When Hearst Is not Enough: Improving Hypernymy Detection from Corpus with Distributional Models

Changlong Yu1∗ Jialong Han2 Peifeng Wang3 Yangqiu Song1
Hongming Zhang1 Wilfred Ng1 Shuming Shi4
1HKUST 2Amazon 3University of Southern California 4Tencent AI Lab
{cyuaq, yqsong, hzhangal, wilfred}@cse.ust.hk
jialonghan@gmail.com, peifengw@usc.edu, shumingshi@tencent.com

Abstract

We address hypernymy detection, i.e., whether an is-a relationship exists between words \((x, y)\), with the help of large textual corpora. Most conventional approaches to this task have been categorized to be either pattern-based or distributional. Recent studies suggest that pattern-based ones are superior, if large-scale Hearst pairs are extracted and fed, with the sparsity of unseen \((x, y)\) pairs relieved. However, they become invalid in some specific sparsity cases, where \(x\) or \(y\) is not involved in any pattern. For the first time, this paper quantifies the non-negligible existence of those specific cases. We also demonstrate that distributional methods are ideal to make up for pattern-based ones in such cases. We devise a complementary framework, under which a pattern-based and a distributional model collaborate seamlessly in cases which they each prefer. On several benchmark datasets, our framework achieves competitive improvements and the case study shows its better interpretability.

1 Introduction

A taxonomy is a semantic hierarchy of words or concepts organized w.r.t. their hypernymy (a.k.a. is-a) relationships. Being a well-structured resource of lexical knowledge, taxonomies are vital to various tasks such as question answering (Gupta et al., 2018), textual entailment (Dagan et al., 2013; Bowman et al., 2015; Yu et al., 2020b), and text generation (Biran and McKeown, 2013). When automatically building taxonomies from scratch or populating manually crafted ones, the hypernymy detection task plays a central role. For a pair of queried words \((x_q, y_q)\), hypernymy detection requires inferring the existence of a hyponym-hypernym relationship between \(x_q\) and \(y_q\). Due to

∗ Work done when C. Yu, J. Han and P. Wang were with Tencent AI Lab.
tional ones follow, or are inspired by, the Distributional Inclusion Hypothesis (DIH; Geffet and Dagan 2005), i.e., the set of the hyponym’s contexts should be roughly contained by the hypernym’s. Although applicable to any word in a corpus, they are suggested to be inferior to pattern-based ones fed with sufficient extracted pairs (Roller et al., 2018; Le et al., 2019).

Since pattern-based methods have unresolved sparsity issues, while distributional ones are more broadly applicable but globally inferior, neither of them can dominate the other in every aspect. In this light, we are interested in two questions:

- Is the Type-II sparsity severe in practice?
- If so, how to complement pattern-based approaches with distributional ones where the former is invalid?

To answer the first question, we conduct analyses involving estimations on real-world corpora as well as statistics of common hypernymy detection datasets. Results from both resources indicate that the likelihood of encountering the Type-II sparsity in practice could even reach up to more than 50%, which is thus non-negligible.

For the second question, we present ComHyper, a complementary framework (Sec. 4.1) which takes advantage of both pattern-based models’ superior performance on Type-I cases and the broad coverage of distributional models on Type-II ones. Specifically, to deal with Type-II sparsity, instead of directly using unsupervised distributional models, ComHyper uses a training stage (Sec. 4.3) to sample from output space of a pattern-based model to train another supervised distribution model implemented by different context encoders (Sec. 4.2). In the inference stage, ComHyper uses the two models to separately handle the type of sparsity they are good at, as illustrated in Figure 1. In this manner, ComHyper relies on the partial use of pattern-based models on Type-I sparsity to secure performance no lower than distributional ones, and further attempts to lift the performance by fixing the former’s blind spots (Type-II sparsity) with the latter. On several benchmarks and evaluation settings, the distributional model in ComHyper proves effective on its targeted cases, making our complementary approach outperform a competitive class of pattern-based baselines (Roller et al., 2018). Further analysis also suggests that ComHyper is robust when facing different mixtures of Type-I and Type-II sparsity.

Our contributions are summarized as: 1) We confirm that a specific type of sparsity issue of current pattern-based approaches is non-negligible. 2) We propose a framework of complementing pattern-based approaches with distributional models where the former is invalid. 3) We systematically conduct comparisons on several common datasets, validating the superiority of our framework.

## 2 Related Work

### Pattern-Based Approaches.

Taxonomies from experts (e.g., WordNet (Miller, 1995)) have proved effective in various reasoning applications (Song et al., 2011; Zhang et al., 2020). Meanwhile, Hearst patterns (Hearst, 1992) make large corpora a good resource of explicit is-a pairs, resulting in automatically built hypernymy knowledge bases (Wu et al., 2012; Seittner et al., 2016) of large scales. The coverage of both words and hypernymy pairs in those resources are far from complete.

To infer unknown hypernymies between known words, e.g., implicit is-a pairs in transitive closures, pattern-based models are proposed. Roller et al. (2018) and Le et al. (2019) show that, on a broad range of benchmarks, simple matrix decomposition or embeddings on pattern-based word co-occurrence statistics provide robust performance. On Probase (Wu et al., 2012) - a Hearst-pattern-based taxonomy, Yu et al. (2015) use embeddings to address the same sparsity problem. Some methods (Vendrov et al., 2016; Athiwaratkun and Wilson, 2018; Nickel and Kiela, 2017, 2018; Ganea et al., 2018) embed WordNet in low-dimensional space. Depending on vectors of words learnt from known is-a pairs, the above pattern-based methods cannot induce more hypernymy pairs whose words do not appear in any pattern.

### Distributional Approaches.

Distributional models are inspired by DIH (Geffet and Dagan, 2005). They work on only word contexts rather than extracted pairs, thus are applicable to any word in a corpus. Early unsupervised models typically propose asymmetric similarity metrics over manual word feature vectors for entailment (Weeds et al., 2004; Clarke, 2009; Santus et al., 2014). In Chang et al. (2018) and Nguyen et al. (2017), the authors inject DIH into unsupervised embedding models to yield latent feature vectors with hypernymy information. Those feature vectors, manual or latent, may serve in unsupervised asymmetric metrics or to train supervised hypernymy classifiers. Shwartz
et al. (2017) explore combinations of manual features and (un)supervised predictors, and suggest that unsupervised metrics are more robust w.r.t. the distribution change of training instances. Projection learning (Fu et al., 2014; Ustalov et al., 2017; Wang and He, 2020) has been used for supervised hypernymy detection.

**Other Improved Methods.** Due to weak generalization ability of Hearst patterns, Anh et al. (2016) and Shwartz et al. (2016) relieve the constraints from strict Hearst patterns to co-occurring contexts or lexico-syntactic paths between two words. They encode the co-occurring contexts or paths using word vectors to train hypernymy embeddings or classifiers. Although leading to better recall than Hearst patterns (Washio and Kato, 2018), they limit the trained embeddings or models from generalizing to every word in a corpus. Nevertheless, they have no ability to cope with the Type-II sparsity, which is the main focus of our work.

Another line of retrofitting methods (Vulić et al., 2018; Vulić and Mrkšić, 2018), i.e., adjusting distributional vectors to satisfy external linguistic constraints, has been applied to hypernymy detection. However, they strictly require more additional resources e.g., synonym and antonym to achieve better performance (Kamath et al., 2019). To the best of our knowledge, we are the first to propose complementing the two lines of approaches to cover every word in a simple yet efficient way, with extensive analysis of the framework’s potential and evaluation of performances.

### 3 Preliminaries

We formally define the aforementioned two types of sparsity, and provide some statistical insights about their impacts on pattern-based methods.

#### 3.1 Notations and Definitions

Let $V$ be the vocabulary of a corpus $C$. By applying Hearst patterns on $C$, a set of extracted pairs $P \subseteq V \times V$, i.e., is-a relationships $\{(x, y)\}$ $(x, y \in V)$, is obtained. As in Section 2, pattern-based approaches usually use $P$ to perform matrix factorization or embedding learning. Due to their nature, only words “seen” in $P$, or $V_P = \{x | (x, y) \in P \lor (y, x) \in P\}$, will have respective columns/rows or embeddings. We refer to them by in-pattern (or IP for short) words. We refer to words without columns/rows or embeddings, i.e., $V \setminus V_P$, by out-of-pattern (or OOP) words.

Suppose a pair of words $q = (x_q, y_q)$ is queried for potential hypernymy. We say $q$ is an IP pair if both $x_p$ and $y_p$ are IP words, or an OOP pair if either of them is OOP. Due to the need of explicit columns/rows or embeddings for both $x_q$ and $y_q$, pattern-based approaches may only make inferences on IP pairs, but are infeasible on OOP ones.

#### 3.2 Observations and Motivation

Given the infeasibility of pattern-based methods on OOP pairs, we are interested in what extent pattern-based methods are limited, i.e., the rough likelihood of encountering OOP pairs in practice. At first sight, Hearst patterns may have very sparse occurrences in a corpus. Nevertheless, words with higher frequencies tend to be covered by Hearst patterns and be IP words. Therefore, the possibility of encountering OOP pairs is not obvious to assess.

To shed light upon the OOP issue of pattern-based methods, we conduct an analysis on the corpora and extracted pairs in Roller et al. (2018). Considering that nouns tend to be queried more for potential hypernymy than, say, verbs, we only focus on nouns. In Figure 2, we show the corpus frequency of all nouns and in-pattern nouns, and draw the following observations.

1) $V_P$ covers well the most frequent nouns in $V$. For the top-10⁴ frequent nouns, the two lines of dots overlap well, indicating that common nouns are very likely to be involved in Hearst patterns.

2) Due to the limited size of $V_P$, it is unable to cover the tail of $V$. With the frequency rank below $10^4$, the two lines begin to separate. Comparing their intersections with the x-axis, it is understandable that a limited number of IP nouns cannot cover both frequent and tail nouns in a vocabulary, whose size is several orders of magnitudes larger.

3) The likelihood of a noun being OOP is non-negligible. The two lines enclose a triangular region, corresponding to the likelihood of a randomly drawn noun being OOP. According to our statistics,
Table 1: Statistics of OOP pairs w.r.t. extracted pairs \( P \).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>OOP (Hyper/All)</th>
<th>Total</th>
<th>OOP Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLESS</td>
<td>44 / 1,829</td>
<td>14,542</td>
<td>12.58%</td>
</tr>
<tr>
<td>EVAL</td>
<td>694 / 3,903</td>
<td>13,450</td>
<td>29.02%</td>
</tr>
<tr>
<td>LEDS</td>
<td>105 / 209</td>
<td>2,770</td>
<td>7.55%</td>
</tr>
<tr>
<td>SHWARTZ</td>
<td>7,209 / 35,266</td>
<td>52,577</td>
<td>67.07%</td>
</tr>
<tr>
<td>W(B)LESS</td>
<td>0 / 46</td>
<td>1,668</td>
<td>2.76%</td>
</tr>
<tr>
<td>HYPERLEX</td>
<td>n/a / 107</td>
<td>2,163</td>
<td>4.95%</td>
</tr>
</tbody>
</table>

Note that neither the datasets nor \( P \) are created in favor of the other. These actual rates may be above or below the estimated interval of 19.9%-35.8%, but are all at considerable levels. Considering the above analyses, we confirm that OOP pairs are non-negligible in practice and give a positive answer to the first question in Section 1.

Motivation of the Study. OOP pairs are problematic for pattern-based methods. Despite their non-negligible existence, former pattern-based methods (Roller et al., 2018; Le et al., 2019) boldly classify them as non-hypernymy in prediction. However, distributional methods are applicable as long as the two queried words have contexts. Thus, they are ideal to complement pattern-based methods on the non-negligible minority of OOP pairs.

4 Our Approach

4.1 Framework

Our framework is illustrated in Figure 1. It consists of a pattern-based model and a distributional model cooperating on the data resource to answer an arbitrarily queried pair of words \( q \in V \times V \).

Data Resource. To train a pattern-based model using prior solutions, our data resource includes extracted pairs \( P \) from some text corpus \( C \). Unlike pattern-based approaches that depend solely on \( P \), our data resource also involves the corpus \( C \) for the sake of the distributional model.

Pattern-Based Model. The pattern-based model works on the extracted pairs \( P \) to serve in two roles. On the one hand, it is responsible for generalizing from statistics on \( P \) to score any in-pattern pair \( q \in V_P \times V_P \) to reflect the plausibility of a hypernymy relationship. To this end, it is sufficient to adopt matrix-factorization-based (Roller et al., 2018) or embedding models (Le et al., 2019). On the other hand, the pattern-based model also provides supervision signals via a sampler for training the distributional model. We will specify this role later. Formally, we denote the pattern-based model by \( f : V_P \times V_P \to \mathbb{R} \).

Distributional Model. Different from the pattern-based model defined on IP pairs \( V_P \times V_P \), the distributional model has a form of \( g : V \times V \to \mathbb{R} \), i.e., it should be capable of predicting on any word pair in \( V \times V \). This invalidates the model’s dependency on extracted pairs involving \( x_q \) or \( y_q \). The separate contexts of \( x_q \) and \( y_q \) in corpus \( C \) turn out to serve as the basis and input of the distributional model, respectively. Given the superior performance of pattern-based models on IP pairs (Roller et al., 2018), the distributional model \( g \) is only responsible to answer OOP pairs.

Various choices exist to implement the distributional model. We may apply unsupervised metrics (Weeds et al., 2004; Clarke, 2009; Santus et al., 2014) on manual features extracted from contexts of \( x_q \) and \( y_q \), which are robust to the distribution change of training data (Shwartz et al., 2017). However, the scores of those metrics are not necessarily in the same scale with those output by the pattern-based model \( f \) for IP pairs. Such inconsistency will harm downstream systems which involve the scores for ranking or calculation.

Given sufficient supervision signals from \( f \) and the inherent noise of natural language, we implement the distributional model \( g \) by a supervised neural-network-based approach. Specifically, the network encodes the contexts of \( x \) and \( y \) in \( C \), i.e., \( C(x) \) and \( C(y) \), to be \( x_H \) and \( y_H \), respectively, and makes predictions by a dot product, i.e.,

\[
g(x ; y) = \langle x_H, y_H \rangle.
\]

Note that hypernymy is essentially asymmetric, so we distinguish \( x_H \) and \( y_H \) by the subscripts to reflect the asymmetry. In practice, we adopt networks with separate parameters for \( C(x) \) and \( C(y) \), which is detailed in the next section.
4.2 Encoding Queried Words

To implement the distributional model, we encode \( C(x) \) and \( C(y) \) into hypernymy-specific representations \( x_h \) and \( y_H \), respectively. There are various off-the-shelf models to encode sentential contexts. We take the following four approaches.

**Transformed Word Vector.** Instead of working directly on the original contexts \( C(x) \) and \( C(y) \), this approach takes as input the pre-trained word vectors (Mikolov et al., 2013; Pennington et al., 2014) \( x \) and \( y \) of \( x \) and \( y \), and apply two Multi-Layer Perceptrons (MLPs), respectively:

\[
x_h = \text{MLP}_h(x), \quad y_H = \text{MLP}_H(y).
\]

The intuition is that word vectors roughly depend on the contexts and encode the distributional semantics. To make the MLPs generalize to \( V \) rather than \( V_p \), the word vectors are fixed during training. Inspired by the post specialization in Vulić et al. (2018), it also takes a similar approach to generalize task-specific word vector transformations to unseen words, though their evaluation task is not hypernymy detection.

**NBoW with MEAN-Pooling.** Given words \( \{c_j\}_{j=1}^n \) in a context \( c \in C(x) \), the Neural Bag-of-Words (NBoW for short) encoder looks up and averages their pre-trained vectors \( c_j \) as \( c \), transforms \( c \) through a MLP, and averages the resulted vectors through a MEAN-pooling layer as \( x_h \):

\[
x_h = \frac{1}{|C(x)|} \sum_{c \in C(x)} \text{MLP}_h(c), \quad c = \frac{1}{n} \sum_{j=1}^n c_j.
\]

To obtain \( y_H \), a similar network is applied, though the two MLPs do not share parameters to reflect the asymmetry of hypernymy. We fix the embeddings of context word vectors during training because satisfactory performance is observed. Due to its simplicity, NBoW is efficient to train. However, it ignores the order of context words and may not well reserve semantics.

**CONTEXT2VEC with MEAN-Pooling.** To study the impacts of positional information within the context, we also attempt to substitute the NBoW with the CONTEXT2VEC encoder (Melamud et al., 2016). In CONTEXT2VEC, two LSTMs are used to encode the left and right contexts \( \overrightarrow{c} \) and \( \overleftarrow{c} \) of an occurrence of \( x \), respectively. The two output vectors are concatenated as the final context representation \( c \) for the same transformation and averaging as for NBoW. Formally,

\[
c = \left[ \text{LSTM}(\overrightarrow{c}); \text{LSTM}(\overleftarrow{c}) \right].
\]

Note that the encoder for \( y \) still has separate parameters from those of \( x \).

**Hierarchical Attention Networks.** NBoW and CONTEXT2VEC with MEAN-Pooling both aggregate every context word’s information into \( x_h \) and \( y_H \). Given several long contexts and the fixed output dimension, it is vital for encoders to capture the most useful information. Inspired by Yang et al. (2016), we incorporate attention on different words and contexts. We use a feed-forward network to estimate the importance, and combine the information, of each context word to obtain \( c \):

\[
\alpha_j = \text{softmax}\left( w_a^T \tanh(W_a c_j) \right), \quad c = \sum_{j=1}^n \alpha_j c_j.
\]

Then, another similar network is applied to all \( c^{(i)} \in C(x) \) to obtain the representation of \( x_h \):

\[
\beta_i = \text{softmax}\left( w_b^T \tanh(W_b c^{(i)}) \right), \quad x_h = \sum_{i=1}^{|C(x)|} \beta_i c^{(i)}.
\]

For word \( y \), the encoder is similar but still has separate parameters from those of \( x \).

4.3 Training the Distributional Model

We train the distributional model \( g \)’s parameters \( \Phi \) with supervision signals from the pattern-based model \( f \). To make output scores of \( f \) and \( g \) comparable, we adopt the square error between the two scores as the loss on a pair \((x, y)\), i.e.,

\[
l(x, y; \Phi) = \left( g(x, y; \Phi) - f(x, y) \right)^2.
\]

Compared with the potentially large size of the output space, a set of random samples from it suffices to train the parameters \( \Phi \). For each IP word \( x \in V_p \), we uniformly sample \( k \) entries from \( \Delta_x \), the column and row involving \( x \) in the output space \( V_p \times V_p \):

\[
\Delta_x = \{(x, y) \mid y \in V_p\} \cup \{(y, x) \mid y \in V_p\}.
\]

The sample for \( x \) is done on \( P_x \), a uniform distribution over \( \Delta_x \). Finally, our objective is

\[
\min_{x \in V_p} \mathcal{L}(x; \Phi),
\]

where \( \mathcal{L}(x; \Phi) \) is the expected loss related to \( x \):

\[
\mathcal{L}(x; \Phi) = \sum_{i=1}^k \mathbb{E}_{(x^{(i)}, y^{(i)}) \sim P_x} l(x^{(i)}, y^{(i)}; \Phi).
\]
5 Experimental Setup
We adopt the widely-used comprehensive evaluation framework\(^1\) provided by Roller et al. (2018); Le et al. (2019). To make experimental results comparable, we align the settings as much as possible.

5.1 Corpora and Evaluation
Corpora. We used the 431k is-a pairs (243k unique) released by Roller et al. (2018). We substitute the Gigaword corpus they used by uKWac (Ferraresi, 2007) because the former is not complimentary. This decision does not affect reproducing pattern-based approaches in Roller et al. (2018).

Evaluation Tasks. The three sub-tasks include 1) ranked hypernym detection: given \((x_q, y_q)\) decide whether \(y_q\) is a hypernym of \(x_q\). Five datasets \(i.e., \) BLESS (Baroni and Lenci, 2011), EVAL (Santus et al., 2015), LEADS (Baroni et al., 2012), SHWARTZ (Schwartz et al., 2016) and WBLESS (Weeds et al., 2014) are used. The positive predictions should be ranked higher over negative ones and Average Precision (AP) is used for evaluation. 2) hypernymy direction classification: determine which word in a pair has a broader meaning. Besides BLESS and WBLESS, we also use BiBLESS (Kiela et al., 2015) and Accuracy (Acc.) is reported for binary classification. 3) graded entailment: predict scalar scores on HYPERLEX (Vulić et al., 2017). Spearman’s correlation \(\rho\) between the labels and predicted scores is reported.

The statistics of datasets are shown in Table 1. The three tasks require algorithms to output scores unsupervisedly, which indicate the strength of hypernymy relationships. Note no external training data is available in the evaluation. Only extracted Hearst pattern pairs may be used for supervision.

5.2 Compared Methods
Pattern-Based Approaches. We reproduce four pattern-based methods \(i.e., \) Count, PPMI, SVD-Count, and SVD-PPMI. As in Roller et al. (2018), SVD-PPMI is generally the most competitive.

Distributional Approaches. We compare with unsupervised distributional baselines in Roller et al. (2018), \(i.e., \) Cosine, Weeds Precision (WP), invCL, and SLQS. For supervised distributional baseline, we adopt the strongest model SDSN in Rei et al. (2018) and take the probability scores of binary classifier as hypernymy predictions. All the 431k extracted pairs serve as true hypernymy pairs and false ones are generated by replacing one of the terms in true pairs with a random term.

Complementary Approaches. We adopt SVD-PPMI as the pattern-based model in our framework. We pre-train 300-dimensional word embeddings with Skip-Gram (Mikolov et al., 2013) on our corpus for the use of the distributional model. Specifically, we compare transformed word vector (W2V), NBoW/CONTEXT2VEC with Mean-Pooling (NBoW/C2V), and Hierarchical Attention Networks (HAN)\(^2\). The output dimension of our four encoders is set to 300. The batch size is set to 128 and learning rate to \(10^{-3}\). We tuned the sampling size \(k\) in \(\{1, 3, 5, 10, 100, 200, 400, 800\}\) on the validation set. We did not tune other hyperparameters since the default settings work well. Our code is available at https://github.com/ccclyu/ComHyper.

6 Experimental Results
We aim to answer: 1) Are our distributional models supervised well by the pattern-based model? 2) Do they improve our complementary methods over the pattern-based ones? 3) Are complementary methods robust \(w.r.t. \) fewer extracted pairs?

<table>
<thead>
<tr>
<th></th>
<th>Detection (AP)</th>
<th>Dir.(Acc.)</th>
<th>Graded((\rho))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLESS</td>
<td>EVAL</td>
<td>LEDS</td>
</tr>
<tr>
<td>Cosine</td>
<td>.106</td>
<td>.172</td>
<td>.736</td>
</tr>
<tr>
<td>WP</td>
<td>.100</td>
<td>.251</td>
<td>.880</td>
</tr>
<tr>
<td>invCL</td>
<td>.096</td>
<td>.211</td>
<td>.887</td>
</tr>
<tr>
<td>SLQS</td>
<td>.020</td>
<td>.166</td>
<td>.423</td>
</tr>
<tr>
<td>W2V</td>
<td>.292</td>
<td>.255</td>
<td>.712</td>
</tr>
<tr>
<td>NBoW</td>
<td>.124</td>
<td>.258</td>
<td>.617</td>
</tr>
<tr>
<td>C2V</td>
<td>.027</td>
<td>.258</td>
<td>.659</td>
</tr>
<tr>
<td>HAN</td>
<td>.346</td>
<td>.250</td>
<td>.602</td>
</tr>
</tbody>
</table>

Table 2: Experimental results on OOP pairs.

\(^1\)https://github.com/facebookresearch/hypernymysuite

\(^2\)Heavy contextualized encoders based on the pretrain-finetune framework did not yield considerable improvement and we focus on efficient traditional encoders which already outperform the baselines. Though we include the BERT encoders in our released code, we suggest to make tradeoffs when choosing encoders as discussed in Xia et al. (2020).
Table 3: Experimental results on all queried pairs. Best ones are marked bold while second-best ones underlined.
6.3 Impacts of Reduced Pairs

To analyze our complementary framework’s robustness w.r.t. sparser extracted pairs $\mathcal{P}$, we randomly sample $\{95\%,75\%,55\%,35\%,15\%\}$ of all $243k$ is-a pairs, and rerun SVD-PPMI, the best pattern-based approach and our complementary approaches. In Figure 4, we only illustrate the results on LEDs for Detection and BLESS for Direction. Observations on the other datasets are similar, thus are omitted. We have the following observations. First, with fewer extracted pairs, the OOP rates increase quickly, and all models generally perform worse. This is not surprising since a sparser $\mathcal{P}$ leads to a less informative SVD-PPMI matrix and less supervision on distributional models. Second, despite the increased OOP rates, our complementary methods consistently outperform SVD-PPMI and suffer less from increasing OOP rates especially on BLESS. Finally, among the four context encoders, HAN performs better than the others when the sampled rate is higher than 75%. However, with lower sampled rates, W2V is more robust than the others on BLESS but fails to exceed HAN on Eval.

7 Conclusion and Future Work

We propose complementing pattern-based and distributional methods for hypernymy detection. As far as we know, this is the first work along this line. We formally depict two types of sparsity that extracted pairs face, and indicate that pattern-based methods are invalid on the Type-II, i.e., out-of-pattern pairs. By analyzing common corpora and datasets, we confirm that OOP pairs are non-negligible for the task. To this end, we devise a complementary framework, where a pattern-based and distributional model handle IP and OOP pairs separately, while collaborating seamlessly to give unified scores. Oracle performance analysis shows that our framework has high potentials on several
datasets. Supervised by the pattern-based model, the distributional model shows robust capability of scoring OOP pairs and pushing the overall performance towards the oracle bounds.

In the future, we will extend the similar approach to multilingual (Yu et al., 2020a) or cross-lingual (Upadhyay et al., 2018) lexical entailment tasks. Moreover, one interesting direction is to use hyperbolic embeddings (Le et al., 2019; Balazevic et al., 2019) for pattern-based models due to their inherent modeling ability of hierarchies.

Acknowledgements

This paper was partially supported by the Early Career Scheme (ECS, No. 26206717), the General Research Fund (GRF, No. 16211520), and the Research Impact Fund (RIF, No. R6020-19) from the Research Grants Council (RGC) of Hong Kong.

References


Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2016. Order-embeddings of images and language. ICLR.


