TaxoEnrich: Self-Supervised Taxonomy Completion via Structure-Semantic Representations

Minhao Jiang[1], Xiangchen Song[2], Jieyu Zhang[3], Jiawei Han[1]
[1]Department of Computer Science, University of Illinois Urbana Champaign, IL, USA
[2]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
[3]The Paul G. Allen School of Computer Science & Engineering, University of Washington, WA, USA
{minhaoj2,hanj}@illinois.edu,xiangchs@cs.cmu.edu,jieyuz2@cs.washington.edu

ABSTRACT

Taxonomies are fundamental to many real-world applications in various domains, serving as structural representations of knowledge. To deal with the increasing volume of new concepts needed to be organized as taxonomies, researchers turn to automatically completion of an existing taxonomy with new concepts. In this paper, we propose TaxoEnrich, a new taxonomy completion framework, which effectively leverages both semantic features and structural information in the existing taxonomy and offers a better representation of candidate position to boost the performance of taxonomy completion. Specifically, TaxoEnrich consists of four components: (1) taxonomy-contextualized embedding which incorporates both semantic meanings of concept and taxonomic relations based on powerful pretrained language models; (2) a taxonomy-aware sequential encoder which learns candidate position representations by encoding the structural information of taxonomy; (3) a query-aware sibling encoder which adaptively aggregates candidate siblings to augment candidate position representations based on their importance to the query-position matching; (4) a query-position matching model which extends existing work with our new candidate position representations. Extensive experiments on four large real-world datasets from different domains show that TaxoEnrich achieves the best performance among all evaluation metrics and outperforms previous state-of-the-art methods by a large margin.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**.

KEYWORDS

Taxonomy Completion, Self-supervised Learning, Knowledge Representation

ACM Reference Format:
Minhao Jiang[1], Xiangchen Song[2], Jieyu Zhang[3], Jiawei Han[1]. 2022. TaxoEnrich: Self-Supervised Taxonomy Completion via Structure-Semantic Representations. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3485447.3511935

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3511935

1 INTRODUCTION

Taxonomies have been widely used for organizing concepts structurally [14, 20, 21, 25, 28]. Specifically, to capture the "is-a" relationship between concept pairs, people often formulate the taxonomy into tree or directed acyclic graph (DAG) structure. Example applications could be found in e-commerce, where Amazon leverages product taxonomies for personalized recommendations and product navigation, and fine-grained named entity recognition where people rely on concept taxonomies (e.g., MeSH) [9] to extract and label useful information from massive corpus.

However, the construction of taxonomies usually requires a substantial amount of human curation. Such process is time-consuming and labor-intensive. Thus, it is extremely hard to handle the large number of emerging new concepts in downstream tasks, which is fairly common nowadays with the rising tide of big data. To tackle this issue, recent work [12, 19, 23, 30, 31, 34] turns to the tasks of the automatic expansion and completion of the existing taxonomy.

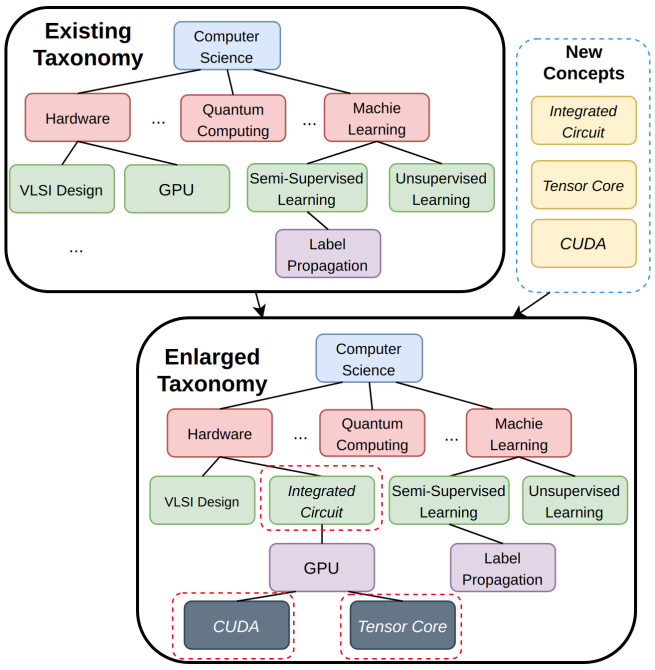


Figure 1: An example of inserting new concepts into an existing taxonomy of computer science terms. For each new concept, we aim to find the relatedness between the concept and each candidate position.

Previous taxonomy construction methods [3, 4, 10, 13, 26, 32] construct taxonomies from scratch, and highly rely on annotated hypernym pairs, which are expensive and sometimes inaccessible in practice. Therefore, automatic taxonomy expansion based on existing taxonomies is in great need and has gained increasing attention.

The recent studies on taxonomy expansion and completion achieved noticeable progress, which mainly contribute from two directions. (1) extract hierarchical information from the existing taxonomy, and utilize different ways to model the structural information in the existing taxonomy, such as local egonet [19], parent-query-child triplet [34], and mini-paths [30]. (2) leverage the supporting corpus to generate the embeddings of concepts directly. They either only used implicit relational semantics [12], or only relied on corpus to construct limited seed-guided taxonomy [5]. Very recently, [31] combines the representations from semantic sentences and local-subgraph encoding as the features of concepts. However, they only utilized light-weight multi-layer perceptron (MLP) for matching, which suffers from the limited representation power. In this paper, we follow [34] to focus on taxonomy completion, which aims to predict the most likely ⟨query, hypernym, hyponym⟩ triplet for a given query concept. For example, in Figure 1, when considering the query "*Integrated Circuit*", we aim to find its true parent "*Hardware*" and child "*GPU*".

To effectively leverage both semantic and structural information for better taxonomy completion performance, in this work, we propose TaxoEnrich, which aims to learn better representations for each candidate position and render new state-of-the-art taxonomy completion performance. Specifically, TaxoEnrich consists of four carefully-designed components. First, we propose a taxonomy-contextualized embedding generation process based on pseudo sentences extracted from existing taxonomy. The two types of pseudo sentences, i.e., ancestral and descendant pseudo sentences, capture taxonomic relations from two directions respectively. Then, the powerful pretrained language models are utilized to produce the taxonomy-contextualized embedding based on the extracted sentences. Secondly, to encode the structural information of the existing taxonomy in both vertical and horizontal views, we develop two novel encoders: a sequential feature encoder based on the pseudo sentences and a query-aware sibling encoder base on the importance of candidate siblings to the matching task. The former aims to learn a taxonomy-aware candidate position representations, while the latter further augments the position representations with adaptively aggregated candidate siblings information. Finally, we develop an effective query-position matching model by extending previous work [34] to incorporate our novel candidate position representations. Specifically, it takes into consideration both fine-grained (query to candidate parent) and coarse-grained (query to candidate position) relatedness for better taxonomy completion performance.

We conducted extensive experiments on four real-world taxonomies from different domains to test the performance of TaxoEnrich framework. Further more, we designed two variations of the framework, TaxoEnrich and TaxoEnrich-S to conduct ablation experiments to explore the utilization of different information under different datasets, along with studies to examine the effectiveness of each sub-module of the framework. Our results show that TaxoEnrich can more accurately capture the correct positions of query nodes than previous methods and achieve state-of-the-art performance on both taxonomy completion and expansion tasks.

To summarize, our major contributions include:

- We propose an effective embedding generation approach which can be applied generally for learning contextualized embedding for each concept node in a given taxonomy.
- We introduce the sequential feature encoders to capture vertical structural information of candidate positions in the taxonomy.
- We design an effective query-aware sibling encoder to incorporate horizontal structural information in the taxonomy.
- Extensive experiments demonstrate that our developed framework enhances the performance on both taxonomy completion and expansion task by a large margin over the previous works.

2 PROBLEM DEFINITION

In this section, we formally define the taxonomy completion task studied in the paper.

Taxonomy. Follow [19], we formulate a taxonomy $\mathcal{T}^0 = \{\mathcal{N}^0, \mathcal{E}^0\}$ as a directed acyclic graph where each node $n \in \mathcal{N}^0$ represents a concept and each directed edge $\langle n_p, n_c \rangle \in \mathcal{E}^0$ represents a taxonomic relationship between two concepts.

Taxonomy Completion. The taxonomy completion task [34] is defined as following: given an existing taxonomy $\mathcal{T}^0$ and a set of new concepts C, assuming that each concept in C is in the same semantic domain as $\mathcal{T}^0$, we aim to automatically find the most possible ⟨ hypernym, hyponym⟩ pairs for each new concept to complete the taxonomy. The output is $\mathcal{T} = \{\mathcal{N}, \mathcal{E}'\}$ where $\mathcal{N} = \mathcal{N}^0 \cup C$ and $\mathcal{E}'$ is the updated edges set after inserting new concepts.

Candidate Positions. In the taxonomy completion task, we define a valid candidate position as a pair of concept nodes in the existing taxonomy $\langle n_p, n_c \rangle$ where n_p is a parent of n_c. Note that one of n_p and n_c could be a pseudo placeholder node in case that the concepts needed to be inserted as root or leaf nodes.

The goal of taxonomy completion is to enrich the existing taxonomy by inserting new concepts. These new concepts are generally extracted from text corpus using entity extraction tools. Since this process is not the focus of the paper, we assume that the set of new concepts C is given, as well as their embedding, which is denoted by e_q for new concept n_q.

3 THE TAXOENRICH FRAMEWORK

In this section, we introduce the TaxoEnrich framework in details. We first introduce the taxonomy-contextualized embedding generation for each concept node in the existing taxonomy. Then, given the extracted taxonomy-contextualized embedding, we develop two encoders to learn the representation of candidate positions from vertical and horizontal views of the taxonomy respectively. Finally, we propose a query-to-position matching model which leverages various structural information and takes into consideration both fine- and coarse-grained relatedness to boost the matching performance. The overall framework of TaxoEnrich is in Figure 2.

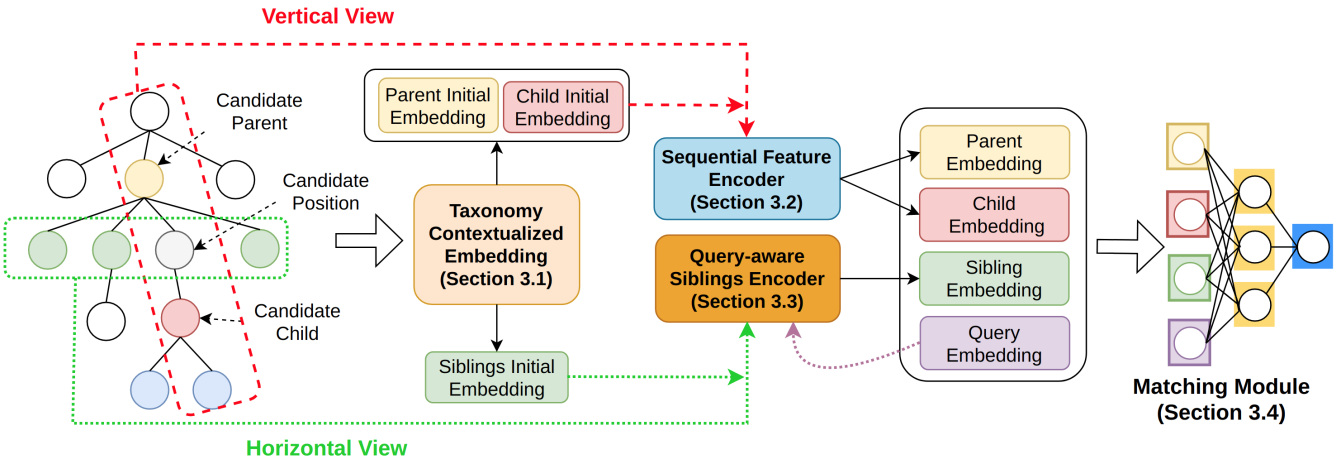


Figure 2: The complete architecture of TaxoEnrich **. The figure describes the workflow of** TaxoEnrich**, and the details are discussed in the corresponding section.**

3.1 Taxonomy-Contextualized Embedding

Here, we describe the generation process of taxonomy-contextualized embedding for each node in the taxonomy. Different from prior work which leverages static word embedding, such as Word2Vec and FastText [19, 34], or contextualized embedding solely based on an additional text corpus [30], we generate taxonomy-contextualized embedding based on taxonomy structure and concept surface name. The reason is that neighboring concepts in taxonomy are likely to share similar semantic meaning and it is hard to distinguish them based on predefined general-purpose embedding. With similar spirit, [31] also leverage pretrained language models to produce contextualized embedding based on limited number of taxonomy neighbor and the surface names is implicitly utilized in fine-tuning the pretrained language models. In contrast, we aim to fuse the information of all the descendant/ancestral concepts of the given concept, without fine-tuning a huge pretrained language model. Specifically, given a concept node, we build pseudo sentences based on Hearst patterns [18] to represent both the positional and semantic information. We separately consider the descendant/ancestral information by constructing descendant/ancestral pseudo sentences respectively as shown in Figure 3.

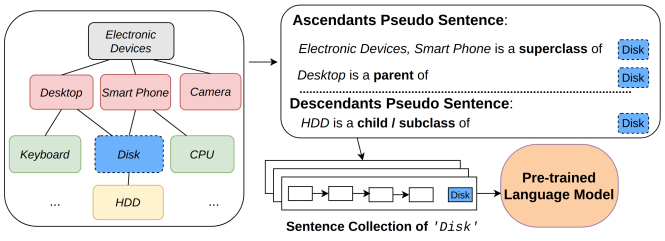


Figure 3: The visualization of taxonomy-contextualized generation process. In this example, we aim to extract the embedding of the concept node "*Disk*".

Formally, given a taxonomy $\mathcal{T}_0 = (\mathcal{V}_0, \mathcal{E}_0)$ and the candidate concept node v, we extract the following two types of pseudo sentences that represent taxonomic relationships:

(1) Ancestral Pseudo Sentences: We first extract paths connecting root and the candidate node v without duplicate nodes. In the extracted path $(p_1, p_2, \ldots, p_l)$, p_1 is the root node and $p_l = v$. We denote the i-th extracted path for node v as $P_a(v)_i = (p_1, p_2, \ldots, p_{l-1})$. Along each path, we generate the ancestral pseudo sentence as below:

$$\text{"}p_1, p_2, \ldots, p_{l-1} \text{ is a superclass of } v\text{"}$$
$$\text{or "}p_1, p_2, \ldots, p_{l-1} \text{ is an ascendant of } v\text{"}$$

All such words like "superclass" or "ascendant" that can represent the hierarchical relationship between the path and candidate nodes can be used for sentence generation. We denote the collection of such ancestral paths as $P_a(v)$ and the generated sentences as $S_a(v)$

(2) Descendant Pseudo Sentences: Similarly, the paths without duplicated nodes starting from the candidate node v to leaf nodes are extracted, denoted as $(p_1, p_2, \ldots, p_l)$ where $p_1 = v$ and p_l is a leaf node. In this case, along each path, we generate the sentence as below

$$\text{"}p_2, p_3, \ldots, p_l \text{ is a subclass of } v\text{"}$$
$$\text{or "}p_2, p_3, \ldots, p_l \text{ is a descendant of } v\text{"}$$

We denote the collection of such descendants paths as $P_d(v)$ and the generated sentences as $S_d(v)$

Then, the set of all the generated pseudo sentences is $S(v) = S_d(v) \cup S_a(v)$. As visualized in Figure 3, aiming to generate embeddings of the concept node "*Disk*", we only consider the ancestral and descendant paths, such as "*Electronic Devices, Smart Phone is a superclass of Disk*". Note that if the candidate node is leaf node or root, we will only consider one side pseudo sentences. Given the generated pseudo sentences, we apply a pretrained language models to generate taxonomy-contextualized embedding for each node. Specifically, we feed the pseudo sentences to the pretrained language model and collect the last hidden state representations of the concept node v,

which is averaged as the final taxonomy-contextualized embedding $\mathbf{x}_v \in \mathbb{R}^d$. In our preliminary experiments, we found that SciBERT is better than other models in representing concept representations. Hence, in this paper, we choose SciBERT [2] following [31]. And the comparison between different pre-trained language models in terms of performance are discussed in 7. Note that the taxonomy-contextualized embeddings are pre-computed and fixed for the following modules, which means we do not fine-tune the large pretrained language model.

3.2 Sequential Feature Encoder

Given the taxonomy-contextualized embedding $\mathbf{x}_v$ for existing node v, we develop a learnable sequential feature encoder $g(v)$ to encode the structural information of candidate positions in a vertical view of taxonomy. For a candidate position $\langle p, c\rangle$ consisting of candidate parent p and child c, we produce parent embedding $g(p)$ and child embedding $g(c)$ respectively. Specifically, for candidate parent p and its corresponding ancestral paths $P_a(p)$, we randomly sample a path p_p from $P_a(p)$ and apply a LSTM sequential encoder which inputs the sampled pseudo sentence and the taxonomy-contextualized embedding. Then, we concatenate the final hidden state of the LSTM encoder and the taxonomy-contextualized embedding $\mathbf{x}_p$ as parent embedding. Formally,

$$g(p) = \mathbf{x}_p \oplus \text{LSTM}(p_p; \Theta_1) \tag{1}$$

where Θ_1 is the learnable parameters of the LSTM encoder and $\oplus$ represents the concatenation operation. Similarly, we generate the child embedding $g(c)$ based on taxonomy-contextualized embedding $\mathbf{x}_c$ and descendant paths p_c from $P_d(c)$:

$$g(c) = \mathbf{x}_c \oplus \text{LSTM}(p_c; \Theta_2) \tag{2}$$

where, Θ_2 represents the learnable parameters of the LSTM encoder. The output $g(u), g(v) \in \mathbb{R}^h$ will be used as the embedding for candidate position nodes.

Through this sequential feature encoder, we are able to fuse the structural information of candidate position in a vertical view. This allows the candidate position representations to be aware of the "depth" information of the candidate position, i.e., whether the candidate position is in the top-level of taxonomy close to the root or in the bottom-level close to leave.

3.3 Query-Aware Siblings Encoder

The aforementioned sequential feature encoder incorporates the taxonomy structural information in a vertical view, however, it is of great importance to also encode the horizontal local information of the candidate position. Thus, we develop another encoder to incorporate the structural information in a horizontal view. Specifically, in addition to the candidate parent and child, we consider the candidate siblings, i.e., the children of candidate parent, of the query node.

However, incorporating candidate siblings is challenging than candidate parent and child. The reasons are twofold. First, compared to the candidate parent and child which compose the candidate position, candidate siblings could introduce noisy information and thus lead to sub-optimal results. For example, for the top-level of taxonomy, the candidate siblings could have quite diverse semantic meanings, which hinder good matching between candidate position and query node. Secondly, since some candidate parent could have substantial amount of children (candidate siblings), it is infeasible to incorporate all the candidate siblings without strategic selection.

To tackle these issues, we develop a query-aware siblings encoder, which adaptively selects part of the candidate siblings. Specifically, we measure the relatedness of a given query embedding e_q and each candidate sibling condition on the representation of candidate parent-child pair. Such relatedness is in turn used to aggregate the sibling information into a single siblings embedding. Mathematically, given candidate position $\langle n_p, n_c\rangle$ with corresponding embedding $g(p), g(c)$ and the set of candidate siblings $C(n_p) = \{s_1, s_2, \ldots, s_t\}$, we use a learnable bilinear matrix $\mathbf{W}_{\text{Sib}} \in \mathbb{R}^{d\times(2h+d)}$ to calculate the relatedness of query and candidate sibling s_i as

$$\phi_{s_i} = e_q^T \mathbf{W}_{\text{Sib}} \left[g(p), g(c), \mathbf{x}_{s_i}\right] \tag{3}$$

Then the relatedness score ϕ_{s_i} is normalized over the set of candidate siblings by a softmax function:

$$\alpha_{s_i} = \sigma_{\text{softmax}}(\phi_{s_i}) = \frac{\exp(\phi_{s_i})}{\sum_{s_j \in C(n_p)} \exp(\phi_{s_j})} \tag{4}$$

The normalized score α_{s_i} captures the importance of candidate sibling s_i for the specific query-position matching. In other words, it highlights the siblings relevant to the query condition on the candidate position while lessen the effect of irrelevant siblings. Finally, the sibling embeddings are aggregate based on the normalized score as

$$a(p) = \sum_{s_i \in C(n_p)} \alpha_{s_i} \mathbf{x}_{s_i} \tag{5}$$

where $a(p) \in \mathbb{R}^d$. During experiments, we found that such a query-aware siblings encoder renders good performance when only a subset of siblings are considered, which alleviates the heavy burden of aggregate over the potentially large amount of candidate siblings.

3.4 Query-Position Matching Model

Finally, given the representation of candidate parent $g(p)$, child $g(c)$ and siblings $a(p)$ as well as the given query embedding e_q, we are ready to present our final matching module, which outputs the matching score of query and candidate position for taxonomy completion task. In particular, we seek to learn a matching model s that outputs the desired relatedness score:

$$s(n_q, \langle n_p, n_c\rangle) = f(e_q, g(p), g(c), a(p)) \tag{6}$$

where f is a parametrized scoring function.

The previous study [34] showed that the simple matching model that learns one-to-one relatedness between the query node and the position pair ignores fine-grained relatedness between query and position component, i.e., the relatedness between $\langle n_q, n_p\rangle$ and $\langle n_q, n_c\rangle$. Therefore, inspired by [34], we propose a new matching model which incorporates the additional siblings embedding and learn more precise matching based on both fine-grained (query to candidate parent/child/siblings) and coarse-grained relatedness (query to position).

To learn both the fine-grained and coarse-grained relatedness between the query node and the candidate positions, we construct multiple auxiliary scorers that separately focus on the relationship between the query node and the candidate parent, the candidate child, the candidate siblings and the candidate position, respectively.

We adopt the Neural Tensor Network (NTN) [22] as the base models. Given vectors $u \in \mathbb{R}^{d_u}, v \in \mathbb{R}^{d_v}$, an NTN can be defined as

$$\text{NTN}(u, v) = \mathbf{w}^T \sigma_{\tanh}(h(u, v)) \tag{7}$$

$$h(u, v) = u^T \mathbf{W}^{[1:k]} v + \mathbf{V} \begin{bmatrix} u \\ v \end{bmatrix} + \mathbf{b} \tag{8}$$

where $\sigma_{\tanh}$ is a tanh function and $\mathbf{w} \in \mathbb{R}^k$, $\mathbf{W}^{[1:k]} \in \mathbb{R}^{d_u \times d_v \times k}$, $\mathbf{V} \in \mathbb{R}^{k \times (d_u + d_v)}$ and $\mathbf{b} \in \mathbb{R}^k$ are learnable parameters. Note that k is a hyperparameter in NTN.

Then our multiple scorer can be defined as

$$S_1(n_q, n_p) = \mathbf{w}_1^T \sigma_{\tanh}(h_1(e_q, g(p))) \tag{9}$$

$$S_2(n_q, n_c) = \mathbf{w}_2^T \sigma_{\tanh}(h_2(e_q, g(c))) \tag{10}$$

$$S_3(n_q, C(n_p)) = \mathbf{w}_3^T \sigma_{\tanh}(h_3(e_q, a(p))) \tag{11}$$

$$S_4(n_q, \langle n_p, n_c \rangle) = \mathbf{w}_4^T \sigma_{\tanh}(h_4(e_q, \left[g(p), g(c), a(p)\right])) \tag{12}$$

We omit the learnable parameters inside each h for notation convenience. In this formulation, S_1, S_2, S_3 aim to learn the fine-grained relatedness for $\langle n_q, n_p \rangle$, $\langle n_q, n_c \rangle$, $\langle n_q, C(n_p) \rangle$ separately by predicting whether the n_p, n_c, and $C(n_p)$ is the reasonable parent, child, and siblings, respectively. Differently, S_4 is designed for coarse-grained relatedness between the query node and the candidate position. Eventually we construct a primal scorer which incorporates the all the auxiliary scorers.

$$S_p(n_q, \langle n_p, n_c \rangle) = \mathbf{u}_p^T \sigma_{\tanh}(\left[h_1, h_2, h_3, h_4\right])) \tag{13}$$

We omit the input of each function h for simplicity. In this case, even though S_p and S_4 share the same supervision signal, the concatenation of internal representations of other auxiliary scorers in S_p will allow it to capture accurate matching information based on S_1, S_2, S_3 when S_4 cannot learn correct coarse-grained relatedness.

3.4.1 Learning Objectives. For each auxiliary scorers, since the model is trained for binary classification task to calculate the relatedness between the query node and the target objective, we adopt the binary cross-entropy loss. Thus, the learning objective for each scorer can be formulated as

$$\mathcal{L}_k = -\frac{1}{|\mathbb{D}|} \sum_{(\mathbf{X}_i, y_i) \in \mathbb{D}} y_i \cdot \log(S_i(\mathbf{X}_i)) + (1 - y_i) \cdot \log(1 - S_i(\mathbf{X}_i)) \tag{14}$$

where $\mathbb{D}$ is the dataset formulated by the self-supervised generation following similar methods proposed in [19, 34], and $(\mathbf{X}_i, y_i)$ is the generated data pair in the dataset, and $k \in \{p, 1, 2, 3, 4\}$ represents each scorer. In this case, the final learning objective $\mathcal{L}(\Theta)$ that focuses on the primal task will naturally be defined as

$$\mathcal{L}(\Theta) = \mathcal{L}_p + \lambda_1 \mathcal{L}_1 + \lambda_2 \mathcal{L}_2 + \lambda_3 \mathcal{L}_3 + \lambda_4 \mathcal{L}_4 \tag{15}$$

4 EXPERIMENTS

4.1 Experiment Setup

Dataset. We evaluate the performance of TaxoEnrich framework on the following four real-world large-scale datasets. The statistics of each dataset are listed in Table 1.

- **Microsoft Academic Graph (MAG):** This public Field-of-Study (FoS) taxonomy contains over 660 thousand scientific concepts and more than 700 thousand taxonomic relations. We follow the data preprocessing in [19] to only select partial taxonomies under the computer science (**MAG-CS**) and psychology (**MAG-PSY**) domain [21].
- **WordNet:** We collect the concepts and taxonomic relations from verbs and nouns sub-taxonomies based on WordNet 3.0 (**WordNet-Noun**, **WordNet-Verb**). These two sub-fields are the only parts that have fully-developed taxonomies in WordNet. In practice, due to the scarcity in the dataset, i.e. there are many disconnected components in the both taxonomies, we added a pseudo root named "*Noun*" and "*Verb*" and connect this root to the head of each connected components in the taxonomies for generate a more complete taxonomic structure.

Table 1: Dataset Statistics. $|\mathcal{N}|$ represents the number of nodes in the taxonomy and $|\mathcal{E}|$ represents the number of edges in the taxonomy. $|\mathcal{D}|$ indicates the taxonomy depth. # of Sentences denotes the number of pseudo sentences generated by the embedding generation module in each taxonomy.

Dataset	$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{D}\rvert$	# of Sentences
MAG-CS	24,754	42,329	6	227,609
MAG-PSY	23,187	30,041	6	111,194
WordNet-Noun	83,073	76,812	20	236,454
WordNet-Verb	13,936	13,403	13	34,654

Follow the dataset splitting settings used in [19, 34], we sample 1000 nodes for validation and test respectively in each dataset. Then we use the remaining nodes to construct the initial taxonomy.

4.2 Compared Methods

To fully understand the performance of our framework, we compare our model with the following methods.

- **Bilinear Model** [24] incorporates the interaction of two concept embeddings, i.e., ⟨parent, child⟩ entity pair embeddings, through a simple bilinear form. This method serves as a baseline result to check the comparable performance of each framework.
- **TaxoExpan** [19] is a state-of-the-art taxonomy expansion framework, which leverages the positional-enhanced graph neural network to capture the relationship between query nodes and local egonet, along with InfoNCE loss [16] to increase the robustness of the model.
- **ARBORIST** [12] is a state-of-the-art taxonomy expansion framework which aims for taxonomies with heterogeneous edge semantics and optimizes a large margin ranking loss with a dynamic margin function.
- **TMN** [34] is a state-of-the-art taxonomy completion framework and also the first framework that proposed the completion task, and computed the matching score between the query concept and ⟨ hypernym, hyponym⟩ pairs.
- **GenTaxo** [31] is a state-of-the-art taxonomy completion framework using both sentence-based and subgraph-based encodings of the nodes to perform the matching. Since part of the framework concentrates on concept name generation tasks, which is not the focus of this paper, we adopt the **GenTaxo++** assuming the newly added nodes are given. [1]

[1]Note that since the implementation code of GenTaxo [31] is not released, we implemented the framework based on the description in the paper.

Table 2: Overall results of Taxonomy Completion task on the four large-scale datasets. ** indicates the results are from [34].

Method	MAG-CS							
	MR	MRR	Recall@1	Recall@5	Recall@10	Precision@1	Precision@5	Precision@10
Bilinear	3360.343 ± 6.126	0.026 ± 0.000	0.000 ± 0.000	0.003 ± 0.000	0.006 ± 0.000	0.001± 0.000	0.002± 0.000	0.003± 0.000
TaxoExpan	823.075 ± 114.638	0.193 ± 0.007	0.030 ± 0.002	0.095 ± 0.004	0.137 ± 0.007	0.132 ± 0.010	0.083 ± 0.003	0.059 ± 0.003
ARBORIST**	1142.335 ± 19.249	0.133 ± 0.004	0.008 ± 0.001	0.044 ± 0.003	0.075 ± 0.003	0.037 ± 0.004	0.038 ± 0.003	0.033 ± 0.001
TMN	436.319 ± 13.128	0.243 ± 0.005	0.056 ± 0.001	0.145 ± 0.004	0.189 ± 0.005	0.245 ± 0.006	0.126 ± 0.003	0.082 ± 0.002
GenTaxo	13213.731 ± 662.688	0.239 ± 0.006	0.082 ± 0.002	0.185 ± 0.008	0.218 ± 0.008	0.254 ± 0.010	0.131 ± 0.007	0.085 ± 0.003
TaxoEnrich-S	**73.680 ± 1.346**	0.545 ± 0.002	0.154 ± 0.006	0.396 ± 0.003	0.534 ± 0.002	0.251 ± 0.016	0.129 ± 0.002	0.087 ± 0.001
TaxoEnrich	87.798 ± 1.512	**0.578 ± 0.001**	**0.162 ± 0.004**	**0.434 ± 0.005**	**0.574 ± 0.003**	**0.274 ± 0.017**	**0.141 ± 0.002**	**0.093 ± 0.002**
Method	**MAG-PSY**							
	MR	**MRR**	**Recall@1**	**Recall@5**	**Recall@10**	**Precision@1**	**Precision@5**	**Precision@10**
Bilinear	2118.204 ± 4.152	0.032 ± 0.000	0.000 ± 0.000	0.001 ± 0.000	0.003 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
TaxoExpan	345.679 ± 24.306	0.441 ± 0.005	0.122 ± 0.003	0.287 ± 0.007	0.364 ± 0.009	0.249 ± 0.007	0.117 ± 0.003	0.074 ± 0.002
ARBORIST**	547.723 ± 20.165	0.344 ± 0.012	0.062 ± 0.009	0.185 ± 0.011	0.256 ± 0.013	0.126 ± 0.018	0.076 ± 0.004	0.052 ± 0.003
TMN	159.550 ± 5.290	0.531 ± 0.007	0.175 ± 0.002	0.369 ± 0.005	0.446 ± 0.009	0.358 ± 0.004	0.150 ± 0.002	0.091 ± 0.002
GenTaxo	7482.516 ± 2600.713	0.464 ± 0.022	0.183 ± 0.116	0.402 ± 0.066	0.440 ± 0.039	0.376 ± 0.119	0.164 ± 0.027	0.090 ± 0.008
TaxoEnrich-S	149.660 ± 3.430	0.561 ± 0.005	0.221 ± 0.010	0.420 ± 0.007	0.480 ± 0.007	0.365 ± 0.020	0.178 ± 0.003	0.117 ± 0.001
TaxoEnrich	**122.247 ± 3.241**	**0.583 ± 0.010**	**0.234 ± 0.009**	**0.424 ± 0.013**	**0.510 ± 0.018**	**0.374 ± 0.021**	**0.186 ± 0.002**	**0.124 ± 0.002**
Method	**WordNet-Noun**							
	MR	**MRR**	**Recall@1**	**Recall@5**	**Recall@10**	**Precision@1**	**Precision@5**	**Precision@10**
Bilinear	3290.858 ± 14.668	0.196 ± 0.000	0.013 ± 0.000	0.063 ± 0.000	0.109 ± 0.000	0.023 ± 0.001	0.022 ± 0.000	0.019 ± 0.000
TaxoExpan	970.858 ± 50.995	0.390 ± 0.004	0.066 ± 0.002	0.186 ± 0.003	0.269 ± 0.007	0.114 ± 0.003	0.065 ± 0.001	0.047 ± 0.001
ARBORIST**	2993.341 ± 114.749	0.217 ± 0.005	0.021 ± 0.001	0.073 ± 0.002	0.125 ± 0.002	0.036 ± 0.021	0.025 ± 0.001	0.022 ± 0.000
TMN	827.371 ± 24.310	0.367 ± 0.006	0.054 ± 0.002	0.169 ± 0.002	0.256 ± 0.004	0.095 ± 0.002	0.058 ± 0.000	0.044 ± 0.001
GenTaxo	57871.589 ± 89.230	0.286 ± 0.162	0.025 ± 0.007	0.169 ± 0.049	0.268 ± 0.118	0.109 ± 0.013	0.024 ± 0.007	0.029 ± 0.001
TaxoEnrich-S	230.576 ± 6.472	0.426 ± 0.018	**0.125 ± 0.019**	0.212 ± 0.012	0.321 ± 0.018	0.216 ± 0.024	0.108 ± 0.004	0.078 ± 0.003
TaxoEnrich	**227.839 ± 12.247**	**0.442 ± 0.018**	0.123 ± 0.012	**0.248 ± 0.011**	**0.351 ± 0.019**	**0.226 ± 0.023**	**0.115 ± 0.002**	**0.098 ± 0.002**
Method	**WordNet-Verb**							
	MR	**MRR**	**Recall@1**	**Recall@5**	**Recall@10**	**Precision@1**	**Precision@5**	**Precision@10**
Bilinear	1866.736 ± 5.020	0.174 ± 0.000	0.012 ± 0.001	0.054 ± 0.000	0.095 ± 0.000	0.017 ± 0.001	0.016 ± 0.000	0.014 ± 0.000
TaxoExpan	853.308 ± 18.302	0.325 ± 0.007	0.069 ± 0.001	0.169 ± 0.003	0.228 ± 0.008	0.104 ± 0.002	0.051 ± 0.001	0.034 ± 0.001
ARBORIST**	2993.341 ± 4.950	0.206 ± 0.011	0.016 ± 0.004	0.073 ± 0.011	0.016 ± 0.011	0.024 ± 0.006	0.022 ± 0.003	0.018 ± 0.002
TMN	832.541 ± 29.589	0.354 ± 0.010	0.081 ± 0.007	0.194 ± 0.013	0.259 ± 0.014	0.121 ± 0.011	0.059 ± 0.004	0.039 ± 0.002
GenTaxo	2765.745 ± 262.631	0.428 ± 0.117	0.118 ± 0.069	0.208 ± 0.104	0.239 ± 0.112	0.235 ± 0.152	0.122 ± 0.038	0.066 ± 0.016
TaxoEnrich-S	**304.565** ± 3.628	0.442 ± 0.004	0.128 ± 0.003	**0.256 ± 0.012**	**0.350 ± 0.009**	0.242 ± 0.005	0.121 ± 0.004	0.074 ± 0.001
TaxoEnrich	320.064 ± 14.153	**0.452 ± 0.005**	**0.143 ± 0.002**	0.252 ± 0.014	0.347 ± 0.006	**0.276 ± 0.004**	**0.126 ± 0.001**	**0.081 ± 0.002**

We also include two variants of TaxoEnrich in experiments for ablation study:

- **TaxoEnrich-S**: In this version, we exclude the sibling information from the matching model, since in sparse taxonomies, such as WordNet, the siblings cannot represent the precise candidate positions, and might still introduce noisy information when computing the relateness between query node and candidate position.
- **TaxoEnrich**: In this version, we adopt the full framework of TaxoEnrich as described above. We will examine the difference between two variants through further experiments.

4.3 Evaluation Metrics

Since the result from the model's output is a ranking list of candidate positions for each query node, following the guidelines in [19, 34], we utilize the following rank-based metrics to evaluate the performance our framework and the comparison methods.

- **Mean Rank (MR)**. This metric measures the average rank position of a query concept's true position among all candidate positions. For queries with multiple correct positions, we first calculate the rank position of each individual triplet and then take the average of all rank positions. Smaller value in this metric indicates the better performance of the model.
- **Mean Reciprocal Rank (MRR)**. We follow [29] to compute the reciprocal rank of a query concept's true positions using a scaled MRR. In the evaluation, we scale the MRR by 10 to enlarge the difference between different models clearly.
- **Recall@k** measures the number of query concepts' true positions ranked in the top k, divided by the total number of true positions of all query concepts.
- **Precision@k** measures the number of query concepts' true positions ranked in the top k, divided by the total number of queries times k.

For all the evaluation metrics listed above except for MR, the larger value indicates better performance of the model. During the evaluation, since MR and MRR are the only metrics that concentrates on the performance of all predictions in the taxonomy in general, we consider them as the most important metric for evaluation.

5 EXPERIMENTAL RESULTS

In this section we will first discuss the experiment results on both taxonomy completion and expansion tasks which demonstrated the superiority of our TaxoEnrich method. Then to further understand the contributions from each of our model design, we conduct ablation studies. Finally we performed case studies to further illustrate the effectiveness of TaxoEnrich.

5.1 Performance on Taxonomy Completion

The overall performance of compared methods and the proposed framework is indicated in Table 2. First, we can see that the performance of the framework tends to become better when the complexity of local structure increases, from the one-to-one matching in TaxoExpan to triplet in TMN , and the neighboring paths and subgraph encoding in GenTaxo. Second, we can generally observe the power of pre-trained language models in the representations of concept nodes in the taxonomy. The frameworks including GenTaxo and TaxoEnrich utilizing language models have generally better performance in the precision@k and recall@k metrics.

In terms of MR , we can see that TaxoEnrich obtained most performance improvement in MAG-CS dataset since the computer science taxonomy has the most complete taxonomic structure compared with other datasets, allowing for more accurate taxonomy-contextualized embeddings generated by Section 3.1. And in WordNet datasets the performance in MR metric is improved by a relatively large margin while all frameworks do not perform as well as in MAG datasets. In terms of precision@k and recall@k, our method also shows noticeable improvement over baseline models. In the previous methods, the static embedding method failed to capture the similar semantic meaning between different concept nodes. And we can see GenTaxo renders competing performance on these two metrics, but tends to be unstable and perform not well in ranking metrics. The primary reason for this observation is that while the language-based embeddings can provide pretty accurate positional information, its light-weight MLP matching module prevents it from capturing useful relatedness between query node and candidate position.

For two WordNet datasets, we can see that other frameworks are inclined to have similarly poor performance due to the scarcity of taxonomies. The non-connectivity causes the matching module difficult to extract the relations during the training. Therefore, the manually added pseudo root for sentence generation would maintain the taxonomic structure information in the representations of concept node and candidate positions, allowing the framework to capture both the structure and semantic information of each node.

In the comparison between TaxoEnrich-S and TaxoEnrich, we can observe that, in two MAG datasets, the incorporation of sibling information in TaxoEnrich would have better performance. However, it will also cause a drop in MR metric except for MAG-PSY and WordNet-Noun datasets since the randomly extracted siblings will still introduce noisy information in the matching module. In the WordNet datasets, the performance of two methods are very similar. This is because with the scarcity in the taxonomies, i.e., the lack of siblings, will mislead the model to incorporate inaccurate sibling information, causing a clear difference for MR metric. On the other hand, the precision in TaxoEnrich is still better than TaxoEnrich-S, which illustrates the effectiveness of siblings in representing the positional information.

5.2 Performance on Taxonomy Expansion

Taxonomy expansion is a special case of the taxonomy completion task where new concepts are all leaf nodes. In this case, we would like to further explore the performance of TaxoEnrich on taxonomy expansion task on MAG-CS and WordNet-Verb dataset, compared with TaxoExpan, TMN, and GenTaxo framework. As indicated in Table 3, we can also observe that TaxoEnrich outperforms other methods by a large margin in all metrics.

Table 3: Results of Taxonomy Expansion task on the MAG-CS and WordNet-Verb datasets.

Method	MAG-CS			
	MR	MRR	Recall@1	Precision@1
TaxoExpan	197.776 ± 16.038	0.562 ± 0.023	0.100 ± 0.011	0.163 ± 0.018
TMN	118.963 ± 6.307	0.689 ± 0.005	0.174 ± 0.002	0.283 ± 0.004
GenTaxo	140.262 ± 40.398	0.634 ± 0.044	0.149 ± 0.020	0.294 ± 0.096
TaxoEnrich-S	**67.947 ± 1.121**	**0.721 ± 0.008**	**0.182 ± 0.005**	**0.304 ± 0.008**
Method	**WordNet-Verb**			
	MR	**MRR**	**Recall@1**	**Precision@1**
TaxoExpan	665.409 ± 137.250	0.406 ± 0.056	0.085 ± 0.018	0.095 ± 0.004
TMN	615.021 ± 166.375	0.423 ± 0.056	0.110 ± 0.021	0.124 ± 0.009
GenTaxo	6046.363 ± 439.305	0.155 ± 0.010	0.094 ± 0.019	0.141 ± 0.079
TaxoEnrich-S	**217.842 ± 5.230**	**0.481 ± 0.071**	**0.162 ± 0.082**	**0.294 ± 0.031**

Table 4: Ablation studies on the incorporation of siblings in embeddings on MAG-CS dataset. The framework that randomly incorporates sibling information in embedding generation module is denoted as TaxoEnrich-Sib **for simplicity.**

Method	MAG-CS			
	MR	MRR	Recall@1	Precision@1
TaxoEnrich-Sib	122.144 ± 3.219	0.513 ± 0.006	0.138 ± 0.000	0.224 ± 0.001
TaxoEnrich-S	**73.680 ± 1.346**	**0.545 ± 0.002**	**0.154 ± 0.006**	**0.251 ± 0.016**

5.3 Ablation Studies

In this section, we conduct the ablation studies on the major components of TaxoEnrich framework: 1) Incorporation of sibling information separated from embedding generation; 2) The implementations of different feature encoder models to capture different structural information. Note that, in the ablation study experiment, we used TaxoEnrich-S model for more direct and simpler comparison between the embeddings and modules. Additional ablation studies about hyperparameters are presented in appendix.

Table 5: Ablation studies on MAG-CS dataset with different feature encoders. Some results are from the main table.

Method	Distribution Model	MAG-CS		
		MR	Recall@1	Precision@1
TaxoExpan	Raw Embedding	3360.343 ± 6.126	0.000 ± 0.000	0.001 ± 0.001
TaxoExpan	Raw + PGAT	823.075 ± 114.638	0.030 ± 0.002	0.132 ± 0.010
TMN	Raw Embedding	636.254 ± 36.465	0.036 ± 0.005	0.156 ± 0.008
TMN	Raw + LSTM + PGAT	436.319 ± 13.128	0.056 ± 0.001	0.245 ± 0.006
TaxoEnrich-S	Raw Embedding	103.016 ± 6.589	0.145 ± 0.004	0.236 ± 0.010
TaxoEnrich-S	Raw + LSTM	**73.680 ± 1.346**	**0.154 ± 0.006**	**0.251 ± 0.024**
TaxoEnrich-S	Raw + LSTM + PGAT	100.188 ± 2.214	0.150 ± 0.004	0.244 ± 0.001

Table 6: Case Studies of predicted positions on MAG-CS with both leaf and internal query concepts

Method	MAG-CS		
	Query	Top 5 Predicted Positions ($^{\text{hypernym}}_{\text{hyponym}}$), True Positions in Red	Ranking of True Positions
TaxoEnrich	heap	($^{\text{programming language}}_{\text{heaps law}}$), ($^{\text{programming language}}_{\text{pseudo leaf}}$), ($^{\text{algorithm}}_{\text{pseudo leaf}}$), ($^{\text{programming language}}_{\text{heap overflow}}$), ($^{\text{algorithm}}_{\text{heap overflow}}$)	1,4,5
TMN		($^{\text{reference counting}}_{\text{pseudo leaf}}$), ($^{\text{garbage collection}}_{\text{pseudo leaf}}$), ($^{\text{programming language}}_{\text{pseudo leaf}}$), ($^{\text{algorithm}}_{\text{pseudo leaf}}$), ($^{\text{binary heap}}_{\text{collection}}$)	530, 634, 4884
TaxoEnrich	perl	($^{\text{programming language}}_{\text{pseudo leaf}}$), ($^{\text{operating system}}_{\text{pseudo leaf}}$), ($^{\text{programming language}}_{\text{perl critic}}$), ($^{\text{programming language}}_{\text{perl 6}}$), ($^{\text{programming language}}_{\text{heap lustre}}$)	2,3
TMN		($^{\text{programming language}}_{\text{pseudo leaf}}$), ($^{\text{software}}_{\text{pseudo leaf}}$), ($^{\text{operating system}}_{\text{pseudo leaf}}$), ($^{\text{database}}_{\text{pseudo leaf}}$), ($^{\text{scripting language}}_{\text{perl critic}}$)	12, 345
TaxoEnrich	all pairs testing	($^{\text{test suite}}_{\text{pseudo leaf}}$), ($^{\text{model based testing}}_{\text{pseudo leaf}}$), ($^{\text{metamorphic testing}}_{\text{pseudo leaf}}$), ($^{\text{test driven development}}_{\text{pseudo leaf}}$), ($^{\text{redundant code}}_{\text{perl critic}}$)	1, 2
TMN		($^{\text{metamorphic testing}}_{\text{pseudo leaf}}$), ($^{\text{random testing}}_{\text{pseudo leaf}}$), ($^{\text{artificial intelligence}}_{\text{pseudo leaf}}$), ($^{\text{algorithm}}_{\text{pseudo leaf}}$), ($^{\text{machine learning}}_{\text{pseudo leaf}}$)	6, 133
TaxoEnrich	sensor hub	($^{\text{embedded system}}_{\text{pseudo leaf}}$), ($^{\text{operating system}}_{\text{pseudo leaf}}$), ($^{\text{computer network}}_{\text{pseudo leaf}}$), ($^{\text{computer hardware}}_{\text{pseudo leaf}}$), ($^{\text{embedded system}}_{\text{virtual metrology}}$)	1, 3, 4
TMN		($^{\text{embedded system}}_{\text{pseudo leaf}}$), ($^{\text{operating system}}_{\text{pseudo leaf}}$), ($^{\text{computer hardware}}_{\text{pseudo leaf}}$), ($^{\text{software}}_{\text{pseudo leaf}}$), ($^{\text{wearable computer}}_{\text{pseudo leaf}}$)	1, 78, 3

5.3.1 The Effectiveness of Query-Aware Sibling Encoder. In this section, we further discuss the effectiveness of approaches of incorporating sibling information in our framework. We argue that simple including sibling information in embeddings would actually introduce noisy information to the framework. In many cases, some high-level concepts, such as "*Artificial Intelligence*" or "*Machine Learning*" in MAG-CS taxonomy, have thousands of children. Therefore, it is unrealistic to consider all siblings, or unreasonable to randomly consider some of them. Thus, we conduct experiments to verify this assumption by randomly selecting at most 5 siblings in the process of embedding generation. However, as shown in Table 4, such operation would not only prevent the framework from recognizing correct positions, but increase the embedding generation time to the three times of the original in the implementation.

5.3.2 Different Feature Encoders in TaxoEnrich . We will continue to examine the superior performance of TaxoEnrich with different feature encoders and the effectiveness of such methods on other frameworks. The techniques of encoding features before matching have been experimented by previous methods [19, 30], showing that the neighboring terms of the candidate position would better utilize the structural information. We implement different feature encoding: the raw embeddings, LSTM encoding, PGAT encoding, and the combinations of these three, i.e., concatenating the output of encoders as input for matching module, for comparison in this section. Through experiments in Table 5, we can see that, the encoded features will improve the performance of all frameworks by a large margin, and TaxoEnrich can still outperform other methods regardless of encoded embeddings of concept nodes.

5.4 Case Studies

We demonstrate the effectiveness of TaxoEnrich framework by predicting true positions of several query concepts in MAG-CS datasets in Table 6. For high-level concepts like "*heap*", TaxoEnrich ranks all the true positions at top 5, while TMN can only identify part of the true positions' information, like ("*algorithm, leaf*") for "*heap*". And for those leaf nodes, such as "*all pairs testing*" and "*sensor hub*", we can observe that TMN will have much better performance. However, it will include some coarse high-level concepts such as "*machine learning*" and "*artificial intelligence*". In general, we can see TaxoEnrich works better than baselines for recovering true positions, and the top predictions by TaxoEnrich generally follow reasonable consistency.

6 RELATED WORK

Automatic taxonomy construction is a long-standing task in the literature. Existing taxonomy construction methods leverage lexical features from the resource corpus such as lexical-patterns [1, 4, 6, 15] or distributional representations [7, 11, 13, 17, 27, 33] to construct a taxonomy from scratch. However, in many real-world applications, some existing taxonomies may have already been laboriously curated and are deployed in online systems, which calls for solutions to the taxonomy expansion problem. To this end, multitudinous methods have been proposed recently to solve the taxonomy expansion problem [12, 14, 19, 30]. For example, ARBORIST [12] studies expanding taxonomies by jointly learning latent representations for edge semantics and taxonomy concepts; TaxoExpan [19] proposes position-enhanced graph neural networks to encode the relative position of terms and a robust InfoNCE loss [16]; STEAM [30] re-formulates the taxonomy expansion task as a mini-path-based prediction task and proposes to solve it through a multi-view co-training objective. Some other methods were proposed for taxonomy completion, such as TMN [34] focuses on taxonomy completion task with channel-gating mechanism and triplet matching network; and GenTaxo [31] collects information from complex local-structure information and learns to generate concept's full name from corpus.

7 CONCLUSION

In this paper, we proposed TaxoEnrich to enhance taxonomy completion task with self-supervision. It captures the hierarchical and semantic information of concept nodes based on the taxonomic relations in the existing taxonomy. Additionally, the selective query-aware attention module and elaborately designed matching module further improves the performance of learning relatedness between query node and candidate position. Extensive experimental results elucidated the effectiveness of TaxoEnrich by showing that it largely outperforms the previous methods achieving state-of-the-art performance on both taxonomy completion and expansion tasks.

REFERENCES

[1] Eugene Agichtein and Luis Gravano. 2000. Snowball: extracting relations from large plain-text collections. In *DL '00*.

[2] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. arXiv:1903.10676 [cs.CL]

[3] Amit Gupta, Rémi Lebret, Hamza Harkous, and Karl Aberer. 2017. Taxonomy induction using hypernym subsequences. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1329–1338.

[4] Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Coling 1992 volume 2: The 15th international conference on computational linguistics*.

[5] Jiaxin Huang, Yiqing Xie, Yu Meng, Yunyi Zhang, and Jiawei Han. 2020. Corel: Seed-guided topical taxonomy construction by concept learning and relation transferring. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1928–1936.

[6] Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M. Kaplan, Timothy P. Hanratty, and Jiawei Han. 2017. MetaPAD: Meta Pattern Discovery from Massive Text Corpora. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2017).

[7] Wengong Jin, Regina Barzilay, and Tommi S. Jaakkola. 2018. Junction Tree Variational Autoencoder for Molecular Graph Generation. In *ICML*.

[8] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

[9] Carolyn E Lipscomb. 2000. Medical subject headings (MeSH). *Bulletin of the Medical Library Association* 88, 3 (2000), 265.

[10] Xueqing Liu, Yangqiu Song, Shixia Liu, and Haixun Wang. 2012. Automatic taxonomy construction from keywords. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1433–1441.

[11] Anh Tuan Luu, Yi Tay, Siu Cheung Hui, and See-Kiong Ng. 2016. Learning Term Embeddings for Taxonomic Relation Identification Using Dynamic Weighting Neural Network. In *EMNLP*.

[12] Emaad Manzoor, Rui Li, Dhananjay Shrouty, and Jure Leskovec. 2020. Expanding Taxonomies with Implicit Edge Semantics. In *Proceedings of The Web Conference 2020*. 2044–2054.

[13] Yuning Mao, Xiang Ren, Jiaming Shen, Xiaotao Gu, and Jiawei Han. 2018. End-to-end reinforcement learning for automatic taxonomy induction. *arXiv preprint arXiv:1805.04044* (2018).

[14] Yuning Mao, Tong Zhao, Andrey Kan, Chenwei Zhang, Xin Luna Dong, Christos Faloutsos, and Jiawei Han. 2020. Octet. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Jul 2020). https://doi.org/10.1145/3394486.3403274

[15] Ndapandula Nakashole, Gerhard Weikum, and Fabian M. Suchanek. 2012. PATTY: A Taxonomy of Relational Patterns with Semantic Types. In *EMNLP*.

[16] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

[17] Stephen Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet Selective: Supervised Distributional Hypernymy Detection. In *COLING*.

[18] Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. 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, 358–363. https://doi.org/10.18653/v1/P18-2057

[19] Jiaming Shen, Zhihong Shen, Chenyan Xiong, Chi Wang, Kuansan Wang, and Jiawei Han. 2020. TaxoExpan: Self-supervised taxonomy expansion with position-enhanced graph neural network. In *Proceedings of The Web Conference 2020*. 486–497.

[20] Jiaming Shen, Jinfeng Xiao, Xinwei He, Jingbo Shang, Saurabh Sinha, and Jiawei Han. 2018. Entity Set Search of Scientific Literature: An Unsupervised Ranking Approach. arXiv:1804.10877 [cs.IR]

[21] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*. 243–246.

[22] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[23] Xiangchen Song, Jiaming Shen, Jieyu Zhang, and Jiawei Han. 2021. Who Should Go First? A Self-Supervised Concept Sorting Model for Improving Taxonomy Expansion. *arXiv preprint arXiv:2104.03682* (2021).

[24] Ilya Sutskever, Ruslan R Salakhutdinov, and Joshua B Tenenbaum. 2009. Modelling relational data using bayesian clustered tensor factorization. (2009).

[25] Denny Vrandečić. 2012. Wikidata: A New Platform for Collaborative Data Collection. In *Proceedings of the 21st International Conference on World Wide Web* (Lyon, France) *(WWW '12 Companion)*. Association for Computing Machinery, New York, NY, USA, 1063–1064. https://doi.org/10.1145/2187980.2188242

[26] Chi Wang, Marina Danilevsky, Nihit Desai, Yinan Zhang, Phuong Nguyen, Thrivikrama Taula, and Jiawei Han. 2013. A phrase mining framework for recursive construction of a topical hierarchy. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 437–445.

[27] Julie Weeds, David J. Weir, and Diana McCarthy. 2004. Characterising Measures of Lexical Distributional Similarity. In *COLING*.

[28] Carl Yang, Jieyu Zhang, and Jiawei Han. 2020. Co-embedding network nodes and hierarchical labels with taxonomy based generative adversarial networks. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 721–730.

[29] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 974–983.

[30] Yue Yu, Yinghao Li, Jiaming Shen, Hao Feng, Jimeng Sun, and Chao Zhang. 2020. STEAM: Self-Supervised Taxonomy Expansion with Mini-Paths. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1026–1035.

[31] Qingkai Zeng, Jinfeng Lin, Wenhao Yu, Jane Cleland-Huang, and Meng Jiang. 2021. Enhancing Taxonomy Completion with Concept Generation via Fusing Relational Representations. *arXiv preprint arXiv:2106.02974* (2021).

[32] Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian Sadler, Michelle Vanni, and Jiawei Han. 2018. TaxoGen: Unsupervised Topic Taxonomy Construction by Adaptive Term Embedding and Clustering. *arXiv preprint arXiv:1812.09551* (2018).

[33] Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian M. Sadler, Michelle T. Vanni, and Jiawei Han. 2018. TaxoGen: Constructing Topical Concept Taxonomy by Adaptive Term Embedding and Clustering. In *KDD*.

[34] Jieyu Zhang, Xiangchen Song, Ying Zeng, Jiaze Chen, Jiaming Shen, Yuning Mao, and Lei Li. 2021. Taxonomy Completion via Triplet Matching Network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4662–4670.

A IMPLEMENTATION DETAILS

A.1 Baseline Models

TaxoExpan, ARBORIST [12, 19] were designed for taxonomy expansion task. We follow the implementations in [34] to calculate the ranking of candidate positions from the single score output of their matching model, so that we can have similar output for evaluations. For TaxoExpan [19], we implemented the full framework with PGAT propagation method and InfoNCE Loss [16]. In comparison experiments in [34], all methods only leveraged the initial embeddings without any distribution models for matching model comparison. For TMN , we implemented with Raw embedding + LSTM and PGAT encoders for the full comparison, based on the original triplet matching network. And for GenTaxo, we used the same distribution model as in TaxoEnrich.

Note that in the previous methods, such as TaxoExpan, ARBORIST and TMN , the generation of the initial embeddings was from the static word embedding method. In MAG datasets, the embeddings of each concept node is computed using Word2Vec method to generate a 250-dimensional vectors. And in WordNet datasets, the embeddings were generated using FastText as a 300-dimensional vector.

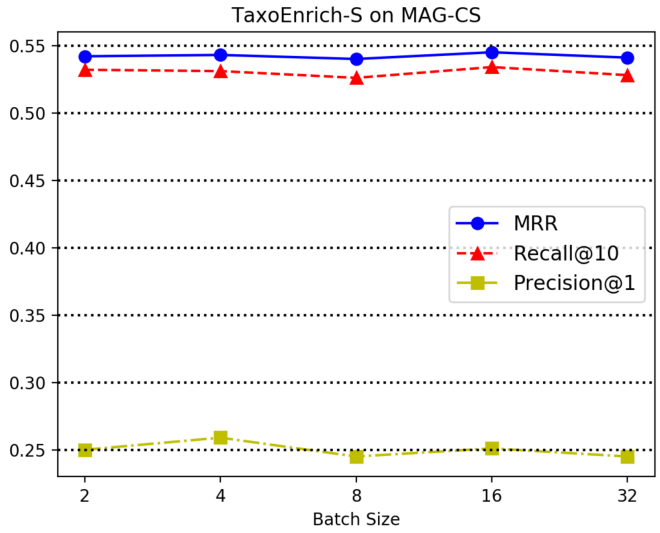
A.2 Hyperparameter Settings

In the implementation of TaxoEnrich, we use Adam optimizer [8] with learning rate 0.001. We applied a scheduler which multiplies the learning rate by a factor of 0.5 after 10 epochs of non-improving metrics. The hidden dimension for LSTM encoders is set as 500 and the number of bilinear models k in the matching module is 10. The number of siblings selected in the attention module t is set as 5 for training and 20 for testing. The model is then trained with 200 epochs with early stop if the MR metric has not improved for more than 10 epochs on validation dataset. For other hyperparameters, we set $l_1, l_2, l_3, l_4 = 1.0, 1.0, 1.0, 0.2$ in all datasets to avoid heavy parameter tuning, batch size as 16 for both TaxoEnrich and TaxoEnrich-S .

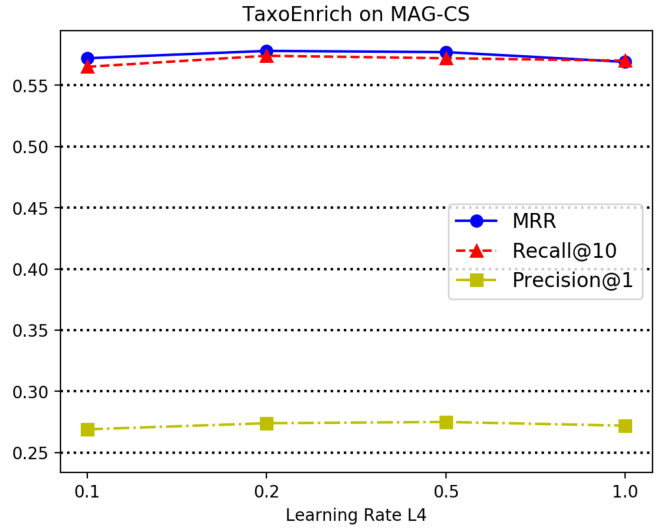
B ADDITIONAL ABLATION STUDIES

B.1 Hyperparameter Tuning

We conduct additional ablation studies on hyperparameter searching. We examine the influence of batch size of TaxoEnrich-S framework. It turns out that the batch size with 16 tends to be better than others.



Experiments on the learning rate of the newly incorporated sibling loss is also explored. We can observe that $l_4 = 0.2$ and 0.5 will result in slightly better performance for MAG-CS datasets, as the information in siblings will still introduce noises if we treat is equally with parent and children relatedness.



B.2 Sentence Encoder Studies

The comparison between different pretrained language models for sentence encoders is also studied under settings described above. And the results are shown in Table 7. We can see that SciBERT achieves the best performance among all language models, and Transformer has very similar results. And BERT has relatively poor performance. The reason may be that BERT is pretrained on general domain, making it less accurate in representing scientific domain-specific concepts in MAG-CS datasets.

Table 7: Ablation studies of TaxoEnrich-S on MAG-CS for the comparison between different pseudo sentence encoders.

Method	MAG-CS		
	MR	Recall@1	Precision@1
Transformer	74.132	0.149	0.248
SciBERT	**73.680**	**0.154**	**0.251**
BERT	253.221	0.082	0.173