RESEARCH **Open Access**

`Hypert`: hypernymy-aware BERT with Hearst pattern exploitation for hypernym discovery

Geonil Yun[1], Yongjae Lee[2], A-Seong Moon[1] and Jaesung Lee[1,3*]

*Correspondence:
curseor@cau.ac.kr

[1] Department of Artificial Intelligence, Chung-Ang University, 84, Heukseok-ro, Dongjak-gu, Seoul 06974, Republic of Korea
[2] S2W Inc, 12, Pangyoyeok-ro 192beon-gil, Bundang-gu, Seongnam-si, Gyeonggi-do 13524, Republic of Korea
[3] AI/ML Innovation Research Center, Chung-Ang University, 84, Heukseok-ro, Dongjak-gu, Seoul 06974, Republic of Korea

Abstract

Hypernym discovery is challenging because it aims to find suitable instances for a given hyponym from a predefined hypernym vocabulary. Existing hypernym discovery methods used supervised learning with word embedding from word2vec. However, word2vec embedding suffers from low embedding quality regarding unseen or rare noun phrases because entire noun phrases are embedded into a single vector. Recently, prompting methods have attempted to find hypernyms using pretrained language models with masked prompts. Although language models alleviate the problem of w embeddings, general-purpose language models are ineffective for capturing hypernym relationships. Considering the hypernym relationship to be a linguistic domain, we introduce `Hypert`, which is further pretrained using masked language modeling with Hearst pattern sentences. To the best of our knowledge, this is the first attempt in the hypernym relationship discovery field. We also present a fine-tuning strategy for training `Hypert` with special input prompts for the hypernym discovery task. The proposed method outperformed the comparison methods and achieved statistically significant results in three subtasks of hypernym discovery. Additionally, we demonstrate the effectiveness of the several proposed components through an in-depth analysis. The code is available at: https://github.com/Gun1Yun/Hypert.

Keywords: Hypernym discovery, Hypernym relationship, Language model, Masked language modeling, Hearst pattern, Natural language processing

Introduction

Hypernymy denotes a semantic relationship characterized by a hierarchical connection between an abstract term and subordinate instances. To illustrate, when presented with a directive to enumerate various exemplars of *vehicle*, one may readily evoke representations such as an automobile, a watercraft, and an aircraft. In this context, these entities materialize as specific manifestations falling within the broader classification of *vehicle*, thereby designating *vehicle* as the hypernym and the entities mentioned above as their respective hyponyms.

The hypernym relation holds significant importance in natural language processing (NLP). This salient semantic connection assumes a crucial role in diverse NLP tasks, including question answering, ontology construction, textual entailment, and lexicon augmentation [1–3]. To facilitate these tasks, a large lexical database, WordNet, was introduced, which delineates semantic relations among words. However, using manual human effort, constructing such a resource is labor-intensive and time-consuming. Consequently, numerous studies have endeavored to automatically extract hypernym relationships from corpora [4–7].

Hypernym discovery entails identifying all possible instances of hypernyms for a given query within the vocabulary [8]. Recently, hypernym discovery studies have used word embeddings to capture the meaning and relationship between words. These studies can be further categorized into two classes: word2vec embedding methods [9–12] and prompting methods [13–16]. The word2vec embeddings are based on the distribution hypothesis to vectorize the meaning of a word into a vector space [17]. However, noun phrases are mapped to a single vector, and thus rare noun phrases in the corpus can be poorly embedded in this way.

The prompting method presents a potential solution to alleviate this issue, employing pretrained language models (PLMs) with subword tokenization algorithm [18]. The prompt takes sentences as input, which consist of a query token x and a `[MASK]` token (e.g., "A/An x is a `[MASK]`") [13]. Despite the promise that harnessing PLMs holds for addressing challenges associated with infrequent terms, certain limitations persist. Only one word can be predicted from the prompt using one `[mask]` token, even though the gold hypernyms can be multiple words. Additionally, it is essential to acknowledge that PLMs, in their original design, are not inherently geared towards discerning hypernym relationships. Hence, customary procedures involving supplementary rounds of pretraining and subsequent fine-tuning are often employed to tailor the PLM to the specific demands of domain-specific tasks [19, 20].

Within the realm of hypernym discovery, the application of these processes (i.e., pretraining and fine-tuning) remains unexplored. An inherent shortcoming of existing studies concerning hypernym discovery is their limited efficacy in deciphering the semantics and hypernymic relationships inherent in noun phrases. Approaches grounded in word2vec suffer vulnerability when confronted with noun phrases due to their amalgamation into singular vectors. Even though prompting methodologies offer potential relief, the all-encompassing nature of general-purpose PLMs renders them unsuited for comprehending hypernymic information, given their training on broad-spectrum sentences.

To address the limitations posed by existing approaches, in this work, we propose `Hypert`, a hypernymy-aware pretrained language model for hypernym discovery that harnesses Hearst pattern sentences. The method we present aligns with established domain adaptation practices for PLMs. To elucidate, the PLM is subjected to additional training using a corpus consisting of sentences based on the Hearst pattern, thereby heightening its sensitivity to hypernymic constructs. The Hearst pattern, though simple, possesses a solid foundation, thereby facilitating the construction of a corpus rich in hypernym relationships [21]. More specifically, we employ the extended Hearst patterns, which embody the core concept of the original Hearst patterns. Then, the augmented pretrained model undergoes fine-tuning on a dedicated dataset tailored for the hypernym discovery task, employing a specialized input prompt. By amplifying the hypernym awareness of language models through supplementary pretraining, our approach envisages an elevation in the efficacy of hypernym discovery, ultimately yielding enhanced performance.

The effectiveness of the proposed method is assessed through a comparison with conventional word2vec-based methods and the prompting approach, utilizing the SemEval-2018 task9 dataset [8]. The experimental results show that the proposed method significantly outperforms the comparison methods. Furthermore, an in-depth analysis was also conducted to evaluate the effectiveness of individual components of the proposed method. We also presented the distribution of utilized Hearst patterns in the corpus and analyzed them. Additionally, our investigation reveals the efficacy of further pretraining, contrasting favorably with BERT [22]. To discover the robustness of our method, we conduct a comparative evaluation of prediction outcomes for rare words. Lastly, the *t*-distributed stochastic neighbor embedding (tSNE) plots are presented to visually represent the classification token (`[CLS]`) embedding representation space in the context of hypernymy.

The contributions of this work are be summarized as follows:

- We propose a further pretraining method to enhance the hypernymy awareness of language models by denoising Hearst pattern sentences from a corpus and using a special input prompt for fine-tuning to achieve the best performance among the comparison methods.
- An in-depth analysis is conducted regarding the further pretraining. The results reveal that the proposed further pretraining method improves performance compared to BERT.
- The proposed method solves the low embedding quality of noun phrases that rarely appear and demonstrates this through the case study.

Related work

In traditional hypernym discovery and detection studies, pattern-based methods have been used to identify hypernym-hyponym pairs from a corpus. In the work of [1], the author defined lexico-syntactic patterns, called Hearst patterns, which can automatically filter hypernym-hyponym pairs from large corpora. For instance, a pattern like “y such as x” indicates that x is a hypernym of y. Because such pattern-based syntactic relation extraction is a good starting point, this seminal research affects lots of subsequent studies that can be roughly divided into distributional similarity-based, ontological knowledge-based, and machine learning-based methods.

Several studies based on distributional similarity are inspired by the distributional inclusion hypothesis [23–25], which assumes that the hypernym can substitute for its hyponym. Based on this hypothesis, researchers have proposed unsupervised distributional measures with asymmetric scoring functions. In the work of [26], *WeedsPrec*, a precision-based similarity measure was proposed to quantify the weighted inclusion of a narrow term x to a broad term y (i.e., $\langle x \rightarrow y \rangle$). Additionally, *coWeeds* [27], the geometric average of cosine similarity with *WeedsPrec*, was proposed. In the work of [28], *ClarkeDE*, a variant of *WeedsPrec*, was proposed to compute the degree of inclusion. Alternatively, *invCL* [27] considered the inclusion of a narrow term x to a broad term y and the exclusion of y to x using the *ClarkeDE* measure. However, distributional inclusion hypothesis methods require specific similarity measures to identify semantic relationships. In addition, there may also be a sparsity problem for noun phrases that contain more than one word, making it challenging to identify hypernym relationships.

Since the concept of word2vec embedding was introduced [17], researchers have attempted to use it as a word representation in supervised learning. In the work of [9], the authors demonstrated how to learn semantic hierarchies from word embeddings via projection learning. Projection learning achieved remarkable improvement in the Chinese hypernym detection task compared to other pattern-based and distributed methods using learnable piecewise uniform projection matrices that map queries to various hypernym representations. In addition, RMM [11] repeatedly uses a shared weight projection matrix for a given query with word2vec embeddings, assuming that hypernyms may come from various conceptual hierarchy levels. RMM exploits the attention mechanism [29] and residual connection [30] to capture corresponding candidate hypernyms. Furthermore, SPON [12] uses word2vec embeddings with a simple neural network to enforce hypernym relationship properties, asymmetry, and transitivity, as a soft constraint. However, these methods are vulnerable to rare or unseen words because they map noun phrases to a single vector. Moreover, traditional methods are highly inefficient when dealing with out-of-vocabulary terms because they either use random vectors or require retraining the entire word representation from scratch.

Ontological knowledge also aids hypernym discovery as many distant supervision approaches rely on existing ontologies. [31] utilized BabelNet [32], a multilingual lexicalized semantic network and ontology, to extract sentences containing terms linked by hypernym relations within BabelNet. The sentences are incorporated into the training data when these terms exhibit hypernymic connections. With this training data, they built classifiers to determine whether a given sentence contains expressions indicative of hypernymic relationships. Similarly, [33] use BabelNet [32] and embed pairs of its synsets (namely, term-hypernym) into the sense embedding spaces [34]. For example, *Apple* and the concept *company* could form a term-hypernym pair. In that embedding spaces, the authors learn a hypernym transformation matrix of all term-hypernym pairs and then compute similarity values over the pairs.

The machine learning-based methods try to identify the patterns of hypernym relationships from given data. For example, HyperNET [35] used an LSTM-based network to recognize hypernym syntactic patterns by representing dependency tree paths as sequential data. Despite the many efforts to find hypernyms with syntactic patterns, pattern-based methods have sparsity problems in which hypernym-hyponym pairs that match the pattern are rare in the corpus [10, 36]. Recently, matrix factorization techniques, such as Singular Value Decomposition, have been used to mitigate the sparsity problem of pattern-based methods and showed improved results [21].

With the emergence of the transformer [37] in NLP, many researchers have recently used transformer-based PLM, such as BERT [22], in various NLP tasks and applications. The transformer [37] is a novel encoder-decoder network architecture based solely on the attention mechanism, called self-attention, and does not rely on recurrence or other convolutions. In addition, BERT [22] is an effective PLM that can be fine-tuned for a wide range of NLP tasks using the encoder block of the transformer and has achieved state-of-the-art results on 11 benchmark datasets. Following the success of the transformer and BERT, many BERT variants have been proposed [19, 20, 38–40]. For example, BioBERT [20] and FinBERT [19] improved their performance by further pre-training BERT with a domain-specific corpus.

To use a PLM for identifying hypernym relationships, [16] evaluated the ability of BERT through human language experiments called prompting and demonstrated proficient results in hypernym retrieval. Moreover, several studies have used human language experiments to evaluate the linguistic knowledge of PLM [41–43]. For example, singular and plural prompts were used to probe the hypernymy knowledge of BERT [15]. In addition, various prompts, specifically Hearst patterns, natural sentences, and hand-written context, have been used to find hypernyms with BERT [14]. In the work of [13], the authors investigated the performance of general-domain and domain-adapted language models on financial hypernymy pair datasets using prompting masked language models. However, the prompting approaches can incur unstable identification because the performance varies depending on the prompt type. In addition, due to the format of the prompt, only one token within the vocabulary of the language model used can be predicted.

Lastly, a group of studies adopts a strategy to hybridize multiple approaches to maximize identification performance. CRIM [10] can be a notable hybrid approach that combines pattern-based and projection learning methods. The supervised learning approach of CRIM uses projection learning with multiple parallel projection matrices. The pattern-based part of CRIM uses Hearst patterns to assign weight to word2vec embeddings. Then, the cosine similarity is used as a score between two words. Like CRIM, [44] proposed a hybrid approach to discover hypernym relations using a pattern-based and distributional model. Their model begins with finding seed hypernyms using extended Hearst patterns, then adds the hypernyms of the nearest neighbor.

Inspired by the effectiveness and generality of PLM, this study aims to find hypernyms using a language model further pretrained by masked language modeling (MLM) using Hearst pattern sentences. We employ a specially formatted input sentence consisting of noun phrases and special tokens. Moreover, projection learning is adopted to capture semantic relationships between noun-phrase embeddings.

Proposed method

This section introduces the proposed hypernym discovery system with `Hypert`. We illustrate the architecture of the proposed hypernym discovery system in Fig. 1. Data preparation (left) presents the dataset and generation of the input data for the model. Model training (middle) represents the model training and inference for `Hypert`. Prediction (right) sorts the model output, taking the top 15 candidates.

Data preparation

This study employs the SemEval-2018 Task9 (Hypernym Discovery) [8] dataset for the benchmark. The dataset contains five subtasks: three subtasks for general purposes for three languages (English, Spanish, and Italian) and two domain-specific subtasks in English (Medical and Music). We considered three of these subtasks for the experiments: 1A (English), 2A (Medical), and 2B (Music). Table 1 presents the examples of the query-hypernym pair dataset. The queries and candidate hypernyms in the

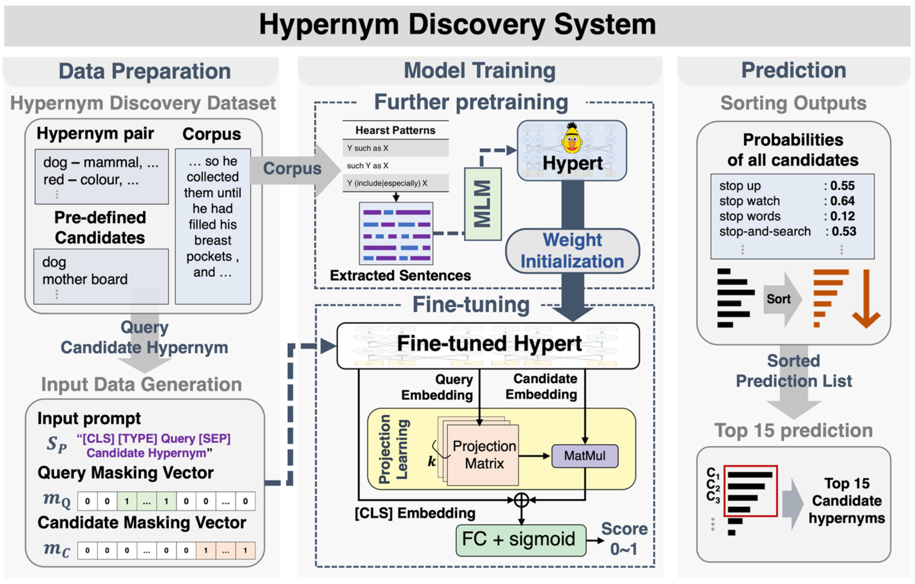


Fig. 1 Architecture of the hypernym discovery system in this study. (Left) Data preparation used for SemEval-2018 task9 to generate input data. (Middle) Model training applied the `Hypert` of each subtask to initialize the decision model, adopting projection learning. (Right) Prediction sorts the top 15 output results

Table 1 Examples from the SemEval2018 Task9-hypernym discovery dataset

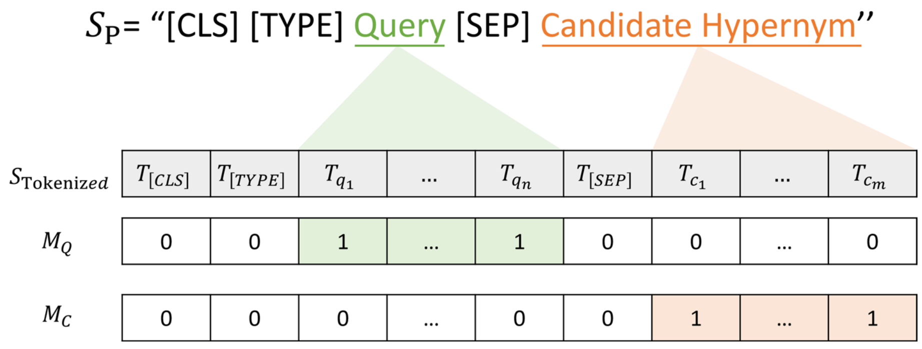
Subtask	Query	Type	Gold Hypernym(s)
1A English	fuse	Concept	igniter, microcomputer, electrical conductance, electronic component, party, resistor
2A Medical	solitary pulmonary nodule	Concept	clinical finding, lung mass, single lesion, nodule
2B Music	Louis Armstrong	Entity	cornetist, trumpeter, jazzman, jazz musician, musician, person

Table 2 Dataset statistics for each subtask

Subtask	Corpus size	# of			
		Vocabulary	Train	Valid	Test
1A(English)	16G	218,753	1500	50	1500
2A(Medical)	800M	93,888	500	15	500
2B(Music)	500M	69,118	500	15	500

hypernym discovery task can be one-word, two-word, or three-word noun phrases. Each query can have up to 15 gold hypernyms. In addition, the query is given with a noun phrase and type of query, which is either a concept or entity.

Table 2 lists the statistics of each subtask. Each subtask comprises a corpus, vocabulary, and the training, validation, and testing sets. The corpus was used to train word embeddings. The vocabulary includes noun phrases that can be target hypernyms. The datasets contain query-gold hypernym pairs, where the number of train, the number of valid, and the number of test are the number of queries for training, validation, and


S_P= "[CLS] [TYPE] Query [SEP] Candidate Hypernym"
$S_{\text{Tokenized}}$
$T_{[CLS]}$ $T_{[TYPE]}$ T_{q_1} ... T_{q_n} $T_{[SEP]}$ T_{c_1} ... T_{c_m}
M_Q
0 0 1 ... 1 0 0 ... 0
M_C
0 0 0 ... 0 0 1 ... 1
Fig. 2 Generation of M_Q and M_C for S_{prompt}. M_Q is 1 for query tokens and 0 for all other tokens; M_C is 1 for hypernym tokens and 0 for all other tokens

testing, respectively, and the number of vocabulary represents the number of candidate hypernyms. More dataset details can be found in the work of [8].

We used extended Hearst patterns to build a hypernym-related corpus from the given corpus.[1] 47 patterns were used to extract hypernym-related sentences. Specifically, the number of extracted sentences for 1A, 2A, and 2B are 5 M, 137K, and 153K, respectively. These are 4%, 4%, and 3% of the total sentences for each subtask. Although there may be more efficient patterns, the patterns exploited in this study are sufficient to build a training corpus and achieve the best performance among the comparison methods.

We used a special input prompt sentence S_P as an input sentence of the PLM:

$$S_P = \text{“}\texttt{[CLS] [TYPE]}\ \mathrm{Q}\ \texttt{[SEP]}\ \mathrm{C}'', \quad (1)$$

where Q and C are the query and candidate hypernym terms, respectively. The `[CLS]` token is a special token that is always the first of every input sequence. This special token is a classification token used as the aggregate sequence representation. The `[SEP]` token is also a special token to separate sentences. We used the `[CLS]` token embedding to represent the hypernym relationship and the `[SEP]` token to separate query and candidate hypernym terms in this study. However, in the hypernym discovery task, the query is given with the type of query: concept or entity. For instance, the query "*fuse*" is a concept, and the query "*Louis Armstrong*" is an entity. To provide type information for the input query, we added `[CON]` and `[ENT]` special tokens, referring to concept and entity types. Thus, the `[Type]` token of S_P is `[CON]` when the query type is a concept and is `[ENT]` when the query type is an entity.

However, both input terms, the query and candidate hypernym, were split into multiple tokens because of subword tokenization. Thus, the number of each term token varies for each S_P. The span of the subword tokens that correspond to each term must be identified to obtain word embeddings. To achieve this, we generated $M_Q, M_C \in \mathbb{R}^{l\times 1}$, masking vectors as span information vectors. The length of S_P is l. Figure 2 illustrates the generated M_Q and M_C vectors for the given S_P. Both vectors are one-hot vectors consisting of 0 and 1. In addition, M_Q is the query token span vector, and M_C is the candidate hypernym token span vector. We set the masking vectors to 1 for each token span and

[1] https://github.com/abyssnlp/Hearst-Hypernym-Extractor.

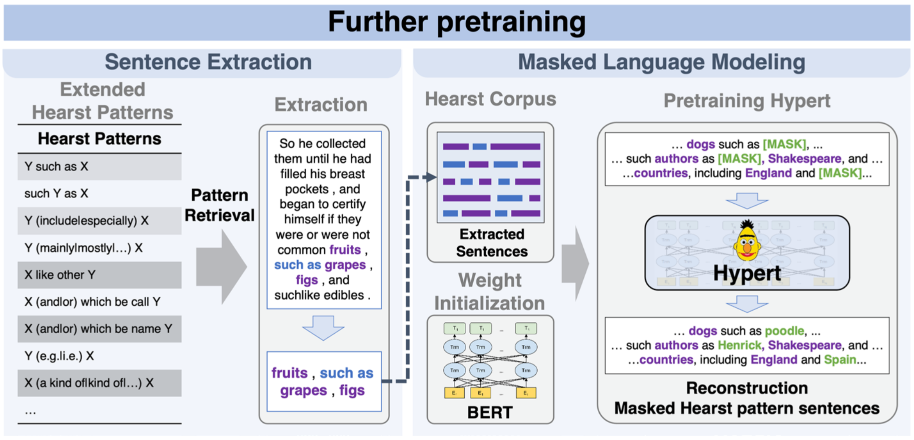

Further pretraining
Sentence Extraction
Masked Language Modeling
Extended Hearst Patterns
Hearst Patterns
Y such as X
such Y as X
Y (includelespecially) X
Y (mainlylmostlyl…) X
X like other Y
X (andlor) which be call Y
X (andlor) which be name Y
Y (e.g.li.e.) X
X (a kind oflkind ofl…) X
…
Pattern Retrieval
Extraction
So he collected them until he had filled his breast pockets , and began to certify himself if they were or were not common fruits , such as grapes , figs , and suchlike edibles .
fruits , such as grapes , figs
Hearst Corpus
Extracted Sentences
Weight Initialization
BERT
Pretraining Hypert
… dogs such as [MASK], …
… such authors as [MASK], Shakespeare, and …
…countries, including England and [MASK]...
Hypert
… dogs such as poodle, …
… such authors as Henrick, Shakespeare, and …
…countries, including England and Spain...
Reconstruction
Masked Hearst pattern sentences

Fig. 3 Overview of the further pretraining method. (Left) Sentence extraction used extended Hearst patterns for sentence retrieval. (Right) Masked language modeling exploits each extracted subtask corpus and creates the Hypert for each subtask

0 for the others. For instance, M_Q was set to 1 for the query token span and 0 for the others.

`Hypert`: hypernymy-aware BERT

We introduce `Hypert`, a PLM for the hypernym relationship. The overall process of further pretraining is illustrated in Fig. 3. Sentence extraction (left) indicates the pattern retrieval process to build a pretraining corpus. In addition, MLM (right) uses extracted sentences and generates `Hypert`, the hypernymy-aware BERT, for each subtask.

We hypothesize that the language models can learn hypernym relation knowledge from specific sentences in this study. As with BioBERT [20], FinBERT [19], and DarkBERT [45], domain-specific tasks are significantly improved with a further pretraining using the domain corpus. By considering the hypernym relationship to be a specific domain, the language model can be further pretrained on the hypernymy-related corpus to improve hypernym relationship awareness. To achieve this, sentences representing a hypernym relation are required to construct a hypernymy-related domain corpus. The Hearst pattern is devised to detect hypernym-hyponym pairs from the corpus [1]. For example, if the sentence "mammal such as dog" matches "*y* such as *x*," one of the Hearst patterns, then (dog, mammal) can be extracted as a hypernym relationship. We exploited the Hearst pattern to identify sentences that contain hypernym relationships. Sentences matching the Hearst patterns were extracted to build the hypernym-related corpus. When extracting sentences, only the part matching the pattern in the sentence was extracted, not the entire sentence containing the pattern.

Similar to BioBERT and FinBERT, `Hypert` is initialized with BERT, a pretrained model consisting of transformer encoder layers before further pretraining [22]. We also employed the BERT tokenizer and added special tokens `[CON]` and `[ENT]`. Subword tokenization in BERT splits words into multiple subtokens defined in the vocabulary pool of BERT, allowing rare or unseen words to be represented with proper subtoken embeddings. However, this advantage leads to the need for M_Q and M_C, as mentioned previously. In contrast, further pretraining on a corpus related to hypernymy allows

BERT to gain a multiple-perspective understanding of the critical information regarding the hypernym relationship between input query tokens and candidate tokens. While MLM effectively improves contextual hypernymy understanding, next sentence prediction is irrelevant to this task as we are unconcerned with the relationship between two consecutive sentences. Therefore, the pretrained model initialized with $\text{BERT}_{\text{base}}$ is further trained using the constructed corpus from above without the next sentence prediction objective. Additionally, `Hypert` is generated separately for each constructed subtask corpus. In other words, there are three `Hypert` models for 1A,[2] 2A,[3] and 2B.[4]

Fine-tuning and prediction

We present a fine-tuning and prediction process using `Hypert`. The output of `Hypert` $H \in \mathbb{R}^{l \times d_{\text{model}}}$ is obtained by $f(S_{\text{P}})$, where f is the proposed `Hypert`. The length of the input sentence tokens is l, and d_{model} is 768, the dimension of the $\text{BERT}_{\text{base}}$ model:

$$H = f(S_{\text{P}}). \tag{2}$$

The embedding of each term is computed by averaging the token embedding, which can be obtained by multiplying H and each span information vector described above, divided by the sum of masking vectors as follows:

$$\hat{e}_q = \frac{H^{\top} \cdot M_Q}{\sum_i^l M_{Q_i}} \tag{3}$$

and

$$\hat{e}_c = \frac{H^{\top} \cdot M_C}{\sum_i^l M_{C_i}}, \tag{4}$$

where $\hat{e}_q, \hat{e}_c \in \mathbb{R}^{d_{\text{model}} \times 1}$ are embeddings of query and candidate hypernyms. In addition, the embedding of the `[CLS]` token $\hat{e}_{\texttt{[CLS]}} \in \mathbb{R}^{d_{\text{model}} \times 1}$ is obtained by taking the first index of the final hidden state of `Hypert` H. An affine transformation is applied to reduce the dimensions of each embedding. Thus, $e_{\texttt{[CLS]}}$ can be defined as follows:

$$e_{\texttt{[CLS]}} = W_{\texttt{[CLS]}} \cdot \hat{e}_{\texttt{[CLS]}} + b_{\texttt{[CLS]}}. \tag{5}$$

The query embedding e_q can be given as follows:

$$e_q = W_Q \cdot \hat{e}_q + b_Q, \tag{6}$$

and the candidate hypernym embedding is defined as follows:

$$e_c = W_C \cdot \hat{e}_c + b_C. \tag{7}$$

The dimensions of query embedding $\hat{e}_q$ and candidate hypernym embedding $\hat{e}_c$ are reduced by d with $W_Q, W_C \in \mathbb{R}^{d \times d_{\text{model}}}$, and $b_Q, b_C \in \mathbb{R}^{d \times 1}$. The embedding of the

[2] https://huggingface.co/HeroGeonil/Hypert.

[3] https://huggingface.co/HeroGeonil/Hypert-medical.

[4] https://huggingface.co/HeroGeonil/Hypert-music.

`[CLS]` token $\hat{e}_{\texttt{[CLS]}}$ is used in the last layer, so the dimensions are reduced by the number of projection matrices k with $W_{\texttt{[CLS]}} \in \mathbb{R}^{k \times d_{\text{model}}}$, and $b_{\texttt{[CLS]}} \in \mathbb{R}^{k \times 1}$. All W and b are learnable wnd biases.

Previous studies have used projection learning for the supervised approach [9–11]. In this study, we adopted the projection learning method, using projection matrices Φ to capture the relationship between the query and candidate hypernym embeddings produced by `Hypert`. The projection matrix was created by applying a normal distribution $\mathcal{N}(0, 1/d)$ as noise to the identity matrix as follows:

$$\Phi_i = I + \epsilon_i, \; \epsilon_i \sim \mathcal{N}(0, 1/d)\,, \tag{8}$$

where I denotes an identity matrix, and $\epsilon_i \in \mathbb{R}^{d \times d}$ represents the ith noise term sampled from a normal distribution. Each projection matrix Φ_i was generated by adding the individual noise ϵ_i to I.

Then, the query embedding e_q was multiplied by multiple k square projection matrices $\Phi_i \in \mathbb{R}^{d \times d}$ to obtain projected matrices P_i, where $i = \{1, ..., k\}$. P can be defined as

$$P_i = \left(\Phi_i \cdot e_q\right)^{\top}. \tag{9}$$

Finally, the score matrix $s \in \mathbb{R}^{k \times 1}$ was computed using $P \in \mathbb{R}^{k \times d}$ and the candidate hypernym embedding e_c as follows:

$$s = P \cdot e_c. \tag{10}$$

The embedding of the `[CLS]` token was used for relation representation. To achieve this, $e_{\texttt{[CLS]}}$ and s were concatenated to $F \in \mathbb{R}^{2k \times 1}$ as follows:

$$F = [\; e_{\texttt{[CLS]}} \;;\; s \;], \tag{11}$$

The input prompt includes the query and candidate hypernyms; thus, the output of the proposed model is the probability of a hypernym relationship. Thus, the final layer is a feedforward network with a sigmoid activation function to output [0, 1] as follows:

$$o = \sigma(W_o \cdot F + b_o), \tag{12}$$

where o is the probability of a hypernym relationship, W_o and b_o represent learnable parameters, and σ denotes a sigmoid function.

Algorithm 1 Inference of the proposed method

```
Input: Query $Q$, candidate hypernyms $C = \{c^{(1)}, ..., c^{(|C|)}\}$, Proposed Model $\mathcal{M}$
Output: Prediction List $K$
1:  $L \leftarrow$ empty list
2:  $K \leftarrow$ empty list
3:  for $i \leftarrow 1$ to $|C|$ do
4:      $S_P \leftarrow make\_prompt(Q, c^{(i)})$                ▷ generate an input prompt
5:      $M_Q, M_C \leftarrow make\_masking\_vector(S_P)$          ▷ generate masking vectors
6:      $o_{c^{(i)}} \leftarrow \mathcal{M}(S_P, M_Q, M_C)$
7:      $L$.append($o_{c^{(i)}}$)                                  ▷ append score for $c^{(i)}$
8:  end for
9:  $I \leftarrow \arg\mathrm{sort}_{1 \le i \le |L|} L_i$          ▷ sort in order of high score
10: for $i \leftarrow 1$ to 15 do
11:     $K$.append($c^{(I_i)}$)
12: end for
```

However, hypernym discovery aims to retrieve suitable hypernyms from a given predefined vocabulary. Therefore, as many input prompts as the number of all words in the predefined vocabulary are generated and calculated for one query. The number of o for each query equals the number of words in the predefined vocabulary. Each query has a maximum of 15 gold hypernyms; thus, we sorted the output, taking the top 15 candidates. The inference of the proposed method is described in Algorithm 1. In Line 4, *make_prompt* is a function that generates S_P for the query Q and candidate hypernym $c^{(i)}$, as demonstrated in Eq. 1 whereas the function *make_masking_vectors* of Line 5 produces masking vectors M_Q and M_C illustrated in Fig. 2 for a given S_P.

Experimental results

This section presents the performance of the proposed and conventional methods. In addition, it describes the experimental settings for the hypernym discovery dataset, evaluation measures, and employed statistical tests.

Experimental settings

The HuggingFace transformers library[5] with PyTorch [46] was used for the implementations. The experiment was conducted using an Intel i9-10980XE, three NVIDIA GeForce RTX 3090 GPUs, and 128GB RAM. In addition, distributed training was employed by using Data Parallel functionality in PyTorch. In the further pretraining of `Hypert`, the BERT$_{\text{base}}$ model[6] initialized the PLM. We set the batch size, learning rate, and cosine scheduler warm-up steps to 216, 5e-5, and 500, respectively. The maximum training step was also limited to 10k for each subtask dataset.

After further pretraining, the proposed pretrained model was fine-tuned on the training dataset of each subtask. In the fine-tuning model, k was set to 24, and d was set to 200. We set the batch size to 32 and the maximum epoch to 15 for training.

[5] https://github.com/huggingface/transformers.

[6] https://huggingface.co/bert-base-uncased.

In addition, we used negative sampling because the dataset consists of only positive samples. For each positive sample, 50 negative samples were generated. The model with the best validation mean average precision (MAP) score epoch was used for testing. The loss function was set to binary cross-entropy loss and minimized using the AdamW optimizer [47]. The binary cross-entropy loss function is defined as follows:

$$L(q, c, t) = t \times \log(y) + (1 - t) \times \log(1 - y), \tag{13}$$

where L, q, c, t, and y refer to the loss value, query, candidate hypernym, label, and prediction of the proposed model. The label is 0 for negative pairs and 1 for positive pairs. We conducted hold-out cross-validation for each experiment. The training, validation, and testing sets were combined and randomly selected in equal proportions to the given split. For each subtask dataset, the experiment was repeated 10 times. We obtained 10 performance values for each measure.

The proposed method was compared to three conventional hypernym discovery methods: RMM, SPON, and prompting BERT. Details of each method are provided below.

- **RMM** [11]: This method utilizes a projection matrix with word2vec embeddings. The shared projection matrix is applied to hyponym term embedding recurrently to obtain representations of higher concept-level of words. To obtain word2vec embeddings for RMM, we set embedding dimensions and window sizes to 200 and seven, respectively. Then it is trained based on ten negative samples with ten epochs training for each given corpus. Next, to train the RMM model, we set the batch size to 32, and the number of negative samples was set to 50. The maximum training epoch was set to 1,000 with 200 patience. Lastly, the best validation MAP model was selected for testing. RMM was chosen for comparison because it is a representative method based on the projection matrix.
- **SPON** [12]: This method creates a distance-to-satisfaction vector for a given hyponym and candidate hypernym. The output representation is subtracted from the candidate hypernym term. All the parameter settings and procedures for obtaining word2vec embeddings and training SPON are the same as the experimental settings of RMM. Again, the best validation MAP model was selected for testing. In our comparison, SPON was chosen because it effectively reflects asymmetricity and transitivity properties which are essential for identifying hypernym relations.
- **Prompting BERT (is-a)** [14]: Prompting BERT generates hypernym for a given prompt by predicting `[MASK]` token. Because the original BERT is used directly, an additional fine-tuning process is unnecessary. In our experiment, we considered the prompt "A/An x is a `[MASK]`." because of its simplicity in identifying hypernym relationships. The strategy to prompt BERT was chosen to validate the superiority between `Hypert` and the pre-trained language model for our task. Another reason for choosing the prompting strategy is that it does not rely on distributional similarity, for example, word2vec, in contrast to RMM and SPON.
- **Prompting BERT (such as)** [14]: Similar to experimental settings of Prompting BERT (is-a), we considered a prompt "A `[MASK]` such as A/An x." (such as) because of its superior performance in identifying hypernym relationships.

Next, we employed three evaluation measures as follows:

- **Mean average precision (MAP)**: The MAP is the mean of average precision, the average of each obtained hypernym from the search space, for a given query word. The MAP is defined as

$$\mathrm{MAP} = \frac{1}{|Q|}\sum_{q\in Q}^{|Q|}\mathrm{AP}(q), \tag{14}$$

  where Q and $|Q|$ refer to the given set of query words and the size of the set, respectively.
- **Mean reciprocal rank (MRR).** The MRR is usually used to evaluate the effectiveness of an information retrieval system [48, 49]. The reciprocal rank is the reciprocal of the first relevant or correct outcomes. The MRR is the average of the reciprocal rank for each given query word and is defined as

$$\mathrm{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}, \tag{15}$$

  where the $rank_i$ refers to the rank position of the first correct hypernym of i-th query.
- **Precision at k (P@k).** The P@k metric calculates the top-k hypernym outcome precision and is defined as

$$\mathrm{P@}k = \frac{\mathrm{TP@}k}{(\mathrm{TP@}k) + (\mathrm{FP@}k)}, \tag{16}$$

  where TP and FP refer to the true positive and false positive, respectively. Specifically, we set the cut-off threshold k to 1, 3, 5, or 15 in this study.

We compared each method on different iterations using the Wilcoxon signed-rank test [50] because we are interested in the superiority of the proposed method over the comparison methods. We let d_i be the difference between the performance of the two methods on the ith iteration. The differences were ranked according to their absolute values: the smallest d_i was assigned to the first rank. In the case of ties, average ranks were assigned. We let R^+ be the sum of the ranks for the iterations on which the compared method outperforms the proposed method, defined as

$$R^+ = \sum_{d_i>0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i)$$

and R^- is the opposite, as follows:

$$R^- = \sum_{d_i<0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i)$$

Then, according to the critical values for the Wilcoxon's test, for a confidence level of $\alpha = 0.05$ and with $N = 10$, the difference between the compared methods is significant if $\min(R^+, R^-) \leq 8$. In this case, the null hypothesis of equal performance is rejected.

Table 3 Model performance on SemEval2018-task9 dataset

Subtask	Evaluation measures	Proposed method	RMM	SPON	Prompting BERT (is a)	Prompting BERT (such as)
1A English	MRR	**38.68 ± 2.00**	27.21 ± 3.50	24.94 ± 4.10	19.74 ± 0.44	19.77 ± 0.41
	MAP	**24.17 ± 1.26**	18.25 ± 1.16	15.72 ± 1.75	11.43 ± 0.16	10.93 ± 0.19
	P@1	**29.57 ± 1.98**	18.71 ± 4.40	17.02 ± 3.95	12.10 ± 0.58	13.54 ± 0.50
	P@3	**21.56 ± 1.38**	15.57 ± 1.76	13.53 ± 2.33	10.19 ± 0.22	9.64 ± 0.25
	P@5	**21.27 ± 1.25**	16.14 ± 1.20	13.77 ± 1.87	10.20 ± 0.19	9.31 ± 0.14
	P@15	**27.52 ± 1.35**	21.47 ± 2.12	18.42 ± 1.16	13.02 ± 0.23	12.73 ± 0.22
2A Medical	MRR	**64.83 ± 3.32**	42.25 ± 2.01	46.54 ± 2.89	49.28 ± 1.60	41.40 ± 1.53
	MAP	**50.24 ± 2.29**	32.44 ± 2.72	34.19 ± 1.87	21.99 ± 0.73	18.75 ± 0.67
	P@1	**53.28 ± 3.86**	30.12 ± 2.44	35.26 ± 3.31	38.86 ± 1.65	30.62 ± 1.53
	P@3	**46.69 ± 3.52**	28.68 ± 2.82	32.63 ± 2.63	25.07 ± 1.09	19.83 ± 0.94
	P@5	**46.60 ± 2.91**	29.18 ± 2.73	32.02 ± 2.11	21.18 ± 0.92	17.30 ± 0.74
	P@15	**54.69 ± 1.74**	37.72 ± 3.40	37.35 ± 1.55	19.28 ± 0.63	17.81 ± 0.56
2B Music	MRR	**67.43 ± 2.37**	54.37 ± 3.06	60.47 ± 3.91	19.84 ± 0.98	19.54 ± 0.97
	MAP	**55.03 ± 1.98**	47.52 ± 2.75	48.38 ± 2.07	8.91 ± 0.49	8.42 ± 0.44
	P@1	**56.68 ± 2.98**	42.52 ± 3.45	48.34 ± 5.34	12.32 ± 0.65	12.42 ± 0.76
	P@3	**52.94 ± 2.05**	44.53 ± 2.51	45.29 ± 3.25	9.08 ± 0.61	8.64 ± 0.37
	P@5	**52.92 ± 2.37**	45.24 ± 2.92	46.00 ± 2.23	8.55 ± 0.47	7.72 ± 0.48
	P@15	**58.59 ± 2.05**	52.72 ± 2.90	52.81 ± 1.00	8.86 ± 0.62	8.54 ± 0.53

Bold indicates the best performance across the comparison models

Comparison results

Table 3 presents the results of the experiments on three subtask datasets. This table contains the MRR, MAP, and precision at ranks $k = \{1, 3, 5, 15\}$ (P@k) of the proposed and comparison methods. The average performance of the holdout cross-validation with the corresponding standard deviation is presented for each evaluation measure and method, and the best performance among the methods is represented in **bold**. As listed in Table 3, the proposed method outperforms all measures across subtasks.

The MRR indicates the ability of the related item to be ranked high, suggesting that the proposed method performs more accurately in identifying hypernyms than other methods. For example, in the results of the 1A dataset, the MRR value of the proposed method is 38.86. Compared to RMM, which uses word2vec embedding with projection learning, the average performance difference is 11.47. The MAP value of the proposed method is 24.17, the first rank, and the difference in average performance from RMM, which is the second rank, is 5.92. The MAP considers the precision of all related items. Therefore, the results indicate that the proposed method predicts more gold hypernyms regarding ranking problems than other methods. The results of P@k also support this. These results appear the same in all other subtask datasets.

Table 4 reveals the results of the Wilcoxon signed-rank test of the proposed method against the comparison methods for 10 iterations on the 1A dataset. The table confirms that the proposed method significantly outperforms other methods because all p-values are less than the significance level of $\alpha = 0.05$, rejecting the null hypothesis. For each evaluation measure, the winning method is remarked with **bold**, and p-values are

Table 4 Wilcoxon signed-rank test results of the proposed method against comparison methods for the 1A dataset with 10 iterations

Comparison methods	Evaluation measures	Proposed versus	
		(*p*-value)	
RMM [11]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
SPON [12]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
Prompting BERT (is a)[14]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
Prompting BERT (such as)[14]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)

At the significance level of $\alpha = 0.05$ (*p*-values in parentheses)

presented in the parenthesis. For the 2A and 2B datasets, the Wilcoxon signed-rank test result is the same as that for 1A, which can be observed in the Appendix 1.

In-depth analysis

This study introduces a further pretraining and fine–tuning process for hypernym discovery. Specifically, the pretraining phase uses MLM with extended Hearst patterns extracted from the given corpus, and the fine–tuning phase adopts projection learning with `Hypert`. To assess the influence of the choices, we examined several components of the proposed method. We discuss the effects of the proposed pretraining method and provide the results of the outcomes from each pretraining step. We also defined and evaluated two subgroups in the 1A subtask dataset to validate the robustness of `Hypert` against conventional methods. This study provides the pattern distribution with statistics and analyzes which patterns appeared frequently. We speculated that the proposed method could handle rare noun phrases. To support this, we present the prediction list of the proposed method and comparison methods. Additionally, the tSNE plots of the $e_{\texttt{[CLS]}}$ representation space are presented to analyze the effectiveness of using the `[CLS]` token as a hypernym relationship information vector.

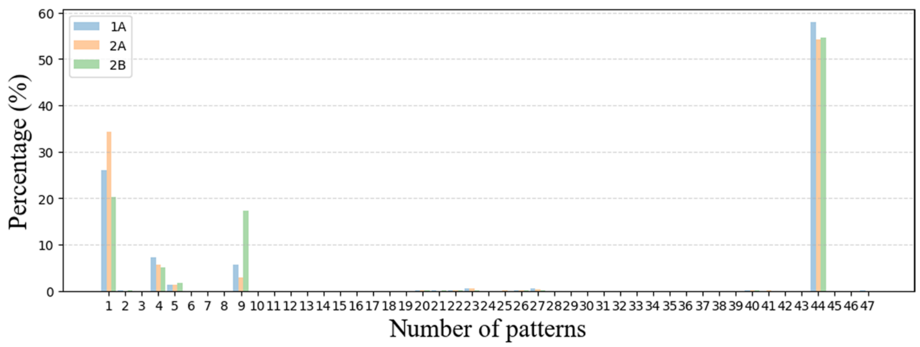


Fig. 4 Distribution of patterns for each subtask

Table 5 Top five patterns and counts

Pattern	Subtask	Count (%)	
((NP_\w+ ?(,)?)+(and or)?as NP_\w+)	1A	2,901,699	(57.87)
	2A	67,962	(54.21)
	2B	66,401	(54.64)
(NP_\w+ (,)?such as (NP_\w+ ?(,)?(and or)?)+)	1A	1,305,599	(26.04)
	2A	43,013	(34.31)
	2B	24,527	(20.18)
(NP_\w+ (,)?like (NP_\w+ ? (,)?(and or)?)+)	1A	285,906	(5.70)
	2A	3540	(2.82)
	2B	21,128	(17.38)
(NP_\w+ (,)?include (NP_\w+ ?(,)?(and or)?)+)	1A	359,496	(7.17)
	2A	7103	(5.66)
	2B	6188	(5.09)
(NP_\w+ (,)?especially (NP_\w+ ?(,)?(and or)?)+)	1A	65,892	(1.31)
	2A	1664	(1.32)
	2B	2006	(1.65)

Pattern proportions (%) are in parentheses

The pattern analysis was conducted to determine which patterns effectively construct a hypernym-related corpus. The pattern distribution for the extracted sentences is displayed in Fig. 4. Most sentences were extracted using almost five patterns. Table 5 presents the counts for the top five patterns. The "NP_y as NP_x" pattern was more than 50% for all subtasks. The "NP_y such as NP_x" pattern was more than 20%. These five patterns comprised over 98% of sentences.

Table 6 represents the results of using the Hypert and BERT models. The results indicate that the proposed pretraining method improves the performance of all subtask datasets across all evaluation measures. In addition, we also employed the Wilcoxon signed-rank test to confirm the superiority of the pretraining method. The results of the Wilcoxon signed-rank test are provided in Table 7. Most results reject the null hypothesis of the Wilcoxon signed-rank test with a significance level of $\alpha = 0.05$, except for the 2A dataset.

Because we chose 1k steps for all subtasks for fairness, we varied the pretraining from step 0k (without pretraining) to 10k to observe the performance of increasing

Table 6 Comparison of evaluation measures on the Hypert and BERT models for each subtask

Subtask	Evaluation measures	`Hypert` model	BERT model
1A English	MRR	**38.68 ± 2.00**	36.44 ± 2.12
	MAP	**24.17 ± 1.26**	23.29 ± 0.78
	P@1	**29.57 ± 1.98**	26.38 ± 2.93
	P@3	**21.56 ± 1.38**	20.90 ± 1.00
	P@5	**21.27 ± 1.25**	20.68 ± 0.79
	P@15	**27.52 ± 1.35**	26.63 ± 0.75
2A Medical	MRR	**64.83 ± 3.32**	62.62 ± 3.20
	MAP	**50.24 ± 2.29**	48.85 ± 1.57
	P@1	**53.28 ± 3.86**	49.94 ± 4.35
	P@3	**46.69 ± 3.52**	45.45 ± 2.27
	P@5	**46.60 ± 2.91**	45.66 ± 1.74
	P@15	**54.69 ± 1.74**	53.36 ± 0.97
2B Music	MRR	**67.43 ± 2.37**	63.19 ± 5.38
	MAP	**55.03 ± 1.98**	49.70 ± 3.37
	P@1	**56.68 ± 2.98**	50.92 ± 7.31
	P@3	**52.94 ± 2.05**	46.88 ± 4.28
	P@5	**52.92 ± 2.37**	47.27 ± 3.59
	P@15	**58.59 ± 2.05**	53.97 ± 2.48

Bold face indicates the best performance between two models

Table 7 Wilcoxon signed-rank test results for the Hypert and BERT models with 10 iterations

	Subtask	Evaluation measures	BERT model	
`Hypert` model versus (p-value)	1A	MRR	Win	(2.73e−2)
		MAP	Win	(6.40e−2)
		P@1	Win	(5.85e−3)
		P@3	Win	(8.39e−2)
		P@5	Win	(1.30e−1)
		P@15	Win	(4.88e−2)
	2A	MRR	Win	(1.93e−1)
		MAP	Win	(1.93e−1)
		P@1	Win	(1.38e−1)
		P@3	Win	(4.92e−1)
		P@5	Win	(4.92e−1)
		P@15	Win	(4.88e−2)
	2B	MRR	Win	(4.88e−2)
		MAP	Win	(9.76e−3)
		P@1	Win	(6.44e−2)
		P@3	Win	(3.90e−3)
		P@5	Win	(1.36e−2)
		P@15	Win	(3.90e−3)

At the significance level of $\alpha = 0.05$ (*p*-values in parentheses)

the steps. Tables 8 and 9 detail the performance of the proposed further pretraining method for each 1k step.

In the 1A dataset, the MRR value of pretraining with the 1k steps model is the best through all steps. Compared to the 0k steps model, which does not use further

Table 8 Comparison of MRR, MAP, and P@1 results by number of pretraining steps

Subtask	Steps	Evaluation measures		
		MRR	**MAP**	**P@1**
1A English	0k	36.44 ± 2.12	23.29 ± 0.78	26.38 ± 2.93
	1k	**38.68 ± 2.00**	24.17 ± 1.26	**29.57 ± 1.98**
	2k	37.52 ± 2.02	23.59 ± 0.98	28.37 ± 2.64
	3k	37.71 ± 1.08	23.76 ± 0.95	28.36 ± 1.39
	4k	37.49 ± 1.05	23.69 ± 0.86	28.03 ± 1.33
	5k	38.06 ± 2.04	**24.51 ± 0.73**	28.42 ± 2.53
	6k	38.46 ± 2.33	24.36 ± 1.18	29.13 ± 2.62
	7k	37.75 ± 1.49	23.89 ± 1.23	28.45 ± 1.52
	8k	37.69 ± 3.08	23.89 ± 1.63	28.46 ± 3.43
	9k	<u>38.67 ± 1.59</u>	<u>24.46 ± 0.89</u>	<u>29.38 ± 1.82</u>
	10k	37.85 ± 1.94	24.40 ± 0.80	27.87 ± 2.78
2A Medical	0k	62.62 ± 3.20	48.85 ± 1.57	49.94 ± 4.35
	1k	64.83 ± 3.32	50.24 ± 2.29	53.28 ± 3.86
	2k	**66.52 ± 2.44**	<u>50.48 ± 1.63</u>	55.64 ± 3.43
	3k	60.91 ± 4.65	47.41 ± 2.60	49.40 ± 5.70
	4k	65.10 ± 4.13	49.60 ± 2.50	54.62 ± 4.92
	5k	64.04 ± 3.20	49.25 ± 1.38	53.34 ± 4.37
	6k	66.21 ± 2.66	50.15 ± 2.23	<u>55.68 ± 2.69</u>
	7k	64.74 ± 3.23	49.62 ± 2.04	54.28 ± 3.84
	8k	<u>66.34 ± 3.81</u>	**50.52 ± 2.15**	**56.16 ± 4.95**
	9k	64.42 ± 2.52	50.15 ± 1.52	53.18 ± 4.00
	10k	63.44 ± 2.91	48.56 ± 1.34	52.62 ± 3.52
2B Music	0k	63.19 ± 5.38	49.70 ± 3.37	50.92 ± 7.31
	1k	**67.43 ± 2.37**	**55.03 ± 1.98**	**56.68 ± 2.98**
	2k	65.71 ± 3.04	53.97 ± 2.24	53.84 ± 4.03
	3k	62.93 ± 4.22	50.53 ± 3.75	50.42 ± 5.50
	4k	65.63 ± 4.65	53.55 ± 3.27	53.86 ± 5.92
	5k	66.89 ± 3.31	53.96 ± 2.33	55.62 ± 4.71
	6k	<u>67.40 ± 3.41</u>	<u>54.53 ± 3.23</u>	<u>56.42 ± 4.46</u>
	7k	66.09 ± 4.51	53.43 ± 3.02	54.84 ± 5.85
	8k	67.21 ± 3.47	54.28 ± 1.93	56.18 ± 4.74
	9k	65.25 ± 6.19	52.37 ± 4.57	53.48 ± 8.48
	10k	64.54 ± 5.24	51.69 ± 3.77	52.92 ± 6.93

0k indicates the BERT model. **Bold** represents the best performance across the comparison models, and <u>underline</u> represents second place

pretraining, the average performance difference is 2.24. Moreover, the results indicate that the proposed further pretraining method improves performance compared to the 0k steps model at every 1k step. The rational choice for each subtask will now be discussed. For the 1A dataset, the model with 9k pretraining steps seems reasonable considering average performance and standard deviation. For the 2A dataset, despite the 8k steps showing most of the best performance, we consider the 2k steps to be selected as the best choice because of the low standard deviation and the negligible difference in performance between them. However, the results for the 2B dataset clearly provide reasonable pretraining steps, 1k. The second is also evident, 6k. Note that we merely selected the model with the 1k steps through all subtask datasets for

Table 9 Comparison of for P@3, P@5, and P@15 results by number of pretraining steps

Subtask	Steps	Evaluation measures		
		P@3	P@5	P@15
1A English	0k	20.90 ± 1.00	20.68 ± 0.79	26.63 ± 0.75
	1k	21.56 ± 1.38	21.27 ± 1.25	27.52 ± 1.35
	2k	21.01 ± 1.05	20.83 ± 0.84	26.79 ± 1.16
	3k	21.25 ± 0.99	21.04 ± 0.93	26.95 ± 1.11
	4k	21.19 ± 0.94	20.88 ± 0.87	26.89 ± 0.95
	5k	21.81 ± 1.25	21.61 ± 0.87	**27.96 ± 0.57**
	6k	<u>21.93 ± 1.63</u>	21.58 ± 1.18	27.52 ± 0.96
	7k	21.48 ± 1.16	21.12 ± 1.11	27.12 ± 1.42
	8k	21.32 ± 2.18	21.13 ± 1.68	27.17 ± 1.10
	9k	21.91 ± 1.08	**21.73 ± 0.80**	27.62 ± 0.91
	10k	**21.93 ± 1.18**	<u>21.64 ± 0.82</u>	<u>27.80 ± 0.70</u>
2A Medical	0k	45.45 ± 2.27	45.66 ± 1.74	53.36 ± 0.97
	1k	46.69 ± 3.52	**46.60 ± 2.91**	54.69 ± 1.74
	2k	46.97 ± 1.56	46.22 ± 1.86	54.80 ± 1.69
	3k	43.50 ± 2.89	43.47 ± 2.68	52.46 ± 2.42
	4k	45.80 ± 2.97	45.21 ± 2.48	53.98 ± 2.00
	5k	45.32 ± 2.04	44.83 ± 1.46	53.98 ± 1.10
	6k	**46.98 ± 2.46**	46.05 ± 2.35	54.34 ± 2.12
	7k	45.94 ± 3.08	45.36 ± 2.57	54.42 ± 1.27
	8k	<u>46.98 ± 2.65</u>	<u>46.28 ± 2.26</u>	<u>54.82 ± 1.83</u>
	9k	46.15 ± 1.78	45.85 ± 2.06	**55.01 ± 1.06**
	10k	44.90 ± 2.34	43.92 ± 1.72	53.40 ± 1.46
2B Music	0k	46.88 ± 4.28	47.27 ± 3.59	53.97 ± 2.48
	1k	**52.94 ± 2.05**	**52.92 ± 2.37**	**58.59 ± 2.05**
	2k	51.13 ± 2.99	51.91 ± 2.39	58.37 ± 1.79
	3k	47.29 ± 4.56	47.95 ± 4.17	55.36 ± 2.78
	4k	50.98 ± 3.89	51.67 ± 3.55	57.64 ± 2.48
	5k	51.45 ± 2.99	51.69 ± 2.47	58.25 ± 1.51
	6k	<u>52.24 ± 4.52</u>	<u>52.25 ± 3.94</u>	<u>58.49 ± 2.41</u>
	7k	51.28 ± 4.33	51.27 ± 3.25	57.46 ± 1.96
	8k	51.76 ± 3.20	52.17 ± 2.01	58.20 ± 1.29
	9k	49.46 ± 6.50	49.79 ± 5.01	56.87 ± 3.42
	10k	48.86 ± 5.33	49.52 ± 4.61	56.04 ± 2.52

0k indicates the BERT model. **Bold** represents the best performance across the comparison models, and <u>underline</u> represents second place

fairness. A comparison of the pretraining steps considered to be the best performance can be found in Appendix 2.

In the 1A dataset, the MRR value of pretraining with the 1k-step model is the best through all steps. Compared to the 0k-step model, which does not use further pretraining, the average performance difference is 2.24. Moreover, the results indicate that the proposed further pretraining method improves performance compared to the 0k-step model at every 1k step. For the 1A dataset, the model with 9k pretraining steps seems reasonable, considering the average performance and standard deviation. For the 2A dataset, despite the 8k steps showing most of the best performance, the 2k steps are the best choice because of the low standard deviation and negligible difference in performance between them. However, the 2B dataset results provide reasonable pretraining

Table 10 Model performance on two subgroups in the 1A dataset

Subgroup	Evaluation measures	Proposed method	RMM	SPON	Prompting BERT (is a)	Prompting BERT (such as)
Person	MRR	**84.78 ± 5.48**	60.88 ± 17.54	66.83 ± 16.51	21.16 ± 1.22	17.08 ± 1.00
	MAP	**45.78 ± 1.50**	38.37 ± 5.31	38.82 ± 5.58	13.68 ± 0.47	11.01 ± 0.68
	P@1	**79.34 ± 7.94**	49.51 ± 21.90	55.01 ± 19.68	10.32 ± 1.23	8.50 ± 1.09
	P@3	**44.82 ± 1.72**	34.92 ± 8.61	36.53 ± 8.61	10.38 ± 0.58	8.24 ± 0.75
	P@5	**40.88 ± 1.36**	34.45 ± 5.77	34.92 ± 5.83	11.88 ± 0.42	8.60 ± 0.71
	P@15	**45.04 ± 1.54**	41.16 ± 2.79	40.33 ± 2.77	17.02 ± 0.58	15.34 ± 0.73
Computer- Software	MRR	**36.69 ± 4.71**	24.37 ± 3.49	17.01 ± 3.96	17.25 ± 3.19	19.52 ± 2.85
	MAP	**21.75 ± 2.13**	14.58 ± 3.01	8.36 ± 2.23	7.01 ± 1.46	8.31 ± 1.14
	P@1	**22.53 ± 5.64**	13.03 ± 3.13	7.94 ± 3.28	11.12 ± 2.35	11.59 ± 2.64
	P@3	**19.44 ± 2.71**	12.58 ± 2.72	6.89 ± 2.44	7.22 ± 1.39	8.68 ± 1.32
	P@5	**20.09 ± 2.31**	13.38 ± 3.29	6.89 ± 2.01	6.98 ± 1.44	7.97 ± 1.20
	P@15	**25.55 ± 2.15**	17.83 ± 4.44	10.91 ± 2.57	6.48 ± 1.42	7.97 ± 1.21

Bold indicates the best performance across the comparison models

steps, 1k. The second is also evident at 6k. We selected the model with 1k steps through all subtask datasets for fairness. A comparison of the pretraining steps considered to have the best performance is provided in Appendix 2.

We defined and evaluated two subgroups to assess the robustness of performance for queries that can be grouped within the 1A dataset. One is the person group, and the other is the computer-software group. The person group consists of the query word if “person” exists in the gold hypernyms of the test set. The computer-software group consists of queries that correspond when “computer” or “software” exists in the test set gold hypernyms. On average, the person group had 320 queries, and the computer-software group had 61 queries. We evaluated the proposed method and comparison methods for the two subgroups. The results are shown in Table 10.

For the person group, the proposed method showed the best performance among the comparison methods for all evaluation measures. The MRR value of the proposed method is 84.78, which is significantly higher than other methods. For the computer-software group, the proposed method also outperforms other methods. Thus, the proposed method consistently outperformed compared methods in the experiment of two subgroups which is a similar result observed from the experiment of the original 1A dataset. In detail, the performance of the person group is substantially higher than the computer-software group. The reason for this result may be found from the characteristics of the person group, that most of the gold hypernym “person” appears first in the gold hypernyms, and the number of gold hypernyms is small. In contrast, the gold hypernyms of the computer-software group are much more varying compared to that of the person group where most gold hypernyms are multi-words, such as (“Xpdf”, “code, computer software, software package,...”), indicating that the hypernym relation is much more difficult to predict.

To assess the robustness of the rare words, we compared the predictions of the proposed method with word2vec-based methods for the rare words. The test query “*open proxy server*,” which appears nine times in the given corpus, was used for analysis. The

Table 11 Prediction results for each method for the rare noun phrase "*open proxy server*" (appearing nine times in the corpus)

Model	Predicted
Gold	**Service***, **Software***, **Computer program***, **Software program***, **Software application***, **Proxy server***, **Application***, **Software package***, **Application software***
Proposed	**Aervice***, **Aoftware***, <u>server</u>, **Computer program***, facility, protocol, **Software program***, platform, company, client, **Software application***, host, **Application***, <u>Computer system</u>, <u>application program</u>
RMM	Person, pseudonymized, granularly, <u>open proxy</u>, ad filtering, intrabank, Otherwise, spoofing attack, actual, statefully, sdl plc, authenticatable, Ip address spoofing, example, solely
SPON	Person, constructed structure, technical specification, town, Single-valued function, city, juridical person, leader, animal, state, boss, Company, **computer program***, measure, political leader

Bold with the * symbol indicates gold hypernyms, and <u>underlines</u> mark hypernymy-relevant words

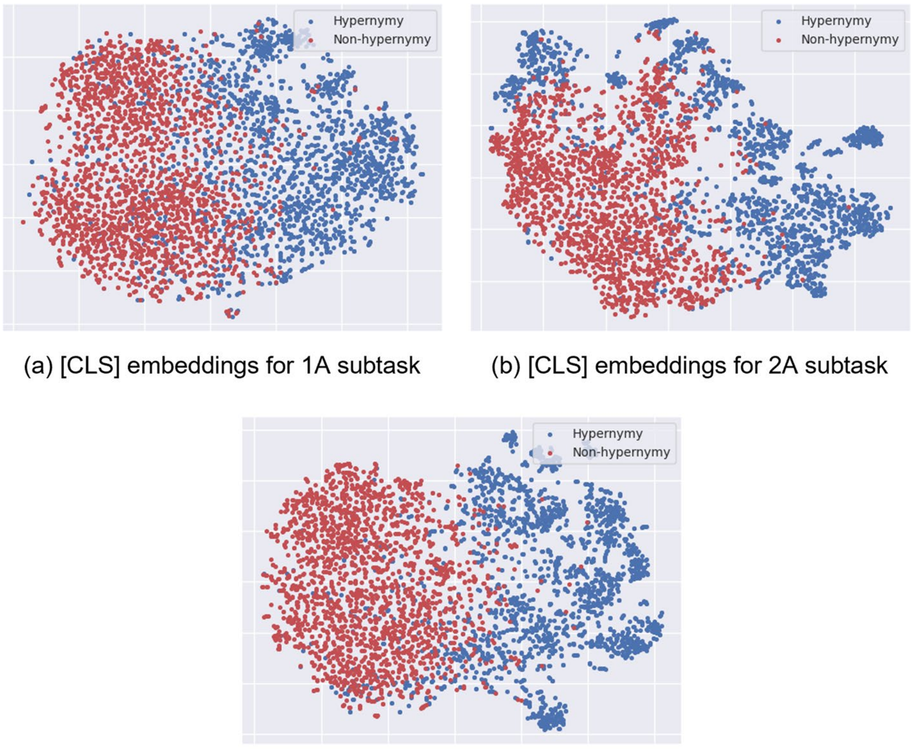
Table 12 Prediction results for each method for the rare noun phrase "*tempestuousness*" (appearing once in the corpus)

Model	Predicted
Gold	**Turbulence***, **State***, **Disorder***, **Weather condition***, **Weather***, **Atmospheric phenomenon***, **Physical phenomenon***, **Natural phenomenon***, **Phenomenon***
Proposed	**Phenomenon***, dislike, <u>emotional state</u>, **Physical phenomenon***, **Atmospheric phenomenon***, experience, circumstance, **State***, motion, **Natural phenomenon***, **Weather condition***, sensation, badness Convergence, binary relation
RMM	Person, others, personality, another, upon, oneself, life, necessarily, relate, Solely, become, intellectual, must, one, personal
SPON	Specifications, constructed structure, town, city, technical specification, Movement, locale, person, **State***, single-valued function, **Phenomenon***, instrument, **Physical phenomenon***, animal, form

Bold with the * symbol indicates gold hypernyms, and <u>underlines</u> mark hypernymy-relevant words

prediction lists of each method and the gold hypernyms are presented in Table 11. The gold hypernyms are represented in **bold** with the * symbol. Because the dataset is handcrafted, there may be more hypernyms. Thus, using <u>underlines</u>, we annotated relevant words on hypernym relationships using our judgment. Each prediction list was produced by sorting the probabilities of modes and taking the top 15 candidates. Hence, the earlier a word appears in the list, the more likely it is to be a hypernym.

The results reveal that the proposed method adequately predicts rare words. In addition, most prediction words, including relevant words, are predicted better than the others. Conversely, RMM and SPON, which are word2vec-based methods, perform poorly on low-frequency words. For example, SPON only corrects for one gold hypernym ranked low on the list. Except for "***computer program****," SPON predicted the wrong words. Although RMM did not correctly predict any gold hl relevant words were present in its prediction list, such as "*pseudonymized*," "*spoofing attack*," and "*IP address spoofing*," but none of them were hypernymy. Table 12 lists the prediction list for "*tempestuousness*," which appeared once in the corpus. The result also suggests that the proposed method contains more gold hypernyms than the others for rare words.



(a) [CLS] embeddings for 1A subtask

(b) [CLS] embeddings for 2A subtask

(c) [CLS] embeddings for 2B subtask

Fig. 5 tSNE plots of the `[CLS]` token embedding representation for subtasks (**a**) 1A, (**b**) 2A, and (**c**) 2B. Blue indicates hypernymy, and red indicates non-hypernymy (please see color version)

In addition, we explored the quality of the `[CLS]` token embeddings. We randomly selected hypernym pairs for each subtask from the testing set. The positive S_P were created from the selected pairs. The negative S_P were also generated by replacing a gold hypernym with a random candidate hypernym that is not gold. Then, each S_P was input into the proposed model to obtain $e_{[\mathrm{CLS}]}$ for each S_P. Figure 5 depicts the tSNE plots of the $e_{[\mathrm{CLS}]}$ representation space for each subtask. Blue indicates the $e_{[\mathrm{CLS}]}$ of positive S_P and red indicates the $e_{[\mathrm{CLS}]}$ of negative S_P. The hypernymy and nonhypernymy clusters are appropriately separated in all three plots, revealing that using the `[CLS]` token as a hypernym relationship information vector effectively identifies hypernym relationships.

Conclusions

Hypernym discovery is challenging because it finds appropriate hypernyms from a large predefined pool of candidates for a given query. In addition, because the candidates contain noun phrases, conventional word2vec-based methods are challenging to handle. In addition, BERT can solve this problem using subword tokenization. However, there have been no attempts to use BERT in hypernym discovery with its widely used training steps of domain adaptation: pretraining and fine-tuning. Therefore, this study presents the following procedures for adapting BERT to the domain tasks by modifying the pretraining and fine-tuning stages.

We proposed MLM with Hearst pattern sentences as a further pretraining procedure to adapt the hypernymy domain. The proposed method outperformed the comparison methods on all evaluation measures and subtask datasets. The Wilcoxon signed-rank test was employed to confirm the superiority of the proposed method. We also conducted an in-depth analysis to confirm the effectiveness of the proposed pretraining procedure, analyzed the distribution of utilized Hearst patterns, and presented effective patterns. The proposed pretraining performs better than BERT without the proposed pretraining stage. In addition, we demonstrated that the proposed method is robust against rare words compared to the comparison models in the case study and can produce stable performance in the viewpoint of subgroups. The results of the case study indicate the robustness of the proposed method for rare words compared to the existing methods. Furthermore, the tSNE plots were presented to demonstrate the representation space of the special prompt component.

Despite the effectiveness of `Hypert`, the computational cost of `Hypert` for inferencing hypernym relationships can be heavier than conventional methods such as Hearst pattern matching. Thus, when a large number of queries and candidates, for example, 200,000 candidates for one query in this study, is considered, `Hypert` can be slower than conventional methods. For example, the proposed `Hypert` expenses 15.52 queries per second (q/s), whereas its counterparts RMM, SPON, and prompting BERT consume 0.02, 0.02, and 0.46 q/s, respectively. In addition, the performance of `Hypert` may still be limited because we employed a general tokenizer instead of developing a domain-specific tokenizer for each general, medical, and music domain. Furthermore, a pretraining process may be required if `Hypert` is applied to a new domain, such as cyber security because the proposed method is based on a general language model.

In the future, we would like to construct additional benchmark datasets for hypernym discovery because most studies in hypernym discovery tasks reported that no additional benchmark datasets are available so far except SemEval2018 Task9-Hypernym Discovery dataset [11, 44]. Specifically, we would like to start our effort to create new datasets for Cyber Threat Intelligence (CTI) from the cyber security domain to evaluate the efficacy of `Hypert` from cybersecurity-oriented documents. In the field of CTI, extracting cyber threat insights from diverse data sources spanning multiple domains, including the Web, is an essential task. In addition, cyber threat information is predominantly communicated through written language in diverse CTI reports involving hypernymic relationships. For example, cyber security practitioners may seek to scrutinize CTI reports containing references to specific malware instances. In this context, hypernym discovery can be used to determine the category of a particular malware, and the proposed `Hypert` can be applied here. We would like to study this issue further.

Appendix 1. Statistical test details

See Tables 13, 14, 15, 16 and 17.

Table 13 Wilcoxon signed-rank test results for the proposed method against the comparison models for the 2A dataset with 10 iterations

Comparison methods	Evaluation measures	Proposed versus (*p*-value)	
RMM [11]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
SPON [12]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
Prompting BERT (is a) [14]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
Prompting BERT (such as) [14]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)

At the significance level of $\alpha = 0.05$ (p-values parentheses)

Tables 13 and 14 are the results of the Wilcoxon signed-rank test for the 2A and 2B datasets against comparison models. All p-values in the result are 1.95e–3, rejecting the null hypothesis.

Appendix 2. Results of the best hypert model for each subtask

Table 15 compares the proposed method with the best Hypert and BERT models. The pretraining steps for the 1A, 2A, and 2B datasets are 9k, 2k, and 1k, respectively. The results of the Wilcoxon signed-rank test for all subtask datasets are provided in Table 16. Except for a few P@k measures, most of the p-values reject the null hypothesis.

Appendix 3. Extended hearst patterns

Table 17 lists the regular expressions of the extended hypernym syntactic patterns used in this study, where NP indicates a noun phrase, and PRON represents a pronoun.[7]

[7] https://github.com/abyssnlp/Hearst-Hypernym-Extractor.

Table 14 Wilcoxon signed-rank test results of the proposed method against comparison models for the 2B dataset with 10 iterations

Comparison methods	Evaluation measures	Proposed versus (*p*-value)	
RMM [11]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
SPON [12]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
Prompting BERT (is a) [14]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)
Prompting BERT (such as) [14]	MRR	Win	(1.95e–3)
	MAP	Win	(1.95e–3)
	P@1	Win	(1.95e–3)
	P@3	Win	(1.95e–3)
	P@5	Win	(1.95e–3)
	P@15	Win	(1.95e–3)

At the significance level of $\alpha = 0.05$ (p-values parentheses)

Table 15 Comparison of evaluation measures on the best Hypert and BERT models for each subtask

Subtask	Evaluation measures	Hypert model	BERT model
1A English	MRR	**38.67 ± 1.59**	36.44 ± 2.12
	MAP	**24.46 ± 0.89**	23.29 ± 0.78
	P@1	**29.38 ± 1.82**	26.38 ± 2.93
	P@3	**21.91 ± 1.08**	20.90 ± 1.00
	P@5	**21.73 ± 0.80**	20.68 ± 0.79
	P@15	**27.80 ± 0.91**	26.63 ± 0.75
2A Medical	MRR	**66.52 ± 2.44**	62.62 ± 3.20
	MAP	**50.48 ± 1.63**	48.85 ± 1.57
	P@1	**55.64 ± 3.43**	49.94 ± 4.35
	P@3	**46.97 ± 1.56**	45.45 ± 2.27
	P@5	**46.22 ± 1.86**	45.66 ± 1.74
	P@15	**54.80 ± 1.69**	53.36 ± 0.97
2B Music	MRR	**67.43 ± 2.37**	63.19 ± 5.38
	MAP	**55.03 ± 1.98**	49.70 ± 3.37
	P@1	**56.68 ± 2.98**	50.92 ± 7.31
	P@3	**52.94 ± 2.05**	46.88 ± 4.28
	P@5	**52.92 ± 2.37**	47.27 ± 3.59
	P@15	**58.59 ± 2.05**	53.97 ± 2.48

Bold face indicates the best performance between two models

Table 16 Wilcoxon signed-rank test results for the Hypert and BERT models with 10 iterations

	Subtask	Evaluation measures	BERT model	
Hypert model versus (*p*-value)	1A	MRR	Win	(9.76e−3)
		MAP	Win	(3.90e−3)
		P@1	Win	(9.76e−3)
		P@3	Win	(2.73e−2)
		P@5	Win	(3.90e−3)
		P@15	Win	(3.71e−2)
	2A	MRR	Win	(4.88e−2)
		MAP	Win	(2.73e−2)
		P@1	Win	(4.88e−2)
		P@3		(1.30e−1)
		P@5		(7.67e−1)
		P@15	Win	(4.88e−2)
	2B	MRR	Win	(4.88e−2)
		MAP	Win	(9.76e−3)
		P@1		(6.44e−2)
		P@3	Win	(3.9e−3)
		P@5	Win	(1.36e−2)
		P@15	Win	(3.9e−3)

At the significance level of $\alpha = 0.05$ (*p*-values parentheses)

Table 17 Regular expressions of extended Hearst patterns used in this study

No.	Pattern
1	(NP_\w+ (,)?such as (NP_\w+ ?(,)?(and or)?)+)
2	(such NP_\w+ (,)?as (NP_\w+ ?(,)?(and or)?)+)
3	((NP_\w+ ?(,)?)+(and or)?other NP_\w+)
4	(NP_\w+ (,)?include (NP_\w+ ?(,)?(and or)?)+)
5	(NP_\w+ (,)?especially (NP_\w+ ?(,)?(and or)?)+)
6	((NP_\w+ ?(,)?)+(and or)?any other NP_\w+)
7	((NP_\w+ ?(,)?)+(and or)?some other NP_\w+)
8	((NP_\w+ ?(,)?)+(and or)?be a NP_\w+)
9	(NP_\w+ (,)?like (NP_\w+ ? (,)?(and or)?)+)
10	such (NP_\w+ (,)?as (NP_\w+ ? (,)?(and or)?)+)
11	((NP_\w+ ?(,)?)+(and or)?like other NP_\w+)
12	((NP_\w+ ?(,)?)+(and or)?one of the NP_\w+)
13	((NP_\w+ ?(,)?)+(and or)?one of these NP_\w+)
14	((NP_\w+ ?(,)?)+(and or)?one of those NP_\w+)
15	example of (NP_\w+ (,)?be (NP_\w+ ? (,)?(and or)?)+)
16	((NP_\w+ ?(,)?)+(and or)?be example of NP_\w+)
17	(NP_\w+ (,)?for example (,)?(NP_\w+ ?(,)?(and or)?)+)
18	((NP_\w+ ?(,)?)+(and or)?which be call NP_\w+)
19	((NP_\w+ ?(,)?)+(and or)?which be name NP_\w+)
20	(NP_\w+ (,)?mainly (NP_\w+ ? (,)?(and or)?)+)
21	(NP_\w+ (,)?mostly (NP_\w+ ? (,)?(and or)?)+)
22	(NP_\w+ (,)?notably (NP_\w+ ? (,)?(and or)?)+)
23	(NP_\w+ (,)?particularly (NP_\w+ ? (,)?(and or)?)+)
24	(NP_\w+ (,)?principally (NP_\w+ ? (,)?(and or)?)+)
25	(NP_\w+ (,)?in particular (NP_\w+ ? (,)?(and or)?)+)
26	(NP_\w+ (,)?except (NP_\w+ ? (,)?(and or)?)+)
27	(NP_\w+ (,)?other than (NP_\w+ ? (,)?(and or)?)+)
28	(NP_\w+ (,)?e.g. (,)?(NP_\w+ ? (,)?(and or)?)+)
29	(NP_\w+ \((e.g. i.e.) (,)?(NP_\w+ ? (,)?(and or)?)+(\.)?\))
30	(NP_\w+ (,)?i.e. (,)?(NP_\w+ ? (,)?(and or)?)+)
31	((NP_\w+ ?(,)?)+(and or)? a kind of NP_\w+)
32	((NP_\w+ ?(,)?)+(and or)? kind of NP_\w+)
33	((NP_\w+ ?(,)?)+(and or)? form of NP_\w+)
34	((NP_\w+ ?(,)?)+(and or)?which look like NP_\w+)
35	((NP_\w+ ?(,)?)+(and or)?which sound like NP_\w+)
36	(NP_\w+ (,)?which be similar to (NP_\w+ ? (,)?(and or)?)+)
37	(NP_\w+ (,)?example of this be (NP_\w+ ? (,)?(and or)?)+)
38	(NP_\w+ (,)?type (NP_\w+ ? (,)?(and or)?)+)
39	((NP_\w+ ?(,)?)+(and or)? NP_\w+ type)
40	(NP_\w+ (,)?whether (NP_\w+ ? (,)?(and or)?)+)
41	(compare (NP_\w+ ?(,)?)+(and or)?with NP_\w+)
42	(NP_\w+ (,)?compare to (NP_\w+ ? (,)?(and or)?)+)
43	(NP_\w+ (,)?among -PRON- (NP_\w+ ? (,)?(and or)?)+)
44	((NP_\w+ ?(,)?)+(and or)?as NP_\w+)
45	(NP_\w+ (,)? (NP_\w+ ? (,)?(and or)?)+ for instance)
46	((NP_\w+ ?(,)?)+(and or)? sort of NP_\w+)
47	(NP_\w+ (,)?which may include (NP_\w+ ?(,)?(and or)?)+)

Acknowledgements
This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (2021-0-01341-003, Artificial Intelligence Graduate School Program (Chung-Ang University)).

Author contributions
GY: Conceptualization, Methodology, Formal analysis and investigation, Writing—original draft preparation, and Writing - review and editing. YL: Writing—review and editing. AM: Writing—original draft preparation. JL: Conceptualization, Methodology, Writing—review and editing, Funding acquisition, Resources, and Supervision. All authors have read and approved the final manuscript.

Funding
This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (2021-0-01341-003, Artificial Intelligence Graduate School Program (Chung-Ang University) and 2021-0-00766, Development of Integrated Development Framework that supports Automatic Neural Network Generation and Deployment optimized for Runtime Environment.).

Availability of data and materials
The datasets generated and/or analysed during the current study are available in the https://github.com/Gun1Yun/Hypert repository and https://competitions.codalab.org/competitions/17119.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 6 June 2023 Accepted: 30 August 2023
Published online: 12 September 2023

References
1. Hearst MA. Automatic acquisition of hyponyms from large text corpora. In: COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics; 1992
2. Yamane J, Takatani T, Yamada H, Miwa M, Sasaki Y. Distributional hypernym generation by jointly learning clusters and projections. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, The COLING 2016 Organizing Committee, Osaka, Japan; 2016, pp. 2016:1871–1879. https://aclanthology.org/C16-1176
3. Roller S, Erk K. Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Austin, Texas, 2016, pp. 2016:2163–2172. https://doi.org/10.18653/v1/D16-1234.
4. Caraballo SA. Automatic construction of a hypernym-labeled noun hierarchy from text. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, College Park, Maryland, USA; pp. 1999:120–126. https://doi.org/10.3115/1034678.1034705.
5. Cederberg S, Widdows D. Using LSA and noun coordination information to improve the recall and precision of automatic hyponymy extraction. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, pp. 111–118
6. Sabirova K, Lukanin A. Automatic extraction of hypernyms and hyponyms from russian texts. In: AIST (supplement), Citeseer 2014; pp. 35–40.
7. Sheena N, Jasmine SM, Joseph S. Automatic extraction of hypernym & meronym relations in English sentences using dependency parser. Procedia Comput Sci. 2016;93:539–46.
8. Camacho-Collados J, Delli Bovi C, Espinosa-Anke L, Oramas S, Pasini T, Santus E, Shwartz V, Navigli R, Saggion H. SemEval-2018 task 9: Hypernym discovery. In: Proceedings of The 12th International Workshop on Semantic Evaluation, Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 712–724. https://doi.org/10.18653/v1/S18-1115.
9. Fu R, Guo J, Qin B, Che W, Wang H, Liu T. Learning semantic hierarchies via word embeddings. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2014, pp. 1199–1209
10. Bernier-Colborne G, Barrière C. CRIM at SemEval-2018 task 9: a hybrid approach to hypernym discovery. In: Proceedings of The 12th International Workshop on Semantic Evaluation, Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 725–731. https://doi.org/10.18653/v1/S18-1116.

11. Bai Y, Zhang R, Kong F, Chen J, Mao Y. Hypernym discovery via a recurrent mapping model. In: Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, 2021, pp. 2912–2921
12. Dash S, Chowdhury MFM, Gliozzo A, Mihindukulasooriya N, Fauceglia NR. Hypernym detection using strict partial order networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2020;34, pp. 7626–7633
13. Peng B, Chersoni E, Hsu Y-Y, Huang C-R. Discovering financial hypernyms by prompting masked language models. In: Proceedings of the 4th Financial Narrative Processing Workshop@ LREC2022, 2022, pp. 10–16
14. Hanna M, Mareček D. Analyzing bert's knowledge of hypernymy via prompting. In: Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, 2021, pp. 275–282
15. Ravichander A, Hovy E, Suleman K, Trischler A, Cheung JCK. On the systematicity of probing contextualized word representations: the case of hypernymy in BERT. In: Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics, Association for Computational Linguistics, Barcelona, Spain (Online), 2020, pp. 88–102. https://aclanthology.org/2020.starsem-1.10
16. Ettinger A. What bert is not: lessons from a new suite of psycholinguistic diagnostics for language models. Trans Assoc Comput Linguist. 2020;8:34–48.
17. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. Adv Neural Inf Process Syst. 2013. https://doi.org/10.48550/arXiv.1310.4546.
18. Song X, Salcianu A, Song Y, Dopson D, Zhou D. Fast wordpiece tokenization. arXiv preprint. 2020 http://arxiv.org/abs/2012.15524
19. Araci D. Finbert: financial sentiment analysis with pre-trained language models. arXiv preprint. 2019. http://arxiv.org/abs/1908.10063
20. Lee J, Yoon W, Kim S, Kim D, Kim S, So CH, Kang J. Biobert: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics. 2020;36(4):1234–40.
21. Roller S, Kiela D, Nickel M. Hearst patterns revisited: Automatic hypernym detection from large text corpora. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, Melbourne, Australia,2 018, pp. 358–363. https://doi.org/10.18653/v1/P18-2057.
22. Devlin J, Chang M-W, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. https://doi.org/10.18653/v1/N19-1423.
23. Weeds J, Weir D. A general framework for distributional similarity. In: Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, 2003, pp. 81–88
24. Cimiano P, Hotho A, Staab S. Learning concept hierarchies from text corpora using formal concept analysis. J Artif Intell Res. 2005;24:305–39.
25. Geffet M, Dagan I. The distributional inclusion hypotheses and lexical entailment. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), 2005, pp. 107–114
26. Weeds J, Weir D, McCarthy D. Characterising measures of lexical distributional similarity. In: COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics, 2004, pp. 1015–1021
27. Lenci A, Benotto G. Identifying hypernyms in distributional semantic spaces. In: * SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012), 2012, pp. 75–79
28. Clarke D. Context-theoretic semantics for natural language: an overview. In: Proceedings of the Workshop on Geometrical Models of Natural Language Semantics, 2009, pp. 112–119
29. Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv preprint. 2014 http://arxiv.org/abs/1409.0473
30. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778
31. Kamel M, Trojahn C, Ghamnia A, Aussenac-Gilles N, Fabre C. A distant learning approach for extracting hypernym relations from wikipedia disambiguation pages. Procedia Computer Science, Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 21st International Conference, KES-20176-8 September 2017, Marseille, France, vol. 112 2017, pp. 1764–1773 https://doi.org/10.1016/j.procs.2017.08.208.
32. Navigli R, Ponzetto SP. Babelnet: the automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artif Intell. 2012;193:217–50. https://doi.org/10.1016/j.artint.2012.07.001.
33. Espinosa-Anke L, Camacho-Collados J, Delli Bovi C, Saggion H. Supervised distributional hypernym discovery via domain adaptation. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Austin, Texas, 2016, pp. 424–435. https://doi.org/10.18653/v1/D16-1041.
34. Iacobacci I, Pilehvar MT, Navigli R. SensEmbed: Learning sense embeddings for word and relational similarity. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, Beijing, China, 2015, pp. 95–105. https://doi.org/10.3115/v1/P15-1010.
35. Shwartz V, Goldberg Y, Dagan I. Improving hypernymy detection with an integrated path-based and distributional method. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Berlin, Germany, 2016, pp. 2389–2398. https://doi.org/10.18653/v1/P16-1226.
36. Yu C, Han J, Wang P, Song Y, Zhang H, Ng W, Shi S. When hearst is not enough: Improving hypernymy detection from corpus with distributional models. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 6208–6217. https://doi.org/10.18653/v1/2020.emnlp-main.502.
37. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. Adv Neural Inf Process Syst. 2017. https://doi.org/10.48550/arXiv.1706.03762.

38. Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V. Roberta: a robustly optimized bert pretraining approach. arXiv preprint. 2019. http://arxiv.org/abs/1907.11692
39. Lan Z, Chen M, Goodman S, Gimpel K, Sharma P, Soricut R. Albert: a lite bert for self-supervised learning of language representations. arXiv preprint. 2019. http://arxiv.org/abs/1909.11942
40. Sanh V, Debut L, Chaumond J, Wolf T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint. 2019. http://arxiv.org/abs/1910.01108
41. Talmor A, Elazar Y, Goldberg Y, Berant J. Olmpics-on what language model pre-training captures. Trans Assoc Comput Linguist. 2020;8:743–58.
42. Jiang Z, Xu FF, Araki J, Neubig G. How can we know what language models know? Trans Assoc Comput Linguist. 2020;8:423–38.
43. Petroni F, Rocktäschel T, Riedel S, Lewis P, Bakhtin A, Wu Y, Miller A. Language models as knowledge bases? In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), for Computational Linguistics, Hong Kong, China, 2019, pp. 2463–2473. Association https://doi.org/10.18653/v1/D19-1250.
44. Held W, Habash N. The effectiveness of simple hybrid systems for hypernym discovery. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 3362–3367. https://doi.org/10.18653/v1/P19-1327.
45. Jin Y, Jang E, Cui J, Chung J-W, Lee Y, Shin S. DarkBERT: A language model for the dark side of the Internet. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Toronto, Canada, 2023, pp. 7515–7533. https://aclanthology.org/2023.acl-long.415
46. Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, Cistac P, Rault T, Louf R, Funtowicz M, et al.: Huggingface's transformers: State-of-the-art natural language processing. arXiv preprint. 2019. http://arxiv.org/abs/1910.03771
47. Loshchilov I, Hutter F. Decoupled weight decay regularization. arXiv preprint. 2017. http://arxiv.org/abs/1711.05101
48. Wu J-C, Chang Y-C, Mitamura T, Chang JS. Automatic collocation suggestion in academic writing. In: Proceedings of the ACL 2010 Conference Short Papers, 2010, pp. 115–119
49. Rodríguez-Fernández S, Anke LE, Carlini R, Wanner L. Semantics-driven recognition of collocations using word embeddings. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2016, pp. 499–505
50. Wilcoxon F. In: Kotz S, Johnson NL. (eds.) Individual comparisons by ranking methods, New York: Springer, 1992, pp. 196–202. https://doi.org/10.1007/978-1-4612-4380-9_16.

Publisher's Note