Learning to Exploit Structured Resources for Lexical Inference

Vered Shwartz† **Omer Levy**† **Ido Dagan**† **Jacob Goldberger**§
† Computer Science Department, Bar-Ilan University
§ Faculty of Engineering, Bar-Ilan University
vered1986@gmail.com {omerlevy,dagan}@cs.biu.ac.il
jacob.goldberger@biu.ac.il

Abstract

Massive knowledge resources, such as Wikidata, can provide valuable information for lexical inference, especially for proper-names. Prior resource-based approaches typically select the subset of each resource's relations which are relevant for a particular given task. The selection process is done manually, limiting these approaches to smaller resources such as WordNet, which lacks coverage of proper-names and recent terminology. This paper presents a supervised framework for automatically selecting an optimized subset of resource relations for a given target inference task. Our approach enables the use of large-scale knowledge resources, thus providing a rich source of high-precision inferences over proper-names.[1]

1 Introduction

Recognizing lexical inference is an important component in semantic tasks. Various lexical-semantic relations, such as synonomy, class-membership, part-of, and causality may be used to infer the meaning of one word from another, in order to address lexical variability. For instance, a question answering system asked "which artist's net worth is $450 million?" might retrieve the candidates $Beyoncé\ Knowles$ and $Lloyd\ Blankfein$, who are both worth $450 million. To correctly answer the question, the application needs to know that Beyoncé is an artist, and that Lloyd Blankfein is not.

[1] Our code and data are available at: https://github.com/vered1986/LinKeR

Corpus-based methods are often employed to recognize lexical inferences, based on either co-occurrence patterns (Hearst, 1992; Turney, 2006) or distributional representations (Weeds and Weir, 2003; Kotlerman et al., 2010). While earlier methods were mostly unsupervised, recent trends introduced supervised methods for the task (Baroni et al., 2012; Turney and Mohammad, 2015; Roller et al., 2014). In these settings, a targeted lexical inference relation is implicitly defined by a training set of term-pairs, which are annotated as positive or negative examples of this relation. Several such datasets have been created, each representing a somewhat different flavor of lexical inference.

While corpus-based methods usually enjoy high recall, their precision is often limited, hindering their applicability. An alternative common practice is to mine high-precision lexical inferences from structured resources, particularly WordNet (Fellbaum, 1998). Nevertheless, WordNet is an ontology of the English language, which, by definition, does not cover many proper-names ($Beyoncé \rightarrow artist$) and recent terminology ($Facebook \rightarrow social\ network$). A potential solution may lie in rich and up-to-date structured knowledge resources such as Wikidata (Vrandečić, 2012), DBPedia (Auer et al., 2007), and Yago (Suchanek et al., 2007). In this paper, we investigate how these resources can be exploited for lexical inference over proper-names.

We begin by examining whether the common usage of WordNet for lexical inference can be extended to larger resources. Typically, a subset of WordNet relations is manually selected (e.g. all synonyms and hypernyms). By nature, each application captures a different aspect of lexical inference, and thus defines different relations as *indicative* of its particular flavor of lexical infer-

Resource Relation	Example
position_held	David Petraeus → Director of CIA
performer	Sheldon Cooper → Jim Parsons
operating_system	iPhone → iOS

Table 1: Examples of Wikidata relations that are indicative of lexical inference.

ence. For instance, the $hypernym$ relation is indicative of the is_a flavor of lexical inference (e.g. $musician \rightarrow artist$), but does not indicate causality.

Since WordNet has a relatively simple schema, manually finding such an optimal subset is feasible. However, structured knowledge resources' schemas contain thousands of relations, dozens of which may be indicative. Many of these are not trivial to identify by hand, as shown in Table 1. A manual effort to construct a distinct subset for each task is thus quite challenging, and an automated method is required.

We present a principled supervised framework, which automates the selection process of resource relations, and optimizes this subset for a given target inference relation. This automation allows us to leverage large-scale resources, and extract many high-precision inferences over proper-names, which are absent from WordNet. Finally, we show that our framework complements state-of-the-art corpus-based methods. Combining the two approaches can particularly benefit real-world tasks in which proper-names are prominent.

2 Background

2.1 Common Use of WordNet for Inference

WordNet (Fellbaum, 1998) is widely used for identifying lexical inference. It is usually used in an unsupervised setting where the relations relevant for each specific inference task are manually selected a priori.

One approach looks for chains of these predefined relations (Harabagiu and Moldovan, 1998), e.g. $dog \rightarrow mammal$ using a chain of hypernyms: $dog \rightarrow canine \rightarrow carnivore \rightarrow placental\ mammal \rightarrow mammal$. Another approach is via WordNet Similarity (Pedersen et al., 2004), which takes two synsets and returns a numeric value that represents their similarity based on WordNet's hierarchical hypernymy structure.

While there is a broad consensus that synonyms entail each other ($elevator \leftrightarrow lift$) and hyponyms entail their hypernyms ($cat \rightarrow animal$), other relations, such as meronymy, are not agreed upon, and may vary depending on task and context (e.g. $living\ in\ London \rightarrow living\ in\ England$, but $leaving\ London \not\rightarrow leaving\ England$). Overall, there is no principled way to select the subset of relevant relations, and a suitable subset is usually tailored to each dataset and task. This work addresses this issue by automatically learning the subset of relations relevant to the task.

Resource	#Entities	#Properties	Version
DBPedia	4,500,000	1,367	July 2014
Wikidata	6,000,000	1,200	July 2014
Yago	10,000,000	70	December 2014
WordNet	150,000	13	3.0

Table 2: Structured resources explored in this work.

2.2 Structured Knowledge Resources

While WordNet is quite extensive, it is handcrafted by expert lexicographers, and thus cannot compete in terms of scale with community-built knowledge bases such as Wikidata (Vrandečić, 2012), which connect millions of entities through a rich variety of structured relations (properties).

Using these resources for various NLP tasks has become exceedingly popular (Wu and Weld, 2010; Rahman and Ng, 2011; Unger et al., 2012; Berant et al., 2013). Little attention, however, was given to leveraging them for identifying lexical inference; the exception being Shnarch et al. (2009), who used structured data from Wikipedia for this purpose.

In this paper, we experimented with such resources, in addition to WordNet. *DBPedia* (Auer et al., 2007) contains structured information from Wikipedia: info boxes, redirections, disambiguation links, etc. *Wikidata* (Vrandečić, 2012) contains facts edited by humans to support Wikipedia and other Wikimedia projects. *Yago* (Suchanek et al., 2007) is a semantic knowledge base derived from Wikipedia, WordNet, and GeoNames.[2]

Table 2 compares the scale of the resources we used. The massive scale of the more recent resources and their rich schemas can potentially increase the coverage of current WordNet-based approaches, yet make it difficult to manually select an optimized subset of relations for a task. Our method automatically learns such a subset, and provides lexical inferences on entities that are absent from WordNet, particularly proper-names.

[2]We also considered Freebase, but it required significantly larger computational resources to work in our framework, which, at the time of writing, exceeded our capacity. §4.1 discusses complexity.

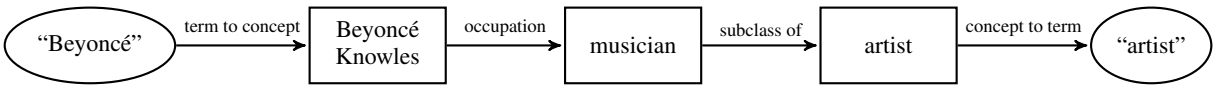

Figure 1: An excerpt of a resource graph (Wikidata) connecting "Beyoncé" to "artist". Resource graphs contain two types of nodes: terms (ellipses) and concepts (rectangles).

3 Task Definition and Representation

We wish to leverage the information in structured resources to identify whether a certain lexical-inference relation $\mathcal{R}$ holds between a pair of terms. Formally, we wish to classify whether a term-pair (x, y) satisfies the relation $\mathcal{R}$. $\mathcal{R}$ is implicitly defined by a training set of (x, y) pairs, annotated as positive or negative examples. We are also given a set of structured resources, which we will utilize to classify (x, y).

Each resource can be naturally viewed as a directed graph G (Figure 1). There are two types of nodes in G: *term* (lemma) nodes and *concept* (synset) nodes. The edges in G are each labeled with a property (*edge type*), defining a wide range of semantic relations between concepts (e.g. occupation, subclass_of). In addition, terms are mapped to the concepts they represent via term-concept edge types.

When using multiple resources, G is a disconnected graph composed of a subgraph per resource, without edges connecting nodes from different resources. One may consider connecting multiple resource graphs at the term nodes. However, this may cause sense-shifts, i.e. connect two distinct concepts (in different resources) through the same term. For example, the concept $January\ 1^{st}$ in Wikidata is connected to the concept $fruit$ in WordNet through the polysemous term $date$. The alternative, aligning resources in the concept space, is not trivial. Some partial mappings exist (e.g. Yago-WordNet), which can be explored in future work.

4 Algorithmic Framework

We present an algorithmic framework for learning whether a term-pair (x, y) satisfies a relation $\mathcal{R}$, given an annotated set of term-pairs and a resource graph G. We first represent (x, y) as the set of paths connecting x and y in G (§4.1). We then classify each such path as indicative or not of $\mathcal{R}$, and decide accordingly whether $x\mathcal{R}y$ (§4.2).

4.1 Representing Term-Pairs as Path-Sets

We represent each (x, y) pair as the set of *paths* that link x and y within each resource. We retain only the shortest paths (all paths $x \rightsquigarrow y$ of minimal length) as they yielded better performance.

Resource graphs are densely connected, and thus have a huge branching factor b. We thus limited the maximum path length to $\ell = 8$ and employed bidirectional search (Russell and Norvig, 2009, Ch.3) to find the shortest paths. This algorithm runs two simultaneous instances of breadth-first search (BFS), one from x and another from y, halting when they meet in the middle. It is much more efficient, having a complexity of $O(b^{\ell/2}) = O(b^4)$ instead of BFS's $O(b^\ell) = O(b^8)$.

To further reduce complexity, we split the search to two phases: we first find all nodes along the shortest paths between x and y, and then reconstruct the actual paths. Searching for relevant nodes ignores edge types, inducing a simpler resource graph, which can be represented as a sparse adjacency matrix and manipulated efficiently with matrix operations (elaborated in appendix A). Once the search space is limited to relevant nodes alone, path-finding becomes trivial.

4.2 Classification Framework

We consider edge types that typically connect between concepts in $\mathcal{R}$ to be "indicative"; for example, the $occupation$ edge type is indicative of the is_a relation, as in "Beyoncé is_a musician". Our framework's goal is to learn which edge types are indicative of a given relation $\mathcal{R}$, and use that information to classify new (x, y) term-pairs.

Figure 2 presents the dependencies between edge types, paths, and term-pairs. As discussed in the previous section, we represent each term-pair as a set of paths. In turn, we represent each path as a "bag of edges", a vector with an entry for each edge type.[3] To model the edges' "indicativeness", we assign a parameter to each edge type, and learn these parameters from the term-pair level supervision provided by the training data.

In this work, we are not only interested in optimizing accuracy or F_1, but in exploring the entire recall-precision trade-off. Therefore, we optimize

[3] We add special markers to the first and last edges within each path. This allows the algorithm to learn that applying term-to-concept and concept-to-term edge types in the middle of a path causes sense-shifts.

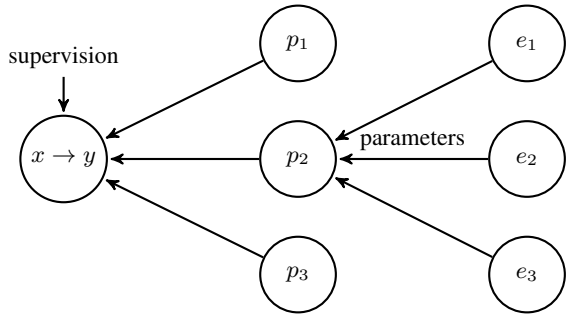

Figure 2: The dependencies between term-pairs ($x \rightarrow y$), paths (p_j), and edge types (e_i).

the F_β objective, where β^2 balances the recall-precision trade-off.[4] In particular, we expect structured resources to facilitate high-precision inferences, and are thus more interested in lower values of β^2, which emphasize precision over recall.

4.2.1 Weighted Edge Model

A typical neural network approach is to assign a *weight* w_i to each edge type e_i, where more indicative edge types should have higher values of w_i. The indicativeness of a path ($\hat{p}$) is modeled using logistic regression: $\hat{p} \triangleq \sigma(\vec{w} \cdot \vec{\phi})$, where $\vec{\phi}$ is the path's "bag of edges" representation, i.e. a feature vector of each edge type's frequency in the path.

The probability of a term-pair being positive can be determined using either the sum of all path scores or the score of its most indicative path (max-pooling). We trained both variants with back-propagation (Rumelhart et al., 1986) and gradient ascent. In particular, we optimized F_β using a variant of Jansche's (2005) derivation of F_β-optimized logistic regression (see suplementary material[5] for full derivation).

This model can theoretically quantify how indicative each edge type is of $\mathcal{R}$. Specifically, it can differentiate weakly indicative edges (e.g. meronyms) from those that contradict $\mathcal{R}$ (e.g. antonyms). However, on our datasets, this model yielded sub-optimal results (see §6.1), and therefore serves as a baseline to the binary model presented in the following section.

4.2.2 Binary Edge Model

Preliminary experiments suggested that in most datasets, each edge type is either indicative or non-indicative of the target relation $\mathcal{R}$. We therefore developed a binary model, which defines a global set of edge types that are indicative of $\mathcal{R}$: a *whitelist*.

Classification We represent each path p as a binary "bag of edges" ϕ, i.e. the set of edge types that were applied in p. Given a term-pair (x, y) represented as a path-set $paths(x, y)$, and a whitelist w, the model classifies (x, y) as positive if:

$$\exists \phi \in paths(x, y) : \phi \subseteq w \quad (1)$$

In other words:

1. A *path* is classified as indicative if all its edge types are whitelisted.
2. A *term-pair* is classified as positive if at least one of its paths is indicative.[6]

The first design choice essentially assumes that $\mathcal{R}$ is a transitive relation. This is usually the case in most inference relations (e.g. hypernymy, causality). In addition, notice that the second modeling assumption is unidirectional; in some cases $x\mathcal{R}y$, yet an indicative path between them does not exist. This can happen, for example, if the relation between them is not covered by the resource, e.g. causality in WordNet.

Training Learning the optimal whitelist over a training set can be cast as a subset selection problem: given a set of possible edge types $E = \{e_1, ..., e_n\}$ and a utility function $u : 2^E \rightarrow \mathbb{R}$, find the subset (whitelist) $w \subseteq E$ that maximizes the utility, i.e. $w^* = \arg\max_w u(w)$. In our case, the utility u is the F_β score over the training set.

Structured knowledge resources contain hundreds of different edge types, making E very large, and an exhaustive search over its powerset infeasible. The standard approach to this class of subset selection problems is to apply local search algorithms, which find an approximation of the optimal subset. We tried several local search algorithms, and found that genetic search (Russell and Norvig, 2009, Ch.4) performed well. In general, genetic search is claimed to be a preferred strategy for subset selection (Yang and Honavar, 1998).

In our application of genetic search, each individual (candidate solution) is a whitelist, represented by a bit vector with a bit for each edge type. We defined the fitness function of a whitelist w according to the F_β score of w over the training set.

[4] $F_\beta = \frac{(1+\beta^2) \cdot precision \cdot recall}{\beta^2 \cdot precision + recall}$

[5] http://u.cs.biu.ac.il/%7enlp/wp-content/uploads/LinKeR-sup.pdf

[6] As a corollary, if $x\mathcal{R}y$, then every path between them is non-indicative.

Dataset	#Instances	#Positive	#Negative
kotlerman2010	2,940	880	2,060
turney2014	1,692	920	772
levy2014	12,602	945	11,657
proper2015	1,500	750	750

Table 3: Datasets evaluated in this work.

We also applied L_2 regularization to reduce the fitness of large whitelists.

The binary edge model works well in practice, successfully replicating the common practice of manually selected relations from WordNet (see §6.1). In addition, the model outputs a human-interpretable set of indicative edges.

Although the weighted model's hypothesis space subsumes the binary model's, the binary model performed better on our datasets. We conjecture that this stems from the limited amount of training instances, which prevents a more general model from converging into an optimal solution.

5 Datasets

We used 3 existing common-noun datasets and one new proper-name dataset. Each dataset consists of annotated (x, y) term-pairs, where both x and y are noun phrases. Since each dataset was created in a slightly different manner, the underlying semantic relation $\mathcal{R}$ varies as well.

5.1 Existing Datasets

kotlerman2010 (Kotlerman et al., 2010) is a manually annotated lexical entailment dataset of distributionally similar nouns. turney2014 (Turney and Mohammad, 2015) is based on a crowdsourced dataset of semantic relations, from which we removed non-nouns and lemmatized plurals. levy2014 (Levy et al., 2014) was generated from manually annotated entailment graphs of subject-verb-object tuples. Table 3 provides metadata on each dataset.

Two additional datasets were created using WordNet (Baroni and Lenci, 2011; Baroni et al., 2012), whose definition of $\mathcal{R}$ can be trivially captured by a resource-based approach using WordNet. Hence, they are omitted from our evaluation.

5.2 A New Proper-Name Dataset

An important linguistic component that is missing from these lexical-inference datasets is proper-names. We conjecture that much of the added value in utilizing structured resources is the ability to cover terms such as celebrities (Lady Gaga) and recent terminology (social networks) that do not appear in WordNet. We thus created a new dataset of (x, y) pairs in which x is a proper-name, y is a common noun, and $\mathcal{R}$ is the is_a relation. For instance, $(Lady\ Gaga, singer)$ is true, but $(Lady\ Gaga, film)$ is false.

To construct the dataset, we sampled 70 articles in 9 different topics from a corpus of recent events (online magazines). As candidate (x, y) pairs, we extracted 24,000 pairs of noun phrases x and y that belonged to the same paragraph in the original text, selecting those in which x is a proper-name. These pairs were manually annotated by graduate students, who were instructed to use their world knowledge and the original text for disambiguation (e.g. $England \rightarrow team$ in the context of football). The agreement on a subset of 4,500 pairs was $\kappa = 0.954$.

After annotation, we had roughly 800 positive and 23,000 negative pairs. To balance the dataset, we sampled negative examples according to the frequency of y in positive pairs, creating "harder" negative examples, such as $(Sherlock, lady)$ and $(Kylie\ Minogue, vice\ president)$.

6 Results

We first validate our framework by checking whether it can automatically replicate the common manual usage of WordNet. We then evaluate it on the proper-name dataset using additional resources. Finally, we compare our method to state-of-the-art distributional methods.

Experimental Setup While F_1 is a standard measure of performance, it captures only one point on the recall-precision curve. Instead, we present the entire curve, while expecting the contribution of structured resources to be in the high-precision region. To create these curves, we optimized our method and the baselines using F_β with 40 values of $\beta^2 \in (0, 2)$.

We randomly split each dataset into 70% train, 25% test and 5% validation.[7] We applied L_2 regularization to our method and the baselines, tuning the regularization parameter on the validation set.

6.1 Performance on WordNet

We examine whether our algorithm can replicate the common use of WordNet (§2.1), by manually constructing 4 whitelists based on the literature

[7] Since our methods do not use lexical features, we did not use lexical splits as in (Levy et al., 2015).

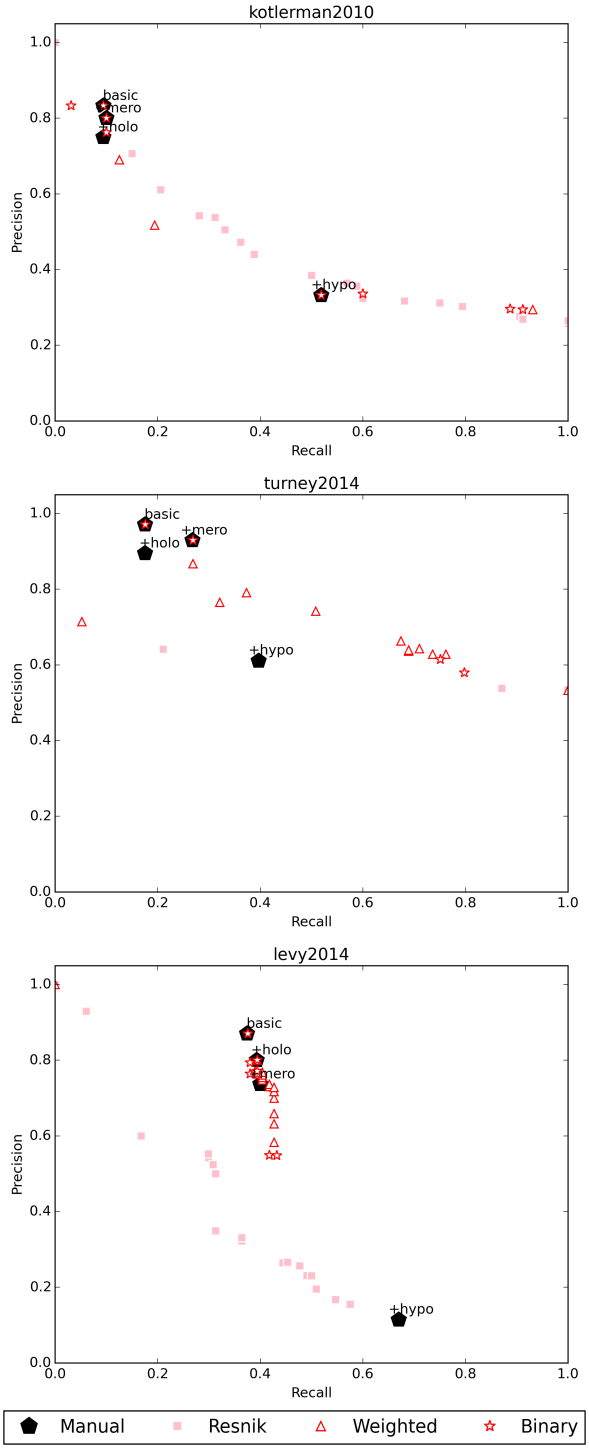

Figure 3: Recall-precision curve of each dataset with WordNet as the only resource. Each point in the graph stands for the performance on a certain value of β. Notice that in some of the graphs, different β values yield the same performance, causing less points to be displayed.

(see Table 4), and evaluating their performance using the classification methods in §4.2. In addition, we compare our method to Resnik's (1995) WordNet similarity, which scores each pair of terms based on their lowest common hypernym. This score was used as a single feature in F_β-optimized logistic regression to create a classifier.

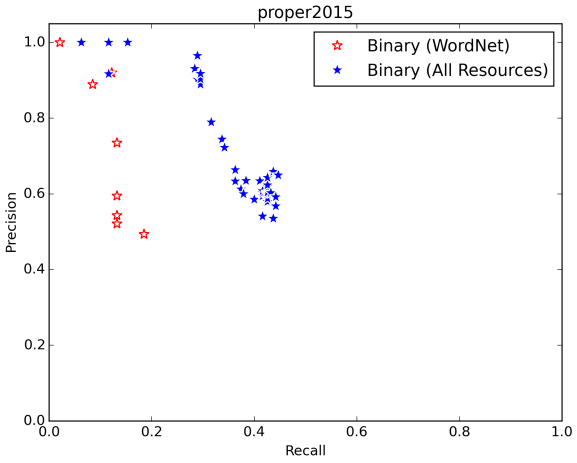

Figure 4: Recall-precision curve for `proper2015`.

Name	Edge Types
basic	{synonym, hypernym, instance_hypernym}
+holo	basic ∪ {holonym}
+mero	basic ∪ {meronym}
+hypo	basic ∪ {hyponym}

Table 4: The manual whitelists commonly used in WordNet.

Figure 3 compares our algorithm to WordNet's baselines, showing that our binary model always replicates the best-performing manually-constructed whitelists, for certain values of β^2. Synonyms and hypernyms are often selected, and additional edges are added to match the semantic flavor of each particular dataset. In `turney2014`, for example, where meronyms are common, our binary model learns that they are indicative by including meronymy in its whitelist. In `levy2014`, however, where meronyms are less indicative, the model does not select them.

We also observe that, in most cases, our algorithm outperforms Resnik's similarity. In addition, the weighted model does not perform as well as the binary model, as discussed in §4.2. We therefore focus our presentation on the binary model.

6.2 Lexical Inference over Proper-Names

We evaluated our model on the new proper-name dataset `proper2015` described in §5.2. This time, we incorporated all the resources described in §2.2 (including WordNet) into our framework, and compared the performance to that of using WordNet alone. Indeed, our algorithm is able to exploit the information in the additional resources and greatly increase performance, particularly recall, on this dataset (Figure 4).[8]

[8] We also evaluated our algorithm on the common-nouns datasets with all resources, but apparently, adding resources did not significantly improve performance.

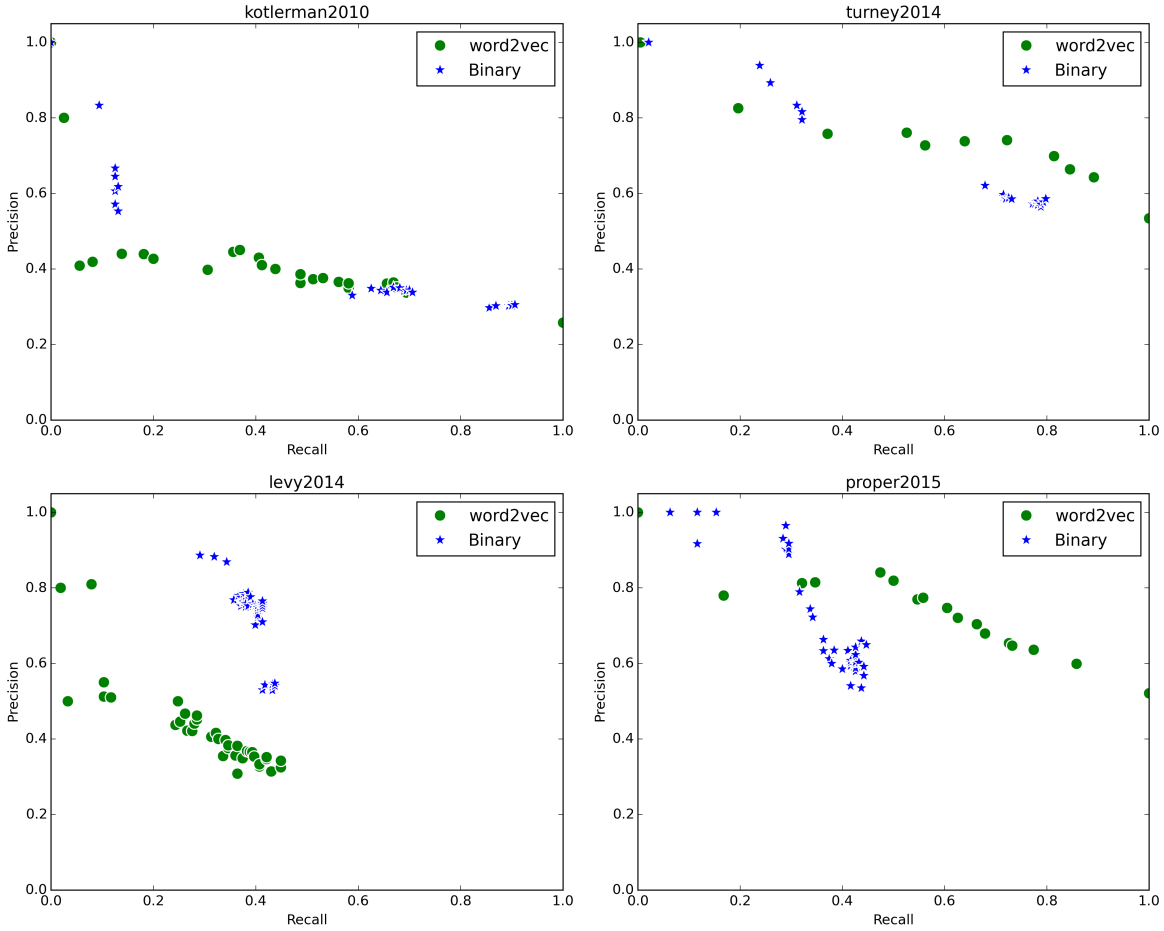

Figure 5: Recall-precision curve of each dataset using: (1) Supervised word2vec (2) Our binary model.

The binary model yields 97% precision at 29% recall, at the top of the "precision cliff". The whitelist learnt at this point contains 44 edge types, mainly from Wikidata and Yago. Even though the *is_a* relation implicitly defined in `proper2015` is described using many different edge types, our binary model still manages to learn which of the over 2,500 edge types are indicative. Table 5 shows some of the learnt edge types (see the supplementary material for the complete list).

The performance boost in `proper2015` demonstrates that community-built resources have much added value when considering proper-names. As expected, many proper-names do not appear in WordNet (*Doctor Who*). That said, even when both terms appear in WordNet, they often lack important properties covered by other resources (*Louisa May Alcott* is a *woman*).

6.3 Comparison to Corpus-based Methods

Lexical inference has been thoroughly explored in distributional semantics, with recent supervised methods (Baroni et al., 2012; Turney and Mohammad, 2015) showing promising results. While these methods leverage huge corpora to increase coverage, they often introduce noise that affects their precision. Structured resources, on the other hand, are precision-oriented. We therefore expect our approach to complement distributional methods in high-precision scenarios.

Edge Type	Example
occupation	Daniel Radcliffe → actor
sex_or_gender	Louisa May Alcott → woman
instance_of	Doctor Who → series
acted_in	Michael Keaton → Beetlejuice
genre	Touch → drama
position_played_on_team	Jason Collins → center

Table 5: An excerpt of the whitelist learnt for `proper2015` by the binary model with accompanying true-positives that do not have an indicative path in WordNet.

To represent term-pairs with distributional features, we downloaded the pre-trained `word2vec` embeddings.[9] These vectors were trained over a huge corpus (100 billion words) using a state-of-the-art embedding algorithm (Mikolov et al., 2013). Since each vector represents a single term (either x or y), we used 3 state-of-the-art meth-

[9] `http://code.google.com/p/word2vec/`

ods to construct a feature vector for each term-pair: concatenation $\vec{x} \oplus \vec{y}$ (Baroni et al., 2012), difference $\vec{y} - \vec{x}$ (Roller et al., 2014; Fu et al., 2014; Weeds et al., 2014), and similarity $\vec{x} \cdot \vec{y}$. We then used F_β-optimized logistic regression to train a classifier. Figure 5 compares our methods to concatenation, which was the best-performing corpus-based method.[10]

In `turney2014` and `proper2015`, the embeddings retain over 80% precision while boasting higher recall than our method's. In `turney2014`, it is often a result of the more associative relations prominent in the dataset ($football \rightarrow playbook$), which seldom are expressed in structured resources. In `proper2015`, the difference in recall seems to be from missing terminology ($Twitter \rightarrow social\ network$). However, the corpus-based method's precision does not exceed the low 80's, while our binary algorithm yields 93% @ 27% precision-at-recall on `turney2014` and 97% @ 29% on `proper2015`.

In `levy2014`, there is an overwhelming advantage to our resource-based method over the corpus-based method. This dataset contains healthcare terms and might require a domain-specific corpus to train the embeddings. Having said that, many of its examples are of an ontological nature (drug x treats disease y), which may be more suited to our resource-based approach, regardless of domain.

7 Error Analysis

Since resource-based methods are precision-oriented, we analyzed our binary model by selecting the setting with the highest attainable recall that maintains high precision. This point is often at the top of a "precision cliff" in Figures 3 and 4. These settings are presented in Table 6.

The high-precision settings we chose resulted in few false positives, most of which are caused by annotation errors or resource errors. Naturally, regions of higher recall and lower precision will yield more false positives and less false negatives. We thus focus the rest of our discussion on false negatives (Table 7).

While structured resources cover most terms, the majority of false negatives stem from the lack of indicative paths between them. Many important relations are not explicitly covered by the resources, such as noun-quality ($saint \rightarrow holiness$), which are abundant in `turney2014`, or causality ($germ \rightarrow infection$), which appear in `levy2014`. These examples are occasionally captured by other (more specific) relations, and tend to be domain-specific.

In `kotlerman2010`, we found that many false negatives are caused by annotation errors in this dataset. Pairs are often annotated as positive based on associative similarity (e.g. $transport \rightarrow environment$, $financing \rightarrow management$), making it difficult to even manually construct a coherent whitelist for this dataset. This may explain the poor performance of our method and other baselines on this dataset.

Dataset	β	Whitelist	Prec.	Rec.
`kotlerman2010`	0.05	basic	83%	9%
`turney2014`	0.05	+mero	93%	27%
`levy2014`	10^{-5}	basic	87%	37%
`proper2015`	0.3	44 edge types from all resources (see supplementary material)	97%	29%

Table 6: The error analysis setting of each dataset.

Error Type	kotlerman 2010	levy 2014	turney 2014	proper 2015
Not Covered	2%	12%	4%	13%
No Indicative Paths	35%	48%	73%	75%
Whitelist Error	6%	3%	5%	8%
Resource Error	15%	11%	7%	0%
Annotation Error	40%	23%	7%	1%
Other	2%	3%	4%	3%

Table 7: Analysis of false negatives in each dataset. We observed the following errors: (1) One of the terms is out-of-vocabulary (2) All paths are not indicative (3) An indicative path exists, but discarded by the whitelist (4) The resource describes an inaccurate relation between the terms (5) The term-pair was incorrectly annotated as positive.

8 Conclusion and Future Work

In this paper, we presented a supervised framework for utilizing structured resources to recognize lexical inference. We demonstrated that our framework replicates the common practice of WordNet and can increase the coverage of proper-names by exploiting larger structured resources. Compared to the prior practice of manually identifying useful relations in structured resources, our contribution offers a principled learning approach for automating and optimizing this common need.

While our method enjoys high-precision, its recall is limited by the resources' coverage. In future work, combining our method with high-recall

[10] Note that the corpus-based method benefits from lexical memorization (Levy et al., 2015), overfitting for the lexical terms in the training set, while our resource-based method does not. This means that Figure 5 paints a relatively optimistic picture of the embeddings' actual performance.

corpus-based methods may have synergistic results. Another direction for increasing recall is to use cross-resource mappings to allow cross-resource paths (connected at the concept-level).

Finally, our method can be extended to become context-sensitive, that is, deciding whether the lexical inference holds in a given context. This may be done by applying a resource-based WSD approach similar to (Brody et al., 2006; Agirre et al., 2014), detecting the concept node that matches the term's sense in the given context.

Acknowledgments

This work was supported by an Intel ICRI-CI grant, the Google Research Award Program and the German Research Foundation via the German-Israeli Project Cooperation (DIP, grant DA 1600/1-1).

References

Eneko Agirre, Oier Lopez de Lacalle, and Aitor Soroa. 2014. Random walks for knowledge-based word sense disambiguation. *Computational Linguistics*, 40(1):57–84.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. *Dbpedia: A nucleus for a web of open data*. Springer.

Marco Baroni and Alessandro Lenci. 2011. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10, Edinburgh, UK, July. Association for Computational Linguistics.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32, Avignon, France, April. Association for Computational Linguistics.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, October. Association for Computational Linguistics.

Samuel Brody, Roberto Navigli, and Mirella Lapata. 2006. Ensemble methods for unsupervised wsd. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 97–104. Association for Computational Linguistics.

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.

Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1199–1209, Baltimore, Maryland, June. Association for Computational Linguistics.

Sanda Harabagiu and Dan Moldovan. 1998. Knowledge processing on an extended wordNet. *WordNet: An electronic lexical database*, 305:381–405.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*, pages 529–545, Nantes, France.

Martin Jansche. 2005. Maximum expected f-measure training of logistic regression models. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 692–699, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(04):359–389.

Omer Levy, Ido Dagan, and Jacob Goldberger. 2014. Focused entailment graphs for open ie propositions. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 87–97, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. Do supervised distributional methods really learn lexical inference relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976, Denver, Colorado, May–June. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet::similarity - measuring the relatedness of concepts. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Demonstration Papers*, pages 38–41, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Altaf Rahman and Vincent Ng. 2011. Coreference resolution with world knowledge. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 814–824, Portland, Oregon, USA, June. Association for Computational Linguistics.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, IJCAI'95, pages 448–453. Morgan Kaufmann Publishers Inc.

Stephen Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet selective: Supervised distributional hypernymy detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1025–1036, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.

D E Rumelhart, G E Hinton, and R J Williams. 1986. Learning representations by back-propagating errors. *Nature*, pages 533–536.

Stuart Russell and Peter Norvig. 2009. *Artificial Intelligence: A Modern Approach*. Prentice Hall.

Eyal Shnarch, Libby Barak, and Ido Dagan. 2009. Extracting lexical reference rules from wikipedia. 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*, pages 450–458, Suntec, Singapore, August. Association for Computational Linguistics.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

Peter D Turney and Saif M Mohammad. 2015. Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering*, 21(03):437–476.

Peter D Turney. 2006. Similarity of semantic relations. *Computational Linguistics*, 32(3):379–416.

Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. 2012. Template-based question answering over rdf data. In *Proceedings of the 21st international conference on World Wide Web*, pages 639–648. ACM.

Denny Vrandečić. 2012. Wikidata: A new platform for collaborative data collection. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 1063–1064. ACM.

Julie Weeds and David Weir. 2003. A general framework for distributional similarity. In Michael Collins and Mark Steedman, editors, *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 81–88.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2249–2259, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.

Fei Wu and Daniel S. Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127, Uppsala, Sweden, July. Association for Computational Linguistics.

Jihoon Yang and Vasant Honavar. 1998. Feature subset selection using a genetic algorithm. In *Feature extraction, construction and selection*, pages 117–136. Springer.

Appendix A Efficient Path-Finding

We split the search to two phases: we first find all nodes along the shortest paths between x and y, and then reconstruct the actual paths. The first phase ignores edge types, inducing a simpler resource graph, which we represent as a sparse adjacency matrix and manipulate efficiently with matrix operations (Algorithm 1). Once the search space is limited to relevant nodes only, the second phase becomes trivial.

Algorithm 1 Find Relevant Nodes

1: **function** NODESINPATH($\vec{n}_x, \vec{n}_y, len$)
2: **if** $len == 1$ **then**
3: **return** $\vec{n}_x \cup \vec{n}_y$
4: **for** $0 < k \leq len$ **do**
5: **if** k is odd **then**
6: $\vec{n}_x = \vec{n}_x \cdot A$
7: **else**
8: $\vec{n}_y = \vec{n}_y \cdot A^T$
9: **if** $\vec{n}_x \cdot \vec{n}_y > 0$ **then**
10: $\vec{n}_{xy} = \vec{n}_x \cap \vec{n}_y$
11: $\vec{n}_{forward} = nodesInPath(\vec{n}_x, \vec{n}_{xy}, \lceil \frac{k}{2} \rceil)$
12: $\vec{n}_{backward} = nodesInPath(\vec{n}_{xy}, \vec{n}_y, \lfloor \frac{k}{2} \rfloor)$
13: **return** $\vec{n}_{forward} \cup \vec{n}_{backward}$
14: **return** $\vec{0}$

The algorithm finds all nodes in the paths between x and y subject to the maximum length (len). A is the resource adjacency matrix and $\vec{n}_x, \vec{n}_y$ are one-hot vectors of x, y.
At each iteration, we either make a forward (line 6) or a backward (8) step. If the forward and backward search meet (9), we recursively call the algorithm for each side (11-12), and merge their results (13). The stop conditions are $len = 0$, returning an empty set when no path was found, and $len = 1$, merging both sides when they are connected by single edges.