernym relations".

correctly identified *is-a* relationships. Note that our system is domain-independent as it can extract a taxonomy from a collection of documents about any specific domain.[2]

Empirically, we evaluate the proposed approach on five real-world corpora, compare it with Kozareva and Hovy (2010) and Velardi et al. (2012), and show how the different components of TAXIFY affect the overall performance.

The contributions of this work can be summarized as follows:

- A novel unsupervised approach for learning a taxonomy from a limited domain-specific corpus;
- A clustering-based inference technique that increases the recall;
- A graph-based algorithm that detects incorrect edges in the taxonomy.

2 Taxonomy Learning

We first briefly introduce the basic terminology used throughout the rest of the document and then we describe our algorithm. A **concept** is a entity (either abstract or concrete) relevant for a certain domain, expressed as a simple noun (e.g., *dolphins*) or a noun phrase (e.g., *Siberian tiger*). When two concepts appear in an *is-a* relation (e.g., *mammals* $\rightarrow$ *fox*), we refer to the most-specific concept (e.g., *fox*) as the **subtype**, and to the broader one as the **supertype** (e.g., *mammals*).

Given a limited, unlabeled domain-specific corpus, TAXIFY learns a probabilistic *is-a* taxonomy. The taxonomy is modeled as a *directed, acyclic graph*, $G = (V, E)$ where V is a set of vertices, each denoting a concept, and E is a set of edges, each denoting an *is-a* relationship. An edge $(x, y) \in E$, written as $x \rightarrow y$, denotes that subtype $y \in V$ has supertype $x \in V$. Using a graph instead of a *tree* allows a concept to have multiple supertypes, which reflects how humans classify objects.

At a high-level, TAXIFY works in four phases. First, an initial set of *is-a* relations are identified and added to the taxonomy. Second, recall is increased through a clustering-based inference procedure. Third, precision is improved by identifying and discarding incorrect edges. Fourth, a confidence value is attached to each fact. Next, we describe each step in more detail.

[2] TAXIFY's source code and online demo are available at http://people.cs.kuleuven.be/~daniele.alfarone/taxify

2.1 Constructing the initial taxonomy

TAXIFY constructs the initial taxonomy in three steps. First, it identifies a seed set of *is-a* relations. Second, it expands the coverage (i.e., recall) by adding additional *is-a* relations through syntactic inference. Third, it performs pruning to improve the taxonomy's precision.

Identify seed set of is-a relations. To identify an initial set of *is-a* relations, TAXIFY applies the Hearst patterns shown in Table 1 to the corpus. We did not learn additional patterns, as it has been previously shown not to be helpful (Ritter et al., 2009).

A problem with Hearst patterns is that the noun phrase (NP) immediately before/after the matched words is not always the correct supertype. To increase precision, some systems (Ritter et al., 2009; Wu et al., 2012) discard all *is-a* relations whose supertype appears in singular form. Since we want our system to also leverage weak (i.e., infrequent) evidence, we keep a window of two NPs and select the closest plural-form NP. This allows TAXIFY to correctly extract *certain animal species* $\rightarrow$ *penguins*, and not *strong corporate social organization* $\rightarrow$ *penguins* from the following sentence:

> *Certain animal species exhibit strong corporate social organization, such as penguins.*

A second problem is that extracted concepts may contain *generic* modifiers, such as *certain* in the previous example, that overspecify *is-a* relations. As a handcrafted blacklist of modifiers may not generalize well across different corpora, TAXIFY computes a domain-specificity score for each word w:

$$ds(w) = \frac{f_{corpus}(w)}{f_{Eng}(w)} \cdot \frac{1}{\log n_{Eng}(w)} \qquad (1)$$

where $n_{Eng}(w)$ and $f_{Eng}(w)$ are the absolute and relative frequency of w in English as approximated by the Google Ngram frequency (Jean-Baptiste Michel and Aiden, 2010). Then, TAXIFY reduces each concept to a *canonical form* by processing the modifiers in the concept from left to right until it encounters the first modifier w such that $ds(w) > \alpha_1$, where α_1 is a user-defined parameter. All modifiers before w are discarded from the concept. After canonicalizing both a subtype y and its corresponding supertype x, the edge $x \rightarrow y$ is added to G. In our running example, the edge *animal species* $\rightarrow$ *penguins* would be added.

x such as $\{y_i,\}$* {(or|and) $y_n\}$
x including $\{y_i,\}$* {(or|and) $y_n\}$
y_1 {, $y_i\}$* and other x
y_1 {, $y_i\}$* or other x

Table 1: Hearst patterns used by TAXIFY. Both *subtypes* (y_i) and their *supertype* (x) must be NPs.

Expand coverage by syntactic inference. TAXIFY identifies additional relationships to include in the taxonomy by performing syntactic inference on concepts containing modifiers (e.g., *marine animal*) as these represent a specialization of another concept. Thus, when TAXIFY adds a concept of the form *<modifier> <head>* to the taxonomy, it also adds *<head>* $\rightarrow$ *<modifier> <head>*. Note that this process accounts for nested NPs and thus can infer *inhibitor* $\rightarrow$ *inhibitor of CYP3A4* from *inhibitor of CYP3A4*. While simple, this rule achieves an accuracy above 90%. Further improvements would require domain knowledge.[3]

TAXIFY further expands the coverage of the taxonomy by, for each non-leaf concept x, scanning the corpus to find all NPs whose head is x and adds them to the taxonomy as a subtype of x.

Improve precision by domain-specific filtering. One way to increase the precision of a taxonomy is to apply domain-specific filtering (Liu et al., 2005; Navigli et al., 2011). Given a domain-specificity threshold α_2, TAXIFY removes all single-word concepts c from the taxonomy where $ds(c) < \alpha_2$ (ds is computed by Equation (1)).

2.2 Inferring novel facts

Since Hearst patterns occur infrequently, many interesting concepts will not be extracted. One way to improve recall is to search for semantically-related concepts within the corpus. A well-studied solution is to exploit concepts that *co-occur* in lists (Cederberg and Widdows, 2003; Snow et al., 2004; Davidov and Rappoport, 2006). However, co-occurrence does not imply that two concepts have the same immediate supertype. For example, knowing *capital* $\rightarrow$ *Rome* and that *Rome* is similar to *Munich*, may lead to the erroneous inference *capital* $\rightarrow$ *Munich*.

To overcome this limitation, TAXIFY clusters the concepts and then performs inference over *clusters* of similar concepts, instead of *pairs*. For example, if *city* $\rightarrow$ *Lyon* is also observed, Figure 1 shows how clustering *Rome*, *Lyon* and *Munich* together allows to correctly infer *city* $\rightarrow$ *Munich*.

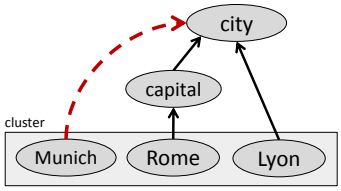

Figure 1: TAXIFY infers that *Munich* is a *city* (dotted edge), and not a *capital*, by clustering it together with *Rome* and *Lyon*.

Algorithm 1 formalizes our approach. First, to discover additional concepts and measure concept similarity, coordination patterns[4] are applied to the entire corpus. The similarity between two concepts is then computed as:

$$sim(c_1, c_2) = \frac{2 \cdot n(c_1 \cap c_2)}{n(c_1) + n(c_2)} \log \min(n(c_1), n(c_2)) \quad (2)$$

where $n(c_i)$ is the number of times c_i appears in a list of concepts, and $n(c_1 \cap c_2)$ is the number of times c_1 and c_2 appear in the same list. The values are then normalized to be between $[0, 1]$ w.r.t. the maximum observed value.

Next, the *K-Medoids algorithm* (Kaufman and Rousseeuw, 1987) clusters the concepts using the above similarity measure. Each cluster is checked to see if it contains at least two known concepts[5] (i.e., have a supertype in our taxonomy) and at least one unknown concept (i.e., found by coordination pattern). For each such cluster, Algorithm 1 finds the *Lowest Common Ancestor* (LCA) among the known concepts C_{known} and assigns this as the supertype to all unknown concepts C_{unk} in the cluster. This procedure is repeated several times to minimize the impact of the initial random seed selection in K-Medoids. At the end of this iterative procedure, all inferred edges are added to G.

Since Algorithm 1 needs to walk the graph, before executing it we break all cycles, if present.

[3] For instance, from *transcription factor AP-1* we erroneously infer *factor AP-1* $\rightarrow$ *transcription factor AP-1*, instead of *transcription factor* $\rightarrow$ *AP-1*.

[4] A coordination pattern matches a list of concepts, such as "*A*, *B* and *C*".

[5] We require that each cluster contains two known concepts as this further constrains the problem and helps avoid incorrect inferences such as in our example about *Rome* and *Munich*.

Algorithm 1: INFERNEWEDGES($G, corpus$)

$A \leftarrow$ *empty list of edges*
$matrix \leftarrow buildSimMatrix(corpus)$
foreach *clustering iteration* **do**
  $\{C_1, \ldots, C_n\} \leftarrow cluster(matrix)$
  foreach *cluster* $C \in \{C_1, \ldots, C_n\}$ **do**
    $(C_{known}, C_{unk}) \leftarrow separate(C)$
    if $|C_{known}| > 1 \wedge |C_{unk}| > 0$ **then**
      $lca \leftarrow findLca(C_{known}, G)$
      if *lca exists* **then**
        foreach *concept* $c \in C_{unk}$ **do**
          add edge $lca \rightarrow c$ to A
        end
      end
    end
  end
end
add all edges in A to G

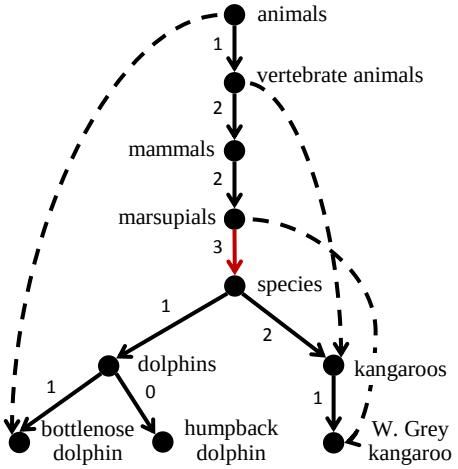

Figure 2: An excerpt of our unpruned animal taxonomy, which includes the wrong edge *marsupials* → *species*. The edge weight represents the number of times that the edge is part of a long path covered by a dotted edge.

2.3 Detecting incorrect edges

Learned taxonomies contain incorrect edges. TAXIFY attempts to detect incorrect edges in an unsupervised fashion by exploiting an observation made by Kozareva and Hovy (2010), stating that "*humans typically exemplify concepts using more proximate ones*". For example, "*mammals such as bottlenose dolphins*" should appear more frequently than "*organisms such as bottlenose dolphins*", even though both are true.

We leverage this observation to assume that it is unlikely for a taxonomy to contain a *long* path connecting x and y, if $x \rightarrow y$ was extracted by a Hearst pattern. If it does, it increases the chance that one of the edges in this long path is incorrect. Thus, the probability that an edge is incorrect goes up each time it appears in a long path between two concepts extracted by a Hearst pattern.

Figure 2, a portion of our learned animal taxonomy, illustrates this. The red edge is the best candidate for exclusion from the taxonomy, because it belongs to the three paths covered by the three dotted edges (representing Hearst pattern extractions).

Our procedure for detecting incorrect edges is outlined in Algorithm 2. As input it receives the taxonomy G, the flat list of all its edges E, and a threshold β that discriminates between short and long paths. The procedure loops through the following steps until no edge is covered by more than ten long paths. First, it identifies the set of *indirect Hearst edges*, which consists of each edge $(x, y) \in E$ extracted through Hearst patterns for which $\exists\ x' \in V$ such that $(x, x'), (x', y) \in E$. Second, it counts how many times each edge appears in a path longer than β which is covered by an indirect Hearst edge. Third, it removes the highest-count edge from the taxonomy, together with all edges extracted from the same context (i.e., the same Hearst pattern in the same sentence).

2.4 Assigning confidences to edges

When assigning a confidence to each edge, TAXIFY differentiates between edges extracted by a Hearst pattern and edges inferred by Algorithm 1.

Extracted edges Following NELL (Carlson et al., 2010), the initial confidence value of each extracted edge is $1 - 0.5^n$, where n is the number of times the edge was extracted by a Hearst pattern. This models the fact that the more times an edge is extracted, more likely it is to be correct. However, in limited corpora many relevant relationships will only be extracted once.[6]

To overcome this issue, we exploit the following intuition. Assume that *mammals* → *deer*, *mammals* → *otter*, and *mammals* → *fox* are extracted by the same Hearst pattern in the same sentence. The

[6] 92% – 98% of all edges in the analyzed corpora.

Algorithm 2: DETECTWRONGEDGES(G, β, E)

```
m ← empty map of <edge, counter>
E_ih ← getIndirectHearstEdges(E, G)
foreach edge e ∈ E_ih do
    P ← getAllPathsCoveredBy(e, G)
    foreach path p ∈ P s.t. size(p) > β do
        foreach edge e_covered ∈ p do
            inc. counter in m at key e_covered
        end
    end
end
e_max ← getHighestCountEdge(m)
if getCount(e_max, m) ≥ 10 then
    remove e_max from G
    E_context ← getContextualEdges(e_max)
    remove all edges in E_context from G
    DETECTWRONGEDGES(G, β, E_ih)
else
    remove all edges in E_ih from G
end
```

first two edges are observed only once, while *mammals* $\rightarrow$ *fox* is extracted multiple times. Intuitively, the additional observations of *mammals* $\rightarrow$ *fox* give further evidence of the correctness of the first two extractions. To capture this intuition, TAXIFY iteratively updates each extracted edge's confidence value to be that of the highest-confidence edge in the sentence.

Inferred edges Each inferred edge, $lca \rightarrow c$, receives the following confidence value:

$$p(lca \rightarrow c) = \max_i p_i(lca \rightarrow c) \quad (3)$$

where i ranges over the clustering iterations where $lca \rightarrow c$ was inferred, and p_i is defined as:

$$p_i(lca \rightarrow c) = \frac{1}{n} \sum_{k=1}^{n} P_{lca \Rightarrow c_k} \cdot sim(c_k, c) \quad (4)$$

where lca is the lowest common ancestor of the known concepts $\{c_1, \ldots, c_n\}$ that appear in the same cluster as c, $sim(\cdot)$ is defined by Equation 2 and $P_{lca \Rightarrow c_k}$ is the product of the edge confidences on the path from lca to c_k.

Intuitively, $p_i(lca \rightarrow c)$ is a similarity-weighted average over paths connecting each known concept c_k to lca. This is similar in spirit to how Snow et al. (2004) update an edge's probability based on *coordinate concepts*, i.e., concepts that share a common ancestor in the graph.

3 Evaluation

The goal of the empirical evaluation is to address the following two questions:

1. How does our approach compare to other state-of-the-art algorithms on this task?
2. What is the effect of each of the system's components on its overall performance?

To answer these questions, we use five real-world plain-text corpora from different domains.

3.1 Methodology

Taxonomy evaluation is a hard task, as significantly different taxonomies can be equally correct in modeling a domain. Moreover, domains can be modeled at various levels of specificity. Furthermore, gold standards are typically not available. Therefore, in the literature taxonomies are evaluated by (1) reconstructing an existing taxonomy or (2) performing manual evaluation. We perform manual evaluation.

For each learned taxonomy, we report *precision* and *number of correct facts* (relative recall), as true recall requires a gold standard.[7] Our taxonomies are too large to perform an exhaustive manual evaluation. Therefore, we divided the *is-a* relations into bins based on their confidence, and estimated the precision and the number of correct facts in each bin using a random sample, as is commonly done (e.g., Schoenmackers et al., 2010).

An edge $x \rightarrow y$ is considered correct if (1) a domain expert would accept the sentence "y *is an* x", and (2) y and x are both significant concepts for the given domain. This means that, for instance, the edge *trees* $\rightarrow$ *oak* would be marked as wrong when evaluating an animal taxonomy.

Parameter setting We used a validation set to determine the relevant parameters. For Section 2.1, we set $\alpha_1 = 0.4$ and $\alpha_2 = 1.68$ by looking at a small set of manually labeled words. For concept filtering the threshold needs to be stricter, because many words are significant as a concept modifier (e.g., *Pink* Pigeon), but not as a single-word concept (e.g., *pink*).

We set k for K-Medoids (Algorithm 1) for each corpus at run-time, as the number of concepts to

[7] Note that, given systems A and B, the ratio of their relative recalls is equal to the ratio of their true recalls, so statements like "*A has double the recall of B*" are still valid.

cluster divided by a parameter ψ. On each iteration, the algorithm varies this value slightly. We set $\psi = 5$ and did not investigate other values.

Finally, to detect incorrect edges (Algorithm 2), a corpus-specific value for β is required, as it should depend on the domain's complexity. Given an unpruned taxonomy, TAXIFY first stores for each indirect Hearst edge the size of the shortest path covered in the taxonomy. Given all path sizes, it sets $\beta = average\_size + 2 \cdot \sigma$, a standard formula for *outlier detection* (Healy, 1979).[8]

3.2 Comparison with state of the art

The most relevant systems for comparison are *K&H* (Kozareva and Hovy, 2010) and *OntoLearn* (Velardi et al., 2012), as they both extract *domain-specific* taxonomies entirely from scratch. All comparisons with *OntoLearn* are made against their best run (*DAG*[0,99]).

Experiment 1 In our first experiment, we compared taxonomies learned from two biomedical corpora, DDI[9] and PMC,[10] for which *OntoLearn*'s results are publicly available.[11] To compare with *K&H*, we re-implemented their system and ran it on the same corpora. Since their bootstrapping approach terminates immediately in limited corpora, we relaxed some constraints to obtain a more meaningful comparison.[12]

As *K&H* also requires root concepts in input, we picked *drug*, *medication*, *agent* for DDI, and *disease*, *gene*, *protein*, *cell* for PMC. Figure 3a shows results both for taxonomies rooted at these concepts, and the unrooted taxonomies, since TAXIFY and *OntoLearn* do not need domain-specific inputs.

On the rooted taxonomies, *OntoLearn* performs poorly. TAXIFY has lower precision than *K&H* on PMC, but their system's recall is almost two orders of magnitude lower. *K&H*'s approach is indeed very "conservative", as it is conceived for working at Web scale. On the other hand, TAXIFY's lower precision on PMC is caused by few incorrect edges that link several irrelevant concepts to the taxonomy. Even though Algorithm 2 mitigates this problem, on large corpora not tied to a narrow domain this remains harmful.

On the unrooted taxonomies, TAXIFY shows high precision. *OntoLearn* achieves very high recall on DDI, but its precision is low, mainly because it extracts several vague/generic concepts (e.g., *time*, *cause*, *rate*, *degree*).

Experiment 2 We compare on the *Animals*, *Plants* and *Vehicles* taxonomies, since results are available for all systems. As our system does not work with the Web, we created three corpora by selecting from Wikipedia all the article abstracts that contain the words *animals*, *plants*, and *vehicles*, respectively. While the results (Figure 3b) are not an apples-to-apples comparison, they still offer some insight.

Unfortunately, as noted by *OntoLearn*, *K&H*'s Web-run precision is not directly comparable, as the evaluation is only performed for nodes that appear in a certain WordNet sub-hierarchy. Anyway, surprisingly, the results on the Web runs show that achieving high recall remains a central issue, even when using the Web as a corpus.

On Wikipedia, TAXIFY outperforms *K&H*'s re-implementation in both precision and recall.

3.3 Evaluation of TAXIFY's subcomponents

We performed an ablation study on *Animals* and *Vehicles*, to analyze the impact of the different components of TAXIFY on the overall performance. The empirical results in Figure 4 show that our clustering-based inference (Section 2.2) increases relative recall by 57% on *Vehicles* and 44% on *Animals*.

Our detection of incorrect edges (Section 2.3) boosts up precision by 9.7% on *Vehicles* and 18.7% on *Animals*. In terms of average depth, the *Animals* taxonomy is reduced from 7.8 to 3.35, and the *Vehicles* taxonomy from 6.05 to 3.37. The only corpus for which Algorithm 2 removes no edges is DDI, as its small size and high data density already yield a high-precision taxonomy.

Finally, the simple syntactic inference significantly increases our recall. Syntactically-inferred edges are fundamental to ensure the connectivity in the taxonomy (e.g., knowing that *bottlenose dolphin* and *humpback dolphin* are specializations of the same concept) and to discover extra concepts (e.g., including *helodermatid lizards*, even though neither Hearst patterns nor the clustering-based inference

[8] More robust, median-based outlier detection techniques cannot be applied, because in most cases the median absolute deviation (MAD) is null.

[9] http://www.cs.york.ac.uk/semeval-2013/task9/

[10] http://www.ncbi.nlm.nih.gov/pmc/

[11] http://www.ontolearn.org/

[12] In detail, (1) we provided ten seed terms instead of one, (2) we relaxed their patterns to match more than two subtypes, and (3) included basic syntactic inference.

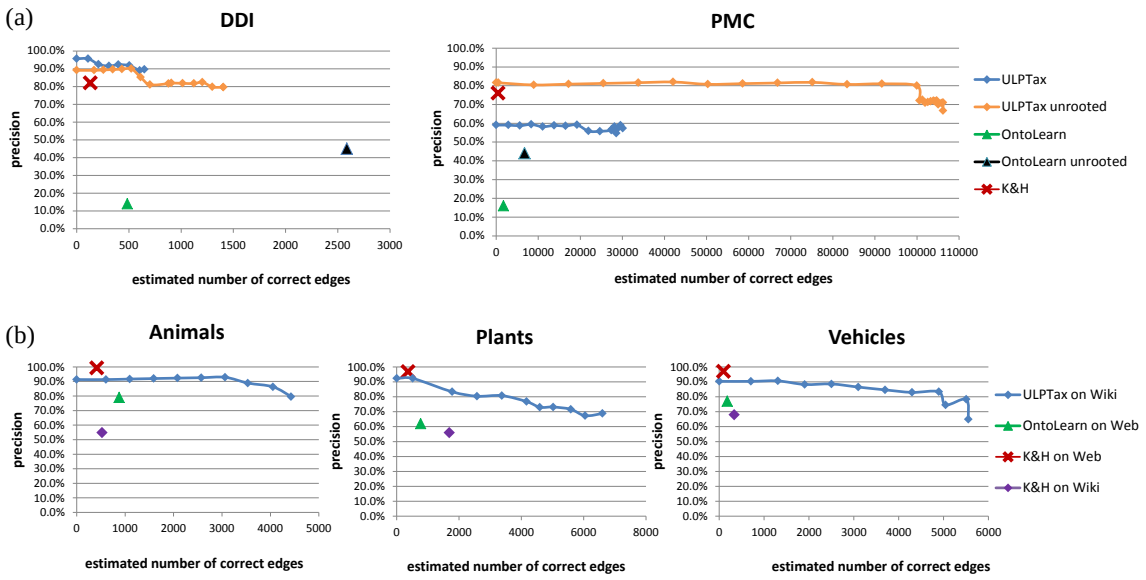

Figure 3: Comparison of TAXIFY with Kozareva and Hovy (2010) and Velardi et al. (2012). Their results are plotted as single dots because no confidence scores are provided.

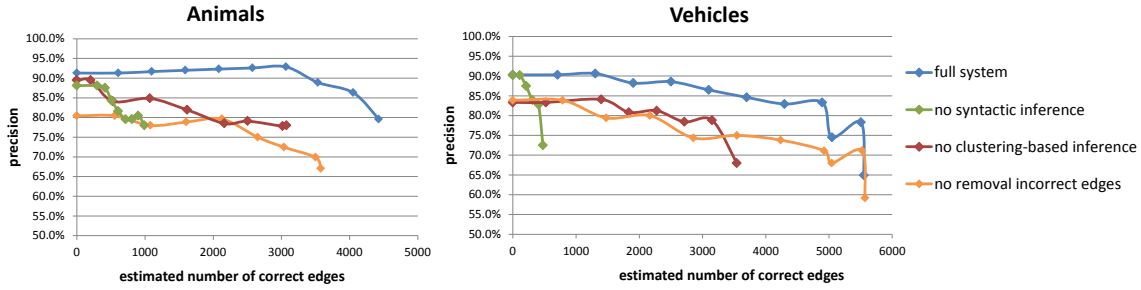

Figure 4: Ablation study for TAXIFY.

could find its supertype).

3.4 Comparison with an existing taxonomy

As a last experiment, we evaluated our drug taxonomy extracted from DDI against *MeSH*,[13] a handcrafted biomedical taxonomy. Table 2 shows that out of 677 correct edges extracted by TAXIFY, only 223 appear in the MeSH taxonomy, suggesting that MeSH cannot be used a gold standard.

The facts not appearing in MeSH can be classified into those that *refine* MeSH, and those that *extend* it. We say that two edges $x \rightarrow x'$ and $x' \rightarrow y$ refine the MeSH taxonomy if the edge $x \rightarrow y$ appears in MeSH. In other words, the concept x' increases the level of detail of the taxonomy. All other edges extend MeSH, by either introducing a new concept or assigning an additional supertype to a known concept.

[13] http://www.ncbi.nlm.nih.gov/mesh

4 Related Work

The task of taxonomy learning can be decomposed into concept extraction and concept organization. While earlier systems focused on the first task only (Ravichandran and Hovy, 2002; Chung, 2003; Liu et al., 2005), more recent efforts go towards tackling both tasks together, as TAXIFY does.

While many taxonomy learning approaches are based on Hearst patterns, one exception is proposed by Davidov and Rappoport (2006). This approach builds a graph connecting concepts that co-occur in a coordination pattern, and then uses graph connectivity to hierarchically cluster similar concepts. Their approach is similar to our clustering-based inference, with the difference that they do not attempt to discover a common supertype for each cluster, thus they cannot induce a taxonomy.

Evaluation	Count	Examples
Already in MeSH	223 (32.9%)	*non-steroidal anti-inflammatory drugs* → *aspirin*, *psychotropic agents* → *tranquilizers*
Refines MeSH	115 (17.0%)	*highly protein-bound drugs* → *captopril*, *oral anticoagulants* → *warfarin*
Extends MeSH	339 (50.1%)	*drugs* → *zileuton*, *TNF-blocking agents* → *HUMIRA*

Table 2: Comparison of the 677 correct facts of our *drug* taxonomy against the MeSH taxonomy.

More recently, Navigli et al. (2011) proposed extracting *is-a* relations from definition sentences. As definition sentences are typically less frequent than Hearst patterns, this approach requires documents from both a domain corpus and the Web. Furthermore, these sentences often extract very generic supertypes, which are irrelevant to the domain of interest, and thus limit the overall precision, as shown in our experiments.

A last exception is Yang (2012), which proposes a new concept similarity metric trained on WordNet and aims at building a taxonomy that maximizes the similarity of all concepts in root-to-leaf paths. However, they assume high data density, as they initially discard all infrequent concepts. On the other hand, TAXIFY builds a taxonomy that covers also concepts observed only once, thus achieving a high coverage even in small corpora.

Hearst patterns-based approaches often tackle taxonomy learning in two steps: (1) increase precision and recall of the extracted *is-a* relations, and (2) organize all relations in a taxonomy.

To increase recall, many systems rely on "*noun coordination*", which searches for coordinate concepts of a concept c (i.e., concepts similar to c) to assign c's supertype to them. Approaches for discovering coordinate concepts include using coordination patterns (Cederberg and Widdows, 2003; Snow et al., 2004), and exploiting distributional similarity through LSA (Snow et al., 2004; Snow et al., 2006) or HMMs (Ritter et al., 2009). Note that recall is not a main concern for systems that access the Web (Kozareva and Hovy, 2010; Navigli et al., 2011; Wu et al., 2012). Our approach makes use of coordination patterns. In contrast to existing work, which considers *pairs* of concepts (c_1, c_2), we consider *clusters* of concepts, as knowing that c_1 is coordinated with c_2 does not imply that their immediate supertype is the same.

Many different techniques have been proposed for increasing precision at local level. One line of work looks at training a classifier, typically with *is-a* relations from WordNet, to combine Hearst patterns with weaker evidence (Snow et al., 2004; Snow et al., 2006; Ritter et al., 2009). Other approaches include accepting a certain *is-a* relation only if its arguments are semantically related according to LSA (Cederberg and Widdows, 2003), or using negative evidence, i.e., extracting information that contradicts a certain *is-a* relation, for instance a *part-of* relation (Ponzetto and Strube, 2011). In contrast, TAXIFY simply filters out supertypes appearing in singular form, and all concepts with low domain specificity.

Finally, the precision can be further improved by considering global information when organizing the *is-a* relations into a taxonomy. Snow et al. (2006) and Wu et al. (2012) populate the taxonomy *iteratively*: a set of relations is added only if there is enough confidence in doing so. More similarly to us, some other approaches first build a dense graph, and then prune it by retaining longest paths only (Kozareva and Hovy, 2010) or by finding the optimal trade-off between long paths and the connectivity of traversed nodes (Navigli et al., 2011). Instead of blindly giving more credit to longer paths, our idea is to check whether their length is justified by the domain, or only an artifact caused by some incorrect edges (Section 2.3).

5 Conclusions

We described TAXIFY, a system for learning a probabilistic taxonomy from scratch by only accessing a limited domain-specific corpus. Empirical results show how our approach outperforms two existing systems in this setting.

Since no supervision is required, TAXIFY can run on any domain-specific set of plain-text documents, and its probabilistic output allows further probabilistic reasoning for a variety of NLP tasks, such as inference rule learning, ontology learning and question answering.

Though it is very interesting to see how a taxonomy learner can perform well in such a restricted setting, in future work we will investigate how our system can leverage the full Web. Additionally, we are interested in exploiting the evidence provided by previously extracted facts to better extract new facts, iteratively.

References

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*.

Scott Cederberg and Dominic Widdows. 2003. Using lsa and noun coordination information to improve the precision and recall of automatic hyponymy extraction. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 111–118. Association for Computational Linguistics.

Teresa Mihwa Chung. 2003. A corpus comparison approach for terminology extraction. *Terminology: international journal of theoretical and applied issues in specialized communication*, 9(2):221–246.

Dmitry Davidov and Ari Rappoport. 2006. Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 297–304. Association for Computational Linguistics.

Maayan Geffet and Ido Dagan. 2005. The distributional inclusion hypotheses and lexical entailment. In *In Proceedings of ACL-2005. Ann Arbor*, pages 107–114.

Sanda M Harabagiu, Marius A Paşca, and Steven J Maiorano. 2000. Experiments with open-domain textual question answering. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 292–298. Association for Computational Linguistics.

MJ Healy. 1979. Outliers in clinical chemistry quality-control schemes. *Clinical chemistry*, 25(5):675–677.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics.

Aviva Presser Aiden Adrian Veres Matthew K. Gray William Brockman The Google Books Team Joseph P. Pickett Dale Hoiberg Dan Clancy Peter Norvig Jon Orwant Steven Pinker Martin A. Nowak Jean-Baptiste Michel, Yuan Kui Shen and Erez Lieberman Aiden. 2010. Quantitative analysis of culture using millions of digitized books. In *Science*.

Leonard Kaufman and Peter Rousseeuw. 1987. Clustering by means of medoids.

Zornitsa Kozareva and Eduard Hovy. 2010. A semi-supervised method to learn and construct taxonomies using the web. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1110–1118. Association for Computational Linguistics.

Tao Liu, XL Wang, Y Guan, ZM Xu, et al. 2005. Domain-specific term extraction and its application in text classification. In *8th Joint Conference on Information Sciences*, pages 1481–1484.

Roberto Navigli, Paola Velardi, and Stefano Faralli. 2011. A graph-based algorithm for inducing lexical taxonomies from scratch. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 1872–1877. AAAI Press.

Simone Paolo Ponzetto and Michael Strube. 2011. Taxonomy induction based on a collaboratively built knowledge repository. *Artificial Intelligence*, 175(9):1737–1756.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 41–47. Association for Computational Linguistics.

Alan Ritter, Stephen Soderland, and Oren Etzioni. 2009. What is this, anyway: Automatic hypernym discovery. In *Proceedings of the 2009 AAAI Spring Symposium on Learning by Reading and Learning to Read*, pages 88–93.

Stefan Schoenmackers, Oren Etzioni, Daniel S Weld, and Jesse Davis. 2010. Learning first-order horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1088–1098. Association for Computational Linguistics.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems 17*.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 801–808. Association for Computational Linguistics.

Paola Velardi, Stefano Faralli, and Roberto Navigli. 2012. Ontolearn reloaded: A graph-based algorithm for taxonomy induction.

Dominic Widdows. 2003. Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 197–204. Association for Computational Linguistics.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. 2012. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 international conference on Management of Data*, pages 481–492. ACM.

Hui Yang and Jamie Callan. 2009. A metric-based framework for automatic taxonomy induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 271–279. Association for Computational Linguistics.

Hui Yang. 2012. Constructing task-specific taxonomies for document collection browsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1278–1289. Association for Computational Linguistics.