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

Changlong Yu[1]* **Jialong Han**[2] **Peifeng Wang**[3] **Yangqiu Song**[1]
Hongming Zhang[1] **Wilfred Ng**[1] **Shuming Shi**[4]
[1]HKUST [2]Amazon [3]University of Southern California [4]Tencent 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 the good coverage and availability, free-text corpora are widely used to facilitate hypernymy detection, resulting in two lines of approaches: *pattern-based* and *distributional*.

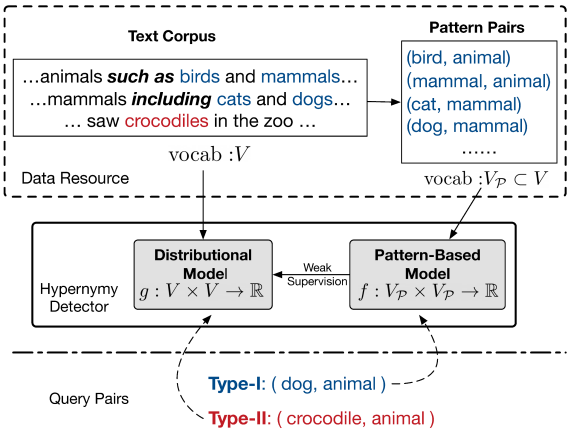


Figure 1: The overall framework of complementary methods for hypernymy detection from corpus. Different sparsity types of queried pairs are handled with pattern-based and distributional models respectively.

Pattern-based approaches employ pattern pairs (x, y) extracted via *Hearst-like patterns* (Hearst, 1992), *e.g.,* "y *such as* x" and "x *and other* y". An example of extracted pattern pairs from corpus are shown in Figure 1. Despite their high precision, the extracted pairs suffer from sparsity which comes in two folds *i.e.,* **Type-I**: x_q and y_q separately appear in some extracted pairs, but the pair (x_q, y_q) is absent *e.g.,* (dog, animal); or **Type-II**: either x_q or y_q is not involved in any extracted pair *e.g.,* (crocodile, animal).

Although matrix factorization (Roller et al., 2018) or embedding techniques (Vendrov et al., 2016; Nickel and Kiela, 2017; Le et al., 2019) are widely adopted to implement pattern-based approaches, they only relieve the **Type-I** sparsity and cannot generalize to unseen words appearing in the **Type-II** pairs. On the other hand, distribu-

* Work done when C. Yu, J. Han and P. Wang were with Tencent AI Lab.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6208–6217,
November 16–20, 2020.

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 **-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; Seitner 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 $\mathcal{C}$. By applying Hearst patterns on $\mathcal{C}$, a set of *extracted pairs* $\mathcal{P} \subseteq V \times V$, *i.e.,* `is-a` relationships $\{(x, y)\}$ $(x, y \in V)$, is obtained. As in Section 2, pattern-based approaches usually use $\mathcal{P}$ to perform matrix factorization or embedding learning. Due to their nature, only words "seen" in $\mathcal{P}$, or $V_{\mathcal{P}} = \{x \mid (x, y) \in \mathcal{P} \vee (y, x) \in \mathcal{P}\}$, 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.,* $V \setminus V_{\mathcal{P}}$, by *out-of-pattern* (or OOP) words.

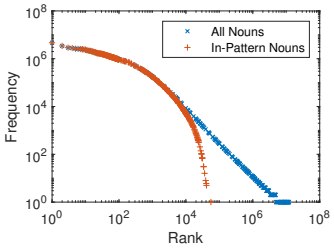


Figure 2: Corpus frequency of all nouns and IP nouns.

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) $V_{\mathcal{P}}$ covers well the most frequent nouns in V. For the top-10^4 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 $V_{\mathcal{P}}$, 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,

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

Table 1: Statistics of OOP pairs *w.r.t.* extracted pairs $\mathcal{P}$. OOP (All) is the number of OOP pairs while OOP (Hyper) is the number of OOP with true labels.

this region accounts for a non-negligible proportion of 19.9% of the total area.

With the likelihood of OOP nouns at hand, we are ready to roughly estimate the likelihood of encountering OOP pairs in practice. Suppose the two words in q are nouns independently sampled from the corpus distribution. Then the probability of q being OOP, *i.e.,* infeasible for pattern-based methods, is $1-(1-0.199)^2 = 35.8\%$. Even if y_q tends to bias towards more common words, the optimistic estimation is still above 19.9%.

Table 1 lists the actual portions of OOP pairs in several commonly used datasets *w.r.t.* $\mathcal{P}$ in Roller et al. (2018). Note that neither the datasets nor $\mathcal{P}$ 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 $\mathcal{P}$ from some text corpus $\mathcal{C}$. Unlike pattern-based approaches that depend solely on $\mathcal{P}$, our data resource also involves the corpus $\mathcal{C}$ for the sake of the distributional model.

Pattern-Based Model. The pattern-based model works on the extracted pairs $\mathcal{P}$ to serve in two roles. On the one hand, it is responsible for generalizing from statistics on $\mathcal{P}$ to score any in-pattern pair $q \in V_{\mathcal{P}} \times V_{\mathcal{P}}$ 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 $f : V_{\mathcal{P}} \times V_{\mathcal{P}} \to \mathbb{R}$.

Distributional Model. Different from the pattern-based model defined on IP pairs $V_{\mathcal{P}} \times V_{\mathcal{P}}$, 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 $\mathcal{C}$ 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 $\mathcal{C}$, *i.e.,* $\mathcal{C}(x)$ and $\mathcal{C}(y)$, to be $\mathbf{x}_h$ and $\mathbf{y}_H$, respectively, and makes predictions by a dot product, *i.e.,*

$$g(x, y) = \langle \mathbf{x}_h, \mathbf{y}_H \rangle.$$

Note that hypernymy is essentially asymmetric, so we distinguish $\mathbf{x}_h$ and $\mathbf{y}_H$ by the subscripts to reflect the asymmetry. In practice, we adopt networks with separate parameters for $\mathcal{C}(x)$ and $\mathcal{C}(y)$, which is detailed in the next section.

4.2 Encoding Queried Words

To implement the distributional model, we encode $\mathcal{C}(x)$ and $\mathcal{C}(y)$ into hypernymy-specific representations $\mathbf{x}_h$ and $\mathbf{y}_H$, 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 $\mathcal{C}(x)$ and $\mathcal{C}(y)$, this approach takes as input the pre-trained word vectors (Mikolov et al., 2013; Pennington et al., 2014) $\mathbf{x}$ and $\mathbf{y}$ of x and y, and apply two Multi-Layer Perceptrons (MLPs), respectively:

$$\mathbf{x}_h = \mathbf{MLP}_h(\mathbf{x}), \quad \mathbf{y}_H = \mathbf{MLP}_H(\mathbf{y}).$$

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 $V_\mathcal{P}$, 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 $c \in \mathcal{C}(x)$, the Neural Bag-of-Words (NBOW for short) encoder looks up and averages their pre-trained vectors $\mathbf{c}_j$ as $\mathbf{c}$, transforms $\mathbf{c}$ through a MLP, and averages the resulted vectors through a MEAN-pooling layer as $\mathbf{x}_h$:

$$\mathbf{x}_h = \frac{1}{|\mathcal{C}(x)|} \sum_{c \in \mathcal{C}(x)} \mathbf{MLP}_h(\mathbf{c}), \quad \mathbf{c} = \frac{1}{n} \sum_{j=1}^{n} \mathbf{c}_j.$$

To obtain $\mathbf{y}_H$, 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 $\mathbf{c}$ for the same transformation and averaging as for NBOW. Formally,

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

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 $\mathbf{x}_h$ and $\mathbf{y}_H$. 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 $\mathbf{c}$:

$$\alpha_j = \text{softmax}\Big(\mathbf{w}_a^\top \tanh(\mathbf{W}_a \mathbf{c}_j)\Big), \mathbf{c} = \sum_{j=1}^{n} \alpha_j \mathbf{c}_j.$$

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

$$\beta_i = \text{softmax}\Big(\mathbf{w}_b^\top \tanh(\mathbf{W}_b \mathbf{c}^{(i)})\Big), \mathbf{x}_h = \sum_{i=1}^{|\mathcal{C}(x)|} \beta_i \mathbf{c}^{(i)}.$$

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 $\mathbf{\Phi}$ 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.,*

$$l(x, y; \mathbf{\Phi}) = \Big(g(x, y; \mathbf{\Phi}) - f(x, y)\Big)^2.$$

Compared with the potentially large size of the output space, a set of random samples from it suffices to train the parameters $\mathbf{\Phi}$. For each IP word $x \in V_\mathcal{P}$, we uniformly sample k entries from Δ_x, the column and row involving x in the output space $V_\mathcal{P} \times V_\mathcal{P}$:

$$\Delta_x = \{(x, y) \mid y \in V_\mathcal{P}\} \cup \{(y, x) \mid y \in V_\mathcal{P}\}.$$

The sample for x is done on P_x, a uniform distribution over Δ_x. Finally, our objective is

$$\min \sum_{x \in V_\mathcal{P}} \mathcal{L}(x; \mathbf{\Phi}),$$

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

$$\mathcal{L}(x; \mathbf{\Phi}) = \sum_{i=1}^{k} \mathbb{E}_{(x^{(i)}, y^{(i)}) \sim P_x} l(x^{(i)}, y^{(i)}; \mathbf{\Phi}).$$

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), LEDS (Baroni et al., 2012), SHWARTZ (Shwartz 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 ρ 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`.

	Detection (AP)				**Dir.**(Acc.)	**Graded**(ρ)
	BLESS	EVAL	LEDS	SHWARTZ	BLESS	HYPERLEX
Cosine	.106	.172	.736	.175	.000	-0.107
WP	.100	.251	.880	.283	.636	0.147
invCL	.096	.211	**.887**	.220	.636	0.062
SLQS	.020	.166	.423	.240	.341	-0.130
W2V	.292	.255	.712	.453	.767	0.313
NBOW	.124	**.258**	.617	.500	**.975**	0.264
C2V	.027	.258	.659	.364	.791	**0.346**
HAN	**.346**	.250	.602	**.574**	**.975**	0.309

Table 2: Experimental results on OOP pairs.

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?

6.1 Performance on OOP Pairs

To ensure that our supervised distributional models are working effectively on OOP pairs, we evaluate on only OOP pairs under the aforementioned settings. Because pattern-based approaches trivially give the lowest scores to OOP pairs, we only compare with distributional approaches.

[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).

		Detection (AP)					Direction (Acc.)			Graded (ρ)
		BLESS	EVAL	LEDS	SHWARTZ	WBLESS	BLESS	WBLESS	BIBLESS	HYPERLEX
Pattern	Count	.486	.368	.710	.288	.744	.466	.690	.617	**.617**
	PPMI	.448	.341	.707	.277	.734	.466	.682	.611	.603
	SVD-Count	.651	.434	.812	.369	.904	.936	.842	.801	.518
	SVD-PPMI	.764	.463	.831	.409	.959	.959	.871	.847	.517
Supervised	SDSN	.749	.458	.841	.432	.958	.959	.874	.851	.588
ComHyper (*Ours*)	W2V	**.773**	.474	**.845**	.509	.957	.963	.873	.849	.522
	NBOW	.770	**.474**	.844	.510	.958	.970	.875	**.853**	.523
	C2V	.767	.472	.843	.480	.959	.966	.872	.847	.521
	HAN	.772	.473	.843	**.515**	**.959**	**.971**	**.875**	.853	.525
	Oracle	.801	.666	.876	.861	.959	.992	n/a	n/a	n/a

Table 3: Experimental results on all queried pairs. Best ones are marked bold while second-best ones underlined.

Table 2 demonstrates the results. Note that the 46 OOP pairs in WBLESS and BIBLESS are all labeled false, causing undefined AP and perfect Acc. scores, so we omit the corresponding columns to save space. Observing from Table 2, except on LEDS, our distributional models generally achieve higher scores than unsupervised approaches. Especially, on the BLESS dataset, Cosine even gets a zero Accuracy score because it is symmetric and cannot suggest the right direction. The higher AP and Accuracy scores suggest that, supervised by the pattern-based model, our distributional models can generate better relative rankings within the scope of OOP pairs.

6.2 Main Results and Case Study

When facing both IP and OOP pairs, it is not enough to rank both types of pairs separately, since downstream systems usually require comparable scores or a unified ranking. We evaluate on the entire datasets under the aforementioned settings. We only compare with pattern-based methods and supervised distributional models because they generally outperform unsupervised ones.

Table 3 provides the main results. Best results are marked bold, and second-best ones are underlined. To better interpret the results, we also provide "Oracle" scores, *i.e.,* the upper-bounds that complementary methods can achieve. For the Detection task, Oracle scores are obtained by assigning OOP pairs having hypernymy relationships (See Table 1) the maximum score and other ones the minimum. For BLESS of Direction, the Oracle score is computed by assuming perfect predictions for OOP pairs. The Oracle scores for WBLESS/BIBLESS of the Direction task and HYPERLEX of Graded Entailment are not straightforward to estimate, thus are omitted.

In Table 3, complementary methods lead to superior results on Detection and Direction tasks. In eight out of nine columns, the best and second best scores are both achieved by complementary methods. Especially, large improvements (up to 25.9%) are observed on SHWARTZ with a higher OOP rate and thus a higher Oracle. In general, the HAN encoder achieves better performances. By attending to the most informative contexts and words, the HAN encoder potentially captures distributional semantics that are relevant to hypernymy relationships between queried words. Note that the relative performances between different context encoders are not necessarily consistent with those in Table 2. This is because the overall performance is not only sensitive to the relative ranking of OOP pairs, but also to their absolute scores.

In addition, with the same extracted $\mathcal{P}$ as supervision signals, our proposed methods show a great superiority over the supervised method (SDSN in Table 3). Both SDSN and our complementary approaches could be regraded as *combining pattern-based and distributional model*. The key difference is that complementary methods solve Type-I sparsity with a pattern-based model, which proved to be better than distributional ones on this case, while SDSN uses a distributional model (though supervised) uniformly on both cases.

Case Study. To explain the superiority of the HAN encoder, we exemplify with two true-hypernymy OOP pairs from two Detection datasets, respectively. Here, the two hyponyms are both uncommon and OOP words. Therefore, pattern-based models such as SVD-PPMI simply assign the pairs with minimum scores and rank them at the bottom. But by examining their contexts in the textual cor-

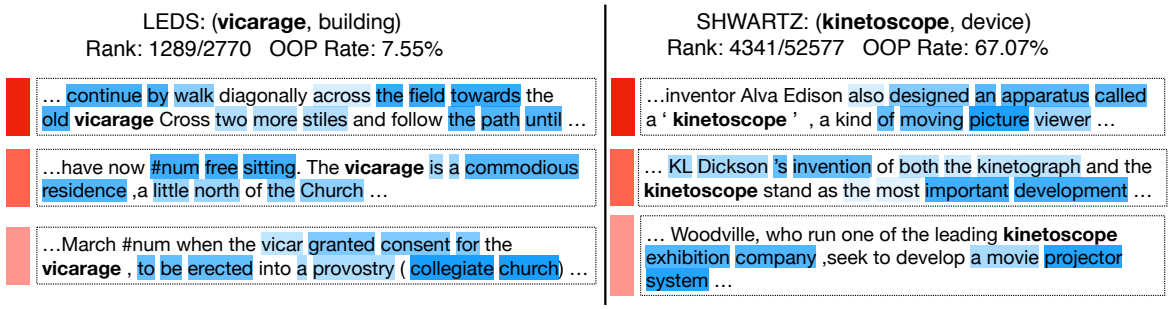
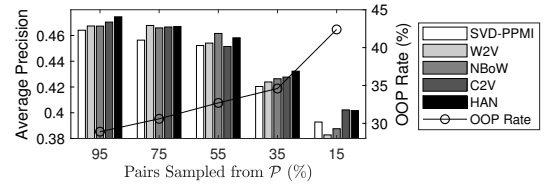
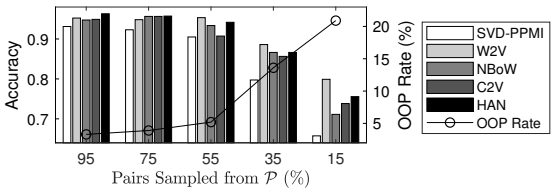


Figure 3: Case study of two queried pairs from two datasets, with OOP rates and actual ranks.



(a) Detection on EVAL



(b) Direction on BLESS

Figure 4: Performance comparison across different amounts of reducing pairs on EVAL and BLESS.

pus, the hypernymy relationships could have been inferred, and they could have been scored higher.

In Figure 3, we show the two OOP pairs, as well as their rank according to HAN and the OOP rates of the corresponding datasets. We also demonstrate the Top-3 contexts scored by HAN and visualize the context- and word-level attention weights. We observe that HAN can attend to informative contexts and words that help capture the semantics of the OOP word. For example, in LEDS, **vicarage** is OOP. HAN suggests three contexts that imply its meaning well. By reading the context words and phrases highlighted by HAN, *e.g., commodious residence*, and *collegiate church*, even people not knowing the word may guess it is a type of building. With our HAN-based distributional model, the pair is successfully promoted to top 50% in the ranking, well out of and above the bottom 7.55% of OOP pairs. Similar observations are drawn for the other pair, *i.e.,* (**kinetoscope**, device) with contexts *moving picture viewer*, and *movie projector system*.

We also observe that wrong predictions may be caused by extremely sparse contexts in the corpus such as *famicom* in the dataset SHWARTZ.

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

Tuan Luu Anh, Yi Tay, Siu Cheung Hui, and See Kiong Ng. 2016. Learning term embeddings for taxonomic relation identification using dynamic weighting neural network. In *EMNLP*, pages 403–413.

Ben Athiwaratkun and Andrew Gordon Wilson. 2018. Hierarchical density order embeddings. In *ICLR*.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. Multi-relational poincaré graph embeddings. In *Advances in Neural Information Processing Systems*, pages 4463–4473.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *EACL*, pages 23–32.

Marco Baroni and Alessandro Lenci. 2011. How we blessed distributional semantic evaluation. In *GEMS Workshop*, pages 1–10.

Or Biran and Kathleen McKeown. 2013. Classifying taxonomic relations between pairs of wikipedia articles. In *IJCNLP*, pages 788–794.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*, pages 632–642.

Haw-Shiuan Chang, Ziyun Wang, Luke Vilnis, and Andrew McCallum. 2018. Distributional inclusion vector embedding for unsupervised hypernymy detection. In *NAACL*, volume 1, pages 485–495.

Daoud Clarke. 2009. Context-theoretic semantics for natural language: an overview. In *Proceedings of the workshop on geometrical models of natural language semantics*, pages 112–119.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. Recognizing textual entailment: Models and applications. *Synthesis Lectures on Human Language Technologies*, 6(4):1–220.

Adriano Ferraresi. 2007. Building a very large corpus of english obtained by web crawling: ukwac. *Masters thesis, University of Bologna, Italy*.

Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In *Proceedings of the ACL*, pages 1199–1209, Baltimore, Maryland.

Octavian Ganea, Gary Becigneul, and Thomas Hofmann. 2018. Hyperbolic entailment cones for learning hierarchical embeddings. In *ICML*, pages 1646–1655.

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

Deepak Gupta, Rajkumar Pujari, Asif Ekbal, Pushpak Bhattacharyya, Anutosh Maitra, Tom Jain, and Shubhashis Sengupta. 2018. Can taxonomy help? improving semantic question matching using question taxonomy. In *COLING*, pages 499–513.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545.

Aishwarya Kamath, Jonas Pfeiffer, Edoardo Maria Ponti, Goran Glavaš, and Ivan Vulić. 2019. Specializing distributional vectors of all words for lexical entailment. In *Proceedings of the RepL4NLP*, pages 72–83, Florence, Italy.

Douwe Kiela, Laura Rimell, Ivan Vulić, and Stephen Clark. 2015. Exploiting image generality for lexical entailment detection. In *ACL*, pages 119–124.

Matthew Le, Stephen Roller, Laetitia Papaxanthos, Douwe Kiela, and Maximilian Nickel. 2019. Inferring concept hierarchies from text corpora via hyperbolic embeddings. In *ACL*, pages 3231–3241.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional lstm. In *CoNLL*, pages 51–61.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Kim Anh Nguyen, Maximilian Keper, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Hierarchical Embeddings for Hypernymy Detection and Directionality. In *EMNLP*, pages 233–243, Copenhagen, Denmark.

Maximilian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. *arXiv preprint arXiv:1806.03417*.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *NIPS*, pages 6338–6347.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Marek Rei, Daniela Gerz, and Ivan Vulić. 2018. Scoring lexical entailment with a supervised directional similarity network. In *ACL*, pages 638–643.

Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst patterns revisited: Automatic hypernym detection from large text corpora. In *ACL*, pages 358–363.

Enrico Santus, Alessandro Lenci, Qin Lu, and S Schulte im Walde. 2014. Chasing hypernyms in vector spaces with entropy. In *EACL*, pages 38–42.

Enrico Santus, Frances Yung, Alessandro Lenci, and Chu-Ren Huang. 2015. Evalution 1.0: an evolving semantic dataset for training and evaluation of distributional semantic models. In *Proceedings of the 4th Workshop on Linked Data in Linguistics: Resources and Applications*, pages 64–69.

Julian Seitner, Christian Bizer, Kai Eckert, Stefano Faralli, Robert Meusel, Heiko Paulheim, and Simone Paolo Ponzetto. 2016. A large database of hypernymy relations extracted from the web. In *LREC*.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *ACL*, pages 2389–2398, Berlin, Germany.

Vered Shwartz, Enrico Santus, and Dominik Schlechtweg. 2017. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In *EACL*, volume 1, pages 65–75.

Yangqiu Song, Haixun Wang, Zhongyuan Wang, Hongsong Li, and Weizhu Chen. 2011. Short text conceptualization using a probabilistic knowledgebase. In *IJCAI*, pages 2330–2336. AAAI Press.

Shyam Upadhyay, Yogarshi Vyas, Marine Carpuat, and Dan Roth. 2018. Robust cross-lingual hypernymy detection using dependency context. In *Proceedings of the NAACL*, pages 607–618, New Orleans, Louisiana.

Dmitry Ustalov, Nikolay Arefyev, Chris Biemann, and Alexander Panchenko. 2017. Negative sampling improves hypernymy extraction based on projection learning. In *Proceedings of the EACL*, pages 543–550, Valencia, Spain.

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

Ivan Vulić, Daniela Gerz, Douwe Kiela, Felix Hill, and Anna Korhonen. 2017. Hyperlex: A large-scale evaluation of graded lexical entailment. *Computational Linguistics*, 43(4):781–835.

Ivan Vulić, Goran Glavaš, Nikola Mrkšić, and Anna Korhonen. 2018. Post-specialisation: Retrofitting vectors of words unseen in lexical resources. In *NAACL*, volume 1, pages 516–527.

Ivan Vulić and Nikola Mrkšić. 2018. Specialising word vectors for lexical entailment. In *NAACL*, pages 1134–1145.

Chengyu Wang and Xiaofeng He. 2020. BiRRE: Learning bidirectional residual relation embeddings for supervised hypernymy detection. In *Proceedings of the ACL*, pages 3630–3640, Online.

Koki Washio and Tsuneaki Kato. 2018. Filling missing paths: Modeling co-occurrences of word pairs and dependency paths for recognizing lexical semantic relations. In *NAACL*, pages 1123–1133.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *COLING*, pages 2249–2259.

Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *COLING*, page 1015.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. 2012. Probase: A probabilistic taxonomy for text understanding. In *SIGMOD*, pages 481–492. ACM.

Patrick Xia, Shijie Wu, and Benjamin Van Durme. 2020. Which *bert? a survey organizing contextualized encoders. In *Proceedings of EMNLP 2020*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL*, pages 1480–1489.

Changlong Yu, Jialong Han, Haisong Zhang, and Wilfred Ng. 2020a. Hypernymy detection for low-resource languages via meta learning. In *Proceedings of the ACL*, Online.

Changlong Yu, Hongming Zhang, Yangqiu Song, Wilfred Ng, and Lifeng Shang. 2020b. Enriching large-scale eventuality knowledge graph with entailment relations. In *Proceedings of AKBC 2020*.

Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang. 2015. Learning term embeddings for hypernymy identification. In *IJCAI*.

Hongming Zhang, Muhao Chen, Haoyu Wang, Yangqiu Song, and Dan Roth. 2020. Analogous process structure induction for sub-event sequence prediction. In *Proceedings of EMNLP 2020*.