QEN: Applicable Taxonomy Completion via Evaluating Full Taxonomic Relations

Suyuchen Wang
suyuchen.wang@umontreal.ca
RALI & Mila, Université de Montréal
Montréal, Québec, Canada

Ruihui Zhao
zacharyzhao@tencent.com
Tencent Jarvis Lab
Shenzhen, Guangdong, China

Yefeng Zheng
yefengzheng@tencent.com
Tencent Jarvis Lab
Shenzhen, Guangdong, China

Bang Liu*†
bang.liu@umontreal.ca
RALI & Mila, Université de Montréal
Montréal, Québec, Canada

ABSTRACT

Taxonomy is a fundamental type of knowledge graph for a wide range of web applications like searching and recommendation systems. To keep a taxonomy automatically updated with the latest concepts, the taxonomy completion task matches a pair of proper hypernym and hyponym in the original taxonomy with the new concept as its parent and child. Previous solutions utilize term embeddings as input and only evaluate the parent-child relations between the new concept and the hypernym-hyponym pair. Such methods ignore the important sibling relations, and are not applicable in reality since term embeddings are not available for the latest concepts. They also suffer from the relational noise of the "pseudo-leaf" node, which is a null node acting as a node's hyponym to enable the new concept to be a leaf node. To tackle the above drawbacks, we propose the **Q**uadruple **E**valuation **N**etwork (QEN), a novel taxonomy completion framework that utilizes easily accessible term descriptions as input, and applies pretrained language model and code attention for accurate inference while reducing online computation. QEN evaluates both parent-child and sibling relations to both enhance the accuracy and reduce the noise brought by pseudo-leaf. Extensive experiments on three real-world datasets in different domains with different sizes and term description sources prove the effectiveness and robustness of QEN on overall performance and especially the performance for adding non-leaf nodes, which largely surpasses previous methods and achieves the new state-of-the-art of the task.[1]

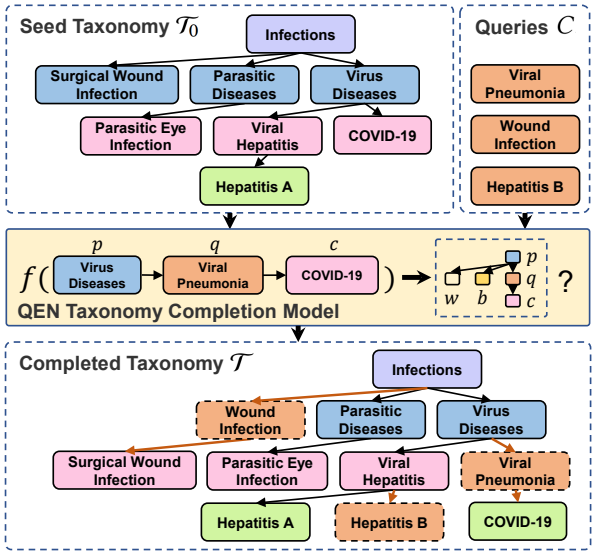


Figure 1: An illustration of the taxonomy completion task based on MeSH [14]. The QEN model is trained using the self-supervision data generated from the seed taxonomy. During inference, it utilizes both parent-child and sibling relations for scoring the fitness of adding a query node q into a candidate position (p, c).

CCS CONCEPTS

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

KEYWORDS

Taxonomy Completion, Self-supervised Learning, Taxonomic Relations

ACM Reference Format:
Suyuchen Wang, Ruihui Zhao, Yefeng Zheng, and Bang Liu. 2022. QEN: Applicable Taxonomy Completion via Evaluating Full Taxonomic Relations. 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/10.1145/3485447.3511943

1 INTRODUCTION

Taxonomy is a type of hierarchical knowledge graph that records the fundamental hypernym-hyponym or "*is-A*" relations among concepts, which is one of the most fundamental forms of knowledge graphs. It has been widely used in industrial web applications such

*Corresponding author.
†Canada CIFAR AI Chair
[1]The code is available at **https://github.com/sheryc/QEN**.

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

ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3511943

as e-commerce catalogues [12], government platforms [6], or academic web services like MeSH [14] or WordNet [23]. Furthermore, taxonomies are also critical in supporting various downstream tasks like recommendation systems [8, 16], web search [15, 36], and information extraction [12].

To construct and maintain an up-to-date taxonomy for downstream online applications, a core research problem is how to capture the newly emerging concepts on the web and insert them into the taxonomy without enormous human-labor and time. To solve this problem, a large number of previous researches focus on the task of *taxonomy expansion* [20, 27, 35, 38], where the model expands an existing *seed taxonomy* by finding the most proper hypernym for each new concept (or *query*) as its parent node in the expanded taxonomy. However, existing solutions for taxonomy expansion only model the parent-child relationship between a candidate parent node and a query. In this case, new queries can only be added as leaf nodes, which largely limits the application of the taxonomy in real scenarios.

To satisfy the need of adding non-leaf nodes to the taxonomy, recently, Zhang et al. [41] proposed the *taxonomy completion* task, which removes the restriction of the "hyponym-only" assumption in the taxonomy expansion task. Instead of finding a single hypernym for the query, taxonomy completion aims to first find a proper $< hypernym, hyponym >$ pair for each query, and then add the query as a node between the hypernym and hyponym in the completed taxonomy. For example, as shown in Fig. 1, to add the new concept *Viral Pneumonia* as a query into the seed taxonomy, one of the best $< hypernym, hyponym >$ pair should be <*Virus Diseases*, *COVID-19*>, where *Virus Diseases* and *COVID-19* are the most proper hypernym and hyponym for *Viral Pneumonia* in the seed taxonomy, respectively. Zhang et al. [41] formalized the taxonomy completion task into a ranking problem: the model ranks all possible $< hypernym, hyponym >$ pairs (or *candidate positions*) in the seed taxonomy for each query based on a matching score evaluating the hypernymy and hyponymy relations between the query and the candidate position. Currently, the Triplet Matching Network (TMN) proposed by Zhang et al. [41] is the only ranking-based taxonomy completion solution.

However, TMN has several drawbacks that limit both its performance and applicability. Firstly, when adding leaf nodes to the seed taxonomy (like *Hepatitis B* in Fig. 1), TMN uses a place-holding "pseudo-leaf" node as the hyponym in the candidate position. The pseudo-leaf has a fixed zero feature vector. The utilization of pseudo-leaf nodes introduces unfairness to the matching score ranking: if the hyponym is an actual node, the matching score reflects both meaningful hypernymy and hyponymy relation evaluation; but when the hyponym is the pseudo-leaf, the evaluation of the hyponymy relation becomes meaningless. Without other input features, this can be a pitfall when the seed taxonomy contains a large number of leaf nodes, since the model would tend to assign higher matching scores for candidate positions that contain pseudo-leaves, downgrading the task into a simpler taxonomy expansion task that only evaluates the hypernym. Secondly, the evaluation of a candidate position only considers the parent-child relation between the query and the candidate position, while the sibling relation is another important taxonomic relation that is empirically proved to improve taxonomy expansion performance [28, 35]. Finally, TMN adopts term embedding vectors trained from a large corpus as input features. However, new terms typically contain new words or new meaning for an existing word that is rarely shown in an existing corpus. Thus, it is hard to assign a meaningful term embedding to a new term, largely limiting the feasibility of TMN in real scenarios.

To overcome the flaws of the previous method, we propose the **Q**uadruple **E**valuation **N**etwork (QEN) for taxonomy completion. To be more specific, we make the following contributions: Firstly, instead of only evaluating the parent-child hypernymy relation between a query and its candidate position, we also evaluate the similarities between the query and two of the hypernym's children (see b and w in Fig. 1). These two children are the most and least semantically similar children of the hypernym with the query, which are representative samples of the query's potential siblings. The introducing of potential siblings can not only improve the model's performance by using full taxonomic relations, but also provide hyponym comparisons when the candidate position involves pseudo-leaf, since the model has the information to decide between adding a leaf or non-leaf node when the hypernym is proper. Therefore, QEN evaluates totally four different relations around the candidate position for inserting the query: the hypernym, the hyponym, and two potential siblings, making it a quadruple evaluation. More specifically, we design a *parent detector* and a *sibling detector* based on Transformer [33] and Multi-Layer Perceptron (MLP) architectures according to the characteristics of the two taxonomic relations.

Secondly, instead of using term embeddings as the input for taxonomy completion, QEN uses the more accessible term descriptions as input features, which enables the model to recognize the latest terms immediately by a short descriptive text of the term. The term descriptions may come from a term database like Wikipedia or MeSH [14], be generated by simple algorithms from dictionary databases like WordNet [23], or be constructed by oneself without reference to any database. In this paper, we conduct experiments on datasets covering all three term description sources. By utilizing the term descriptions as input, we can acquire more abundant semantic information of a new term even if it contains any new words or word meanings and lack of a sufficiently trained embedding vector.

Lastly, we are the first to use Pretrained Language Models (PLM) in the ranking-based taxonomy completion task. To reduce the online computation while maintaining the performance of PLMs, we use a code attention module along with PLMs to produce a fixed number of representations for each node. It moves the computation of seed taxonomy node's representations offline, and the online computation only requires one forward pass of PLM for the query, which largely reduces the online computation overhead. PLMs can produce contextualized term embeddings by extracting a term's meaning based on recognizable sentence patterns. Thus, PLMs are suitable for producing term embeddings for the taxonomy completion task.

We conduct extensive evaluations to compare QEN with the previous state-of-the-art ranking-based taxonomy completion solution and state-of-the-art taxonomy expansion solutions modified to fit the task. Our experiments are conducted on three real-world datasets in three different domains, with different taxonomy size and description source. The results suggest that QEN outperforms the baselines with a high robustness to taxonomy domain, dataset size and description source. QEN also can achieve a much smaller

performance gap for adding leaf vs. non-leaf nodes. We also perform ablation studies and evaluate the auxiliary tasks proposed in TMN, showing the importance of evaluating siblings and the effectiveness of the proposed quadruple evaluation module.

2 RELATED WORK

2.1 Taxonomy Construction and Expansion

Due to the key and basic role of taxonomy and ontology in industrial applications, the research for automatic construction or expansion of these hierarchically structured knowledge graphs began very early. Researchers first focused on automatic taxonomy construction, where a model needs to build a taxonomy with a term set from scratch. Existing models can construct either topic-based taxonomies [17, 26, 40] or instance-based taxonomies [3, 21, 29], with pattern matching solutions [7, 10, 24] or distributional solutions using neural network-based hypernymy detection models [4, 13, 34, 37].

However, constructing a new taxonomy from scratch every time when new concepts are discovered may bring inconsistency and results in computational wastes. Instead, a taxonomy requires consistently updating with newly emerging concepts while maintaining its original design. Thus, there is the taxonomy expansion task that aims to add new terms to an existing seed taxonomy without changing the original structure. A variety of methods were proposed in recent years for this task. For example, Arborist [20] explicitly considered the various meanings of the edges and suggested using attention-based weights for the model to recognize different edge semantics. TaxoExpan [27] enhanced node representations by the Egonet structure around the parent node. STEAM [38] tackled the problem with classification on mini-paths using the ensembling of distributional, contextual and lexical syntactic features. HEF [35] modeled the structure of Ego-tree structure with Transformers and pre-trained language models, and evaluated anchors by the product of four hierarchy-related scores. The concurrent work HyperExpan [19] further extended TaxoExpan by using hyperbolic graph neural nets for the Egonet structure. However, these approaches only model the parent-child edge relation between a single anchor node from the seed taxonomy and the query node, without considering adding the query as a seed node's parent. Thus, they can only add the query as leaf nodes into the taxonomy, which largely limits their application in reality.

2.2 Taxonomy Completion

More recently, Zhang et al. [41] introduced the taxonomy completion task, where a model needs to find both the parent and the child for a query in an existing seed taxonomy. The authors proposed the Triplet Matching Network (TMN) model to perform query-parent, query-child and query-parent-child comparisons with three neural tensor networks [30] as node matching modules, and trained the model by supervising the three matching results as auxiliary tasks. They modeled the taxonomy completion task into a hypernym-hyponym pair ranking task given a specific query. There are also other methods to add non-leaf nodes into the seed taxonomy. For example, TaxoOrder [31] computed the sequential order of inserting a set of query nodes as leaf nodes, thus some queries become non-leaf nodes in the completed taxonomy after the whole insertion process. TaxoGen [39] utilized a generative model to generate non-leaf concepts to be added. However, these models take concept vectors as input, which are not available for newly emerged concepts. The only ranking-based model TMN [41] added leaf nodes based on a pseudo-leaf with null embedding, which resulted in a large gap in the performance of adding leaf vs. non-leaf nodes. In our work, QEN uses term descriptions that are available for any new concepts as input, and reduces the leaf/non-leaf performance gap by introducing sibling relation information.

3 PROBLEM DEFINITION

The Definition of Taxonomy. A taxonomy $\mathcal{T} = (N, E)$ is defined as a directed acyclic graph (DAG), where N and E are the node set and edge set, respectively. Each node $n \in N$ is also called a *term* in this paper. Each term is associated with a surface name and a more informative representation for it, either an initial embedding vector $\boldsymbol{n}$ or a description text of the term X_n. Typically, taxonomies model hypernymy relations, where each directed edge $\langle p, c\rangle \in E$ suggests an implicit relation that p is the most specific superior or the most proper hypernym of c in this relation.

The Definition of the Taxonomy Completion Task. The task of taxonomy completion is to add a set of new terms C into an existing *seed taxonomy* $\mathcal{T}_0 = (N_0, E_0)$. To be more specific, for each *query* term $q \in C$, the model adds this term into the seed taxonomy by finding one or more suitable *candidate positions* (p, c) in the seed taxonomy to append q. The candidate positions are all valid (p, c) pairs where c is p's descendant. When adding q into (p, c), we remove the edge $\langle p, c\rangle$ if there is one, and form two new edges $\langle p, q\rangle$ and $\langle q, c\rangle$. To select the correct edges to add, we calculate a score $f(q, (p, c))$ by the taxonomy completion model indicating the fitness of adding q into (p, c). Following previous studies [27, 35, 38, 41], we do not consider relations among queries, and break the problem into C independent problems of adding a single query into $\mathcal{T}_0$. Note that p and c are not necessarily actual nodes in $\mathcal{T}_0$: When adding root/leaf nodes into $\mathcal{T}_0$, p or c can be a place-holding pseudo-root/pseudo-leaf node with a zero vector as its embedding or an empty string as its description. The task is illustrated in Fig. 1.

4 THE QUADRUPLE EVALUATION NETWORK

Our proposed Quadruple Evaluation Network (QEN) manages to prevent using hardly accessible word vectors for new concepts, integrates PLM to enhance the performance while reducing online computational cost, and distinguishes adding leaf or non-leaf nodes based on the evidence of sibling nodes. The illustration of the QEN model is shown in Fig. 2. The multilevel position matching selects the most representative potential siblings for evaluation; the semantically rich term representation can fully utilize PLMs to generate multiple term representations describing different aspects of the term; and the quadruple evaluation process evaluates multilevel full taxonomic relations, as well as reducing pseudo-leaf's relational noise by conducting placeholder replacement.

4.1 Multilevel Position Matching

To append a query term q to the most appropriate position in a taxonomy $\mathcal{T}_0$, QEN computes the matching score $f_{\text{QEN}}(q, (p, c))$ of the

query q and a candidate position (p, c): $f_{\text{QEN}} : N \times (N \times N) \to \mathbb{R}$ during training or $f_{\text{QEN}} : C \times (N \times N) \to \mathbb{R}$ during inference. We observed that in previous taxonomy completion methods [41], when adding q as a leaf node, c in the scoring function becomes a meaningless place-holding pseudo-leaf node. This approach does not offer any information to distinguish leaf or non-leaf node addition, and can usually become a pitfall when the taxonomy has large amount of leaf nodes. In contrast, we extend the input of the model to include some of the potential siblings of q in the expanded taxonomy. Introducing potential siblings can provide abundant information and features to the model for judging leaf/non-leaf node addition and better evaluating a candidate position.

More specifically, we select two nodes $b, w \in \text{child}_{\mathcal{T}_0}(p)$, which satisfies:

$$b = \underset{n \in \text{child}_{\mathcal{T}_0}(p)}{\arg\max} \ \text{CosSim}(n, q),$$
$$w = \underset{n \in \text{child}_{\mathcal{T}_0}(p)}{\arg\min} \ \text{CosSim}(n, q),$$

where $\text{CosSim}(\cdot, \cdot)$ is the cosine similarity of two terms' averaged Word2Vec [22] embeddings. We use a zero embedding for the terms that do not have a Word2Vec embedding to calculate cosine similarity. When $|\text{child}_{\mathcal{T}_0}(p)| = 0$, b and w become pseudo-leaf nodes and will be further processed in the quadruple evaluation module. These two potential siblings incorporate local structural information about p: when the best matching sibling b is a proper sibling of q, then the current selection of p is more likely correct. The siblings also provide evidence for adding leaf nodes: intuitively, q is better added as a leaf node of p instead of substituting one of p's children when p does not have any children (i.e., when b and w are pseudo-leaf nodes), or even the worst similar sibling w can also be a proper sibling of q (so that no child of p is a suitable child of q).

4.2 Semantically Rich Term Representation

Taxonomies typically need to include the latest concepts, which are likely not having existing term embeddings. However, each new concept can come with a natural language description that can be used for text matching. Thus, unlike existing methods that utilize in-domain word embeddings as node representations, QEN takes term descriptions as initial features. Utilizing descriptions instead of term embedding vectors as input also enables the model to use powerful pretrained language models [5, 25] to generate more informative node representations.

We first obtain the description text for each input term, denoted as X_k where $k \in \{q, p, c, b, w\}$. When a node n is the pseudo-leaf or does not exist, its description text becomes an empty string. We encode term descriptions by a pretrained language model (PLM, e.g., DistilBERT [25] in our implementation):

$$\boldsymbol{D}_k = \text{DistilBERT}(X_k) \in \mathbb{R}^{l_k \times d}, k \in \{q, p, c, b, w\},$$

where l_k is the length of the description text of term k, and d is the dimension of PLM's representations.

Maintaining a variable-length term representation is not memory-efficient. Thus, we adopt a code attention module, which is similar to the one used in the PolyEncoders [9]. To be more specific, we use m trainable codes $\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_m \in \mathbb{R}^d$, and attend them to the resulted PLM's representations to get the code representations for each input $\boldsymbol{C}_k \in \mathbb{R}^{m \times d}$:

$$\alpha^i_{e_j} = \frac{\exp\left(\boldsymbol{e}_j \boldsymbol{D}^{i,\cdot}_k\right)}{\sum_{h=1}^{l_k} \exp\left(\boldsymbol{e}_j \boldsymbol{D}^{h,\cdot}_k\right)},$$
$$\boldsymbol{C}^{j,\cdot}_k = \sum_{i=1}^{l_k} \alpha^i_{\boldsymbol{e}_j} \boldsymbol{D}^{i,\cdot}_k,$$

where $k \in \{q, p, c, b, w\}$. The code representation of each term is the semantically rich term representation, since they are knowledge-enriched with PLMs and contain multiple aspects of the term's meaning extracted by different codes. We emphasize that the computation of each node n's code representation $\boldsymbol{C}_n$ does not involve the information of other nodes, indicating that one can compute all the $\boldsymbol{C}_n$ for $n \in N_0$ offline. During inference, for a single query q, only $\boldsymbol{C}_q$ is required to compute once, which largely increases the efficiency of online inference. A comparison of inference time with/without the code attention module is shown in Appendix B.

4.3 The Quadruple Evaluation Process

After obtaining the code representations of query, parent, child, and siblings (i.e., $\boldsymbol{C}_q, \boldsymbol{C}_p, \boldsymbol{C}_c, \boldsymbol{C}_b$ and $\boldsymbol{C}_w$), we can perform the following two sets of evaluations, which cover both types of taxonomic relations [2]:

- **Parent-child relation evaluation.** This tests if (p, q) and (q, c) form hypernym-hyponym relations.
- **Sibling relation evaluation.** This tests if (q, b) and (q, w) share sibling-like similarities.

Therefore, the evaluation process detects totally four different relationships around q, which forms a quadruple evaluation.

Parent detector. We compute the parent-child relation representation $h_{p,q}, h_{q,c} \in \mathbb{R}^d$. When detecting parent-child relations, The ordering of the input is important. We adopt a two-layer Transformer encoder [33] for this module. We concatenate the first $m-1$ code representations of the two nodes into an input sequence, and take the first output (i.e., The output corresponding to the *CLS* token) of the Transformer. For example, to get the parental relation representation of node u, v, we have:

$$\boldsymbol{h}_{u,v} = \text{Transformer}\left(\left[\boldsymbol{e}_{CLS}, \boldsymbol{C}^{:m-1,\cdot}_u, \boldsymbol{C}^{:m-1,\cdot}_v\right]\right)^{0,\cdot},$$

where $[\cdot, \cdot]$ stands for concatenation, and $e_{CLS} \in \mathbb{R}^{1 \times d}$ is a trainable vector for the representation slot. The function $\text{Transformer}(\cdot)$ adds a sinusoidal positional embedding to each position, making it order-sensitive. Different from TMN [41], we unify the detection of (p, q)'s and (q, c)'s parent-child relations in a single module, reducing the redundancy of parameters.

Sibling detector. Detecting sibling relations is similar to the task of text matching, which also requires the position invariance of inputs. Therefore, we use a single layer feed-forward network to compute the sibling relation representation $s_{q,b}, s_{q,w} \in \mathbb{R}^d$ using the first code representation of each term:

$$\boldsymbol{s}_{u,v} = \boldsymbol{W}_s\left(\left[\boldsymbol{C}^{1,\cdot}_u, \boldsymbol{C}^{1,\cdot}_v, |\boldsymbol{C}^{1,\cdot}_u - \boldsymbol{C}^{1,\cdot}_v|\right]\right) + b_s,$$

where $\boldsymbol{W}_s$ and $\boldsymbol{b}_s$ are trainable parameters.

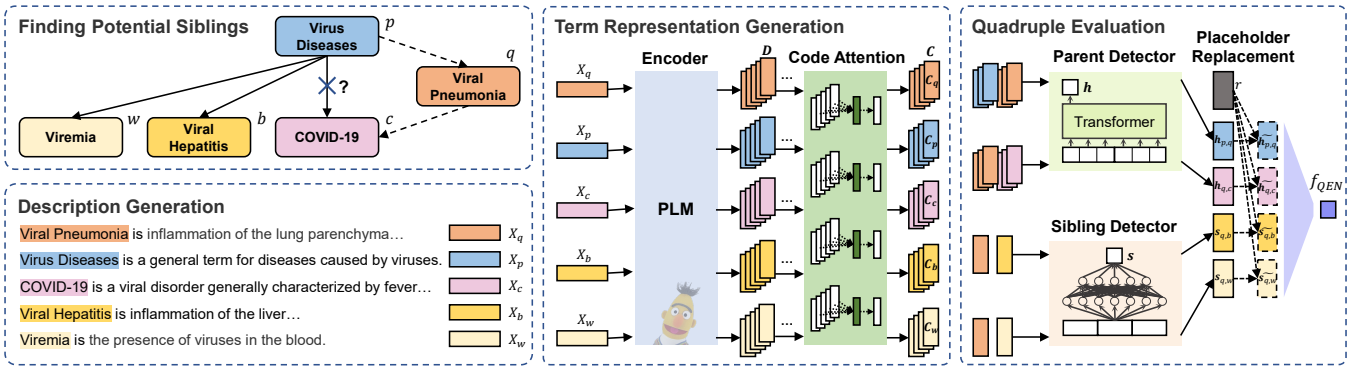


Figure 2: The architecture of the QEN model.

The scoring process. After obtaining the relation representations $\boldsymbol{h}_{p,q}, \boldsymbol{h}_{q,c}, \boldsymbol{s}_{q,b}$ and $\boldsymbol{s}_{q,w}$, we first exclude the relational noise brought by pseudo-leaf nodes. Keeping meaningless pseudo-leaf representations brings noise to the relation detection process and scoring, thus we perform stop-gradient during the scoring process by substituting the representations involving pseudo-leaf or nonexistent nodes with a trainable placeholder vector $\boldsymbol{r} \in \mathbb{R}^d$. This also makes the scoring process to take into account the leaf/non-leaf status of q (by the existence of p and c) and decide whether it is impossible to add a non-leaf node under p (by the existence of b and w),

$$\widetilde{\boldsymbol{h}_{p,q}} = \begin{cases} \boldsymbol{h}_{p,q}, & \text{if } p \in N_0, \\ \boldsymbol{r}, & \text{otherwise}, \end{cases}$$

$$\widetilde{\boldsymbol{h}_{q,c}} = \begin{cases} \boldsymbol{h}_{q,c}, & \text{if } c \in N_0, \\ \boldsymbol{r}, & \text{otherwise}, \end{cases}$$

$$\widetilde{\boldsymbol{s}_{q,b}} = \begin{cases} \boldsymbol{s}_{q,b}, & \text{if } b \in N_0, \\ \boldsymbol{r}, & \text{otherwise}, \end{cases}$$

$$\widetilde{\boldsymbol{s}_{q,w}} = \begin{cases} \boldsymbol{s}_{q,w}, & \text{if } w \in N_0, \\ \boldsymbol{r}, & \text{otherwise}, \end{cases}$$

The scoring function takes the concatenation of the four relation representations into a two-layer MLP and outputs the matching score between the query q and the candidate position (p, c):

$$f_{\text{QEN}}(q,(p,c)) = \boldsymbol{W}_{s2}\left(\tanh\left(\boldsymbol{W}_{s1}\left[\widetilde{\boldsymbol{h}_{p,q}}, \widetilde{\boldsymbol{h}_{q,c}}, \widetilde{\boldsymbol{s}_{q,b}}, \widetilde{\boldsymbol{s}_{q,w}}\right] + b_{s1}\right)\right) + b_{s2},$$

where $\boldsymbol{W}_{s1}, \boldsymbol{W}_{s2}, b_{s1}, b_{s2}$ are trainable parameters.

4.4 The Self-supervised Training Process

To avoid using extra labeled data for training, we train QEN by generating the training data from the seed taxonomy $\mathcal{T}_0$.

In a single epoch, each node in the seed taxonomy will be trained as the query node exactly once. For a single query $q \in N_0$, we generate one positive pair $\left(p_0^q, c_0^q\right)$ by selecting a random parent $p_0^q \in \text{parent}_{\mathcal{T}_0}(q)$ and a random child $c_0^q \in \text{child}_{\mathcal{T}_0}(q)$. We then generate N negative examples $\{(p_1^q, c_1^q), \cdots, (p_N^q, c_N^q)\}$, where $\forall j \in \{1, 2, \cdots, N\}, p_j^q \notin \text{parent}_{\mathcal{T}_0}(q) \vee c_j^q \notin \text{child}_{\mathcal{T}_0}(q)$ holds. In a single mini-batch, the batch size B determines the number of queries, where the number of training instances for each q is $N+1$. Thus, a single mini-batch can be denoted as $\mathcal{B} = \left\{\left(q_i, (p_j^{q_i}, c_j^{q_i}), y\right) | i \in \{1, \cdots, B\}, j \in \{0, \cdots\}\right.$, where $y \in \{0, 1\}$ is the training label for an instance. The stochastic training data generation ensures that the parent detector and sibling detector can learn the order dependence during training. Moreover, to reduce the impact of b and w not being the actual most similar and most dissimilar nodes during inference, we replace b and w with a random node from $\text{child}_{\mathcal{T}_0}(p_j^{q_i})$ with a 10% probability during training to improve the robustness.

The loss of the model is the binary cross-entropy on $f_{\text{QEN}}(q,(p,c))$:

$$\mathcal{L} = -\frac{1}{|\mathcal{B}|} \sum_{(q_i,(p_i,c_i),y_i)\in\mathcal{B}} y_i \log(f_{\text{QEN}}(q_i,(p_i,c_i))) + (1-y_i)\log(1 - f_{\text{QEN}}(q_i,(p_i,c_i))). \quad (1)$$

Table 1: The statistics of datasets. $|N|$, $|E|$, D, L are the node number, edge number, depth of the taxonomy and the number of leaf nodes in the taxonomy, respectively. The percentage of leaf nodes in the taxonomy is given in $L\%$.

Dataset	$\lvert N\rvert$	$\lvert E\rvert$	D	$\lvert L\rvert$	$L\%$
WordNet-Verb [11]	13,936	13,407	12	10,581	75.9%
MeSH [14]	9,710	10,498	10	6,613	68.1%
SemEval16-Food [1]	1,486	1,533	8	1,184	79.7%

5 EXPERIMENTS

In this section, we describe our extensive evaluations on three public datasets to compare QEN with a variety of baseline models, as well as perform ablation studies to show the characteristics of some modules of QEN. The implementation detail of the model is described in Appendix A.

5.1 Experimental Settings

5.1.1 Datasets and Input Features. The QEN model is evaluated on three public datasets from different domains in different scales with different types of description sources. The statistics of the datasets are summarized in Table 1.

WordNet-Verb. This dataset is from the verb taxonomy of SemEval-2016 Task 14 [11], which is the hierarchy of verbs from WordNet 3.0 [23]. The original SemEval task provides each term's description, which is used as QEN's input. Following [41], we use the 300-dimension fastText embedding[2] for the baselines.

MeSH. This dataset contains a subgraph of the Medical Subject Headings (MeSH) [14], which is a widely used biomedical indexing hierarchy. We adopt the scope note of each MeSH term's descriptor data as their description, and train a 300-dimension fastText embedding for the baselines on the PubMed corpus using the code[3] from BioWordVec [42].

SemEval-Food. This dataset derives from the food domain taxonomy of SemEval-2016 Task 13 [1], which is the largest taxonomy of the SemEval task. We use the dynamic programming description generation algorithm from Wang et al. [35] to generate term descriptions for QEN from WordNet, and use the fastText embeddings provided by Shen et al. [27] for all the baselines.

For *WordNet-Verb*, we randomly sample 1,000 nodes for validation and another 1,000 nodes for test, while for the other two, we sample 10% nodes for validation and 10% nodes for test. We build the seed taxonomy using the remaining nodes following [41], where we add extra edges that connect all original parents and children of each validation or test node to avoid the graph becoming disconnected.

5.1.2 Baselines for Comparison. We compare QEN with the following taxonomy expansion or taxonomy completion methods:

- **Bilinear Model** [32]. The model evaluates the candidate positions through a bilinear model.
- **Neural Tensor Network** [30]. The model evaluates the candidate position by the sum of a single-layer model, a biliear model, and a bias vector.
- **TaxoExpan** [27]. One of the state-of-the-art taxonomy expansion models that uses a graph neural net with the Egonet structure instead of the initial node embeddings to incorporate relative level in representations.
- **ARBORIST** [20]. One of the state-of-the-art taxonomy expansion models that emphasizes heterogeneous relations by using an attention-like weight for different edges and trains with large margin ranking loss.
- **TMN** [41]. The state-of-the-art model for ranking-based taxonomy completion, which derives the score using three auxiliary training tasks evaluating (q, p), (q, c) and $(q, (p, c))$ pairs based on neural tensor networks.

Since TMN is currently the only solution for ranking-based taxonomy completion task (when we submit this paper), we also add several state-of-the-art taxonomy expansion models for comparison, where the concatenation of parent and child's embedding would replace the parent's embedding in the original model. Following [41], we replace the GNN in TaxoExpan with the node's initial feature vector to keep the baselines' input aligned. Note that other taxonomy expansion models like STEAM [38] or HEF [35] requires either large extra corpus during training or a much larger amount of computational overhead, therefore these methods are not added to the baseline list. As for the more recent taxonomy completion model GenTaxo [39], it generates extra candidate queries via a generation model and does not use ranking-based metrics for evaluation, so we do not compare QEN with it either. Note that all the baseline methods take node's embedding vector as input, and the sources of embeddings are stated in the dataset section.

5.1.3 Evaluation Metrics. Assume $\left\{(\widehat{p_i^1}, \widehat{c_i^1}), \cdots, (\widehat{p_i^N}, \widehat{c_i^N})\right\}$ to be the list of candidate positions for test query $q_i \in C$ in a descending order of computed matching score where N is the total count of candidate positions, and $T_i := \left\{(p_i^1, c_i^1), \cdots, (p_i^{M_i}, c_i^{M_i})\right\}$ is the ground-truth candidate positions for q_i in the completed taxonomy. We use the following ranking-based criteria for evaluation.

- **Mean Rank (MR)**: It measures the macro average ranking of ground-truth positions among all candidate positions,

$$\text{MR} = \frac{1}{|C|}\sum_{i=1}^{|C|}\left(\frac{1}{M_i}\sum_{j=1}^{M_1}\text{rank}\left((p_i^j, c_i^j)\right)\right).$$

- **Mean Reciprocal Rank (MRR)**: It calculates the macro average reciprocal rank of all ground-truth positions,

$$\text{MRR} = \frac{1}{|C|}\sum_{i=1}^{|C|}\left(\frac{1}{M_i}\sum_{j=1}^{M_1}\frac{1}{\text{rank}\left((p_i^j, c_i^j)\right)}\right).$$

- **Recall@k (R@k)**: The number of ground-truth positions in the top k rankings averaged by the total count of ground-truth positions for all queries,

$$\text{R@}k = \frac{\sum_{i=1}^{|C|}\sum_{j=1}^{k} I_{T_i}\left(\widehat{p_i^j}, \widehat{c_i^j}\right)}{\sum_{i=1}^{|C|} M_i}.$$

  where $I_A(\cdot)$ is the indicator function for set A.
- **Precision@k (P@k)**: The number of ground-truth positions in the top k rankings averaged by k times the total number of queries,

$$\text{P@}k = \frac{\sum_{i=1}^{|C|}\sum_{j=1}^{k} I_{T_i}\left(\widehat{p_i^j}, \widehat{c_i^j}\right)}{k \times |C|}.$$

5.2 Main Results

The performance of QEN is shown in Table 2. QEN achieves the overall best performance on almost all metrics on all three datasets, except for MR on *WordNet-Verb*, where QEN achieves on-par performance with the baselines.

In Table 2, we also provide the results for adding leaf nodes and non-leaf nodes to the seed taxonomy. QEN achieves comparable or better performance with adding leaf nodes, which indicates that adding siblings does not sacrifice the performance when the input involves pseudo-leaf nodes as the hyponym. However, as for adding non-leaf nodes, which is the most essential scenario for the taxonomy completion task, the baselines can sometimes fail due to ranking only the leaf candidate positions high (e.g., TaxoExpan for *MeSH*). On the contrary, QEN significantly surpasses the baselines for non-leaf node addition on all metrics except MR for *WordNet-Verb* thanks to the extra information provided by the siblings. This

[2]We use the official wiki-news-300d-1M-subword.vec.zip version.
[3]https://github.com/ncbi-nlp/BioWordVec.

Table 2: Main experiment results on all three datasets. The best and second best results for each metric are marked by bold and underline, respectively. Metrics with a down arrow (↓) means lower is better, otherwise higher is better.

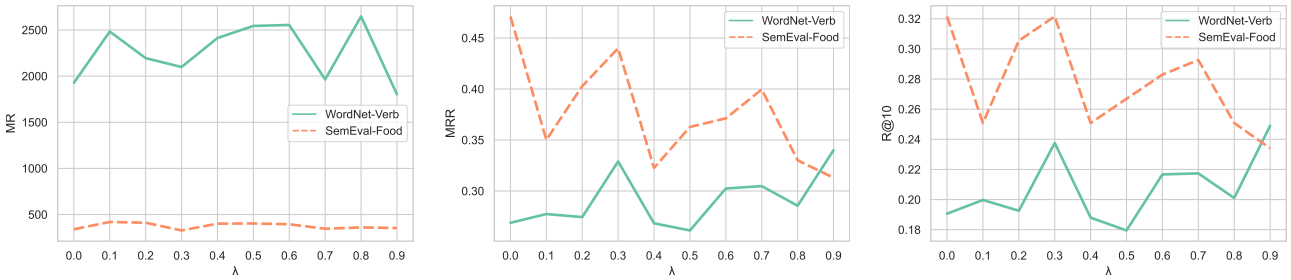
Methods	*WordNet-Verb*													
	Total								Leaf			Non-leaf		
	MR↓	MRR	R@1	R@5	R@10	P@1	P@5	P@10	MR↓	MRR	R@10	MR↓	MRR	R@10
Bilinear [32]	1861.303	0.174	0.012	0.052	0.095	0.018	0.016	0.014	888.546	0.247	0.140	5851.593	0.089	0.044
NTN [30]	1568.618	0.251	0.050	0.124	0.171	0.075	0.037	0.026	819.931	0.413	0.309	4639.764	0.067	0.013
TaxoExpan [27]	2023.849	0.231	0.053	0.122	0.168	0.080	0.037	0.025	1127.277	0.392	0.308	5701.622	0.048	0.007
ARBORIST [20]	**1499.396**	0.238	0.033	0.096	0.149	0.049	0.028	0.023	838.689	0.315	0.204	**4209.641**	0.149	0.086
TMN [41]	1510.165	0.291	0.066	0.154	0.207	0.099	0.047	0.031	**751.151**	0.439	0.342	4623.670	0.121	0.052
QEN	1802.404	**0.340**	**0.081**	**0.186**	**0.249**	**0.124**	**0.057**	**0.038**	1055.865	**0.495**	**0.407**	4909.494	**0.166**	**0.093**

Methods	*MeSH*													
	Total								Leaf			Non-leaf		
	MR↓	MRR	R@1	R@5	R@10	P@1	P@5	P@10	MR↓	MRR	R@10	MR↓	MRR	R@10
Bilinear [32]	985.226	0.273	0.038	0.115	0.173	0.086	0.052	0.039	483.021	0.395	0.284	2064.968	0.192	0.100
NTN [30]	702.321	0.329	0.064	0.167	0.227	0.143	0.075	0.051	408.168	0.542	0.454	1334.750	0.189	0.077
TaxoExpan [27]	6784.300	0.173	0.024	0.085	0.123	0.053	0.028	0.038	466.745	0.434	0.310	20367.045	0.001	0
ARBORIST [20]	800.810	0.343	0.035	0.141	0.229	0.078	0.063	0.051	515.872	0.420	0.310	1413.425	0.292	0.175
TMN [41]	494.311	0.410	0.061	0.197	0.291	0.137	0.088	0.065	**401.702**	**0.555**	**0.459**	693.421	0.315	0.180
QEN	**344.735**	**0.423**	**0.071**	**0.198**	**0.294**	**0.165**	**0.091**	**0.066**	511.933	0.548	0.427	**573.013**	**0.322**	**0.187**

Methods	*SemEval-Food*													
	Total								Leaf			Non-leaf		
	MR↓	MRR	R@1	R@5	R@10	P@1	P@5	P@10	MR↓	MRR	R@10	MR↓	MRR	R@10
Bilinear [32]	700.067	0.140	0.024	0.096	0.110	0.050	0.039	0.022	269.891	0.305	0.244	2816.532	0.005	0
NTN [30]	685.409	0.192	0.037	0.102	0.148	0.074	0.041	0.030	241.654	0.422	0.328	2868.683	0.005	0
TaxoExpan [27]	688.704	0.207	0.041	0.101	0.166	0.083	0.041	0.034	255.644	0.455	0.368	2819.359	0.004	0
ARBORIST [20]	700.794	0.129	0.013	0.053	0.088	0.027	0.022	0.018	260.377	0.280	0.195	2867.647	0.005	0
TMN [41]	559.805	0.221	0.037	0.113	0.160	0.074	0.046	0.032	**179.455**	0.482	0.356	2431.128	0.007	0
QEN	**353.733**	**0.313**	**0.070**	**0.176**	**0.234**	**0.146**	**0.074**	**0.049**	213.234	**0.600**	**0.489**	**1044.990**	**0.094**	**0.038**

Table 3: Results of ablation studies. In "w/o r", pseudo-leaf's relation representation is not replaced with the placeholder; In "w/o b, w", the siblings are removed. The best results are in bold.

Settings	*WordNet-Verb*				*MeSH*				*SemEval-Food*			
	MR↓	MRR	R@10	P@10	MR↓	MRR	R@10	P@10	MR↓	MRR	R@10	P@10
QEN	1802.404	**0.340**	**0.249**	**0.038**	**344.735**	**0.423**	**0.294**	**0.066**	**353.733**	**0.313**	**0.234**	**0.049**
w/o r	**1718.186**	0.324	0.246	**0.038**	447.067	0.389	0.261	0.060	384.202	0.289	0.219	0.046
w/o b, w	4315.287	0.115	0.068	0.010	1432.005	0.167	0.097	0.023	568.423	0.247	0.179	0.038



(a) MR on the two datasets. (b) MRR on the two datasets. (c) Recall@10 on the two datasets.

Figure 3: Sensitivity analysis of model performance under different multi-task learning weight η.

shows the importance of evaluating both parent-child and sibling relations in the taxonomy completion task.

The three datasets cover different sources of description texts, different domains, and different dataset sizes. For *WordNet-Verb*, QEN can make better inferences for the majority of queries, where the model makes worse inference for only a small fraction of queries. It is probably due to the relatively short description texts for this dataset, where the average length of descriptions is merely 9.21 words. However, even with a relatively short description, QEN can still surpass other baselines, which shows the robustness of QEN against brief descriptions.

For *MeSH*, the descriptions are all texts in the biomedical domain with a longer average description length of 31.71 words from the MeSH database. However, although we adopt a PLM pretrained in the general domain as the encoder, QEN can still achieve state-of-the-art results on an out-of-domain dataset, indicating its cross-domain transfer learning robustness.

The *SemEval-Food* dataset is the smallest dataset, which does not provide enough self-supervision data. However, QEN utilizes PLMs, and it effectively reduces the requirement for training data, achieving a significantly better performance compared to the baselines. It is worth noticing that for adding non-leaf nodes to *SemEval-Food*, QEN is the only valid model that achieves a non-zero R@10 and a drastically higher MRR compared to the baselines, which indicates QEN's superior performance when the taxonomy to complete is small.

5.3 Ablation Studies

We also conducted ablation studies. In this section, We remove two of the key designs in QEN. In "w/o r", we remove the stop-gradient mechanism during the quadruple evaluation by not performing placeholder replacement. In "w/o b, w", we do not include the siblings for evaluation and only evaluate parent-child relations for scoring. The results are shown in Table 3.

When removing the placeholder r, the performance of QEN slightly degrades on all datasets. This empirically proves that the relation representations involving pseudo-leaf is a training noise for the model and thus harms performance. When removing the siblings b and w, the model loses the extra information to compare with the pseudo-leaf, which substantially harms the model's performance. This suggests that adding information about the potential siblings is an effective way to reduce the impact of meaningless relations involving the pseudo-leaf node.

5.4 Adding Auxiliary Tasks

Zhang et al. [41] proposed to regularize each of the relation representations as auxiliary tasks to the training process. Therefore, we also test QEN's performance under this setting. To be more specific, in this part, we make the following changes to QEN:

- Add a distinctive single-layer MLP for each relation representation to generate scores for each node-pair relation:

$$h_r = \boldsymbol{W}_r r + \boldsymbol{b}_r,$$

  where $r \in \left\{\widetilde{\boldsymbol{h}_{p,q}}, \widetilde{\boldsymbol{h}_{q,c}}, \widetilde{\boldsymbol{s}_{q,b}}, \widetilde{\boldsymbol{s}_{q,w}}\right\}$, and $\boldsymbol{W}_r, \boldsymbol{b}_r$ are trainable parameters.

- Add additional regularization binary cross-entropy loss to the original loss function (Eqn. 1) based on the previously computed scores:

$$\mathcal{L}_{\text{aux}} = \mathcal{L} + \lambda \sum_{r \in \left\{\widetilde{\boldsymbol{h}_{p,q}}, \widetilde{\boldsymbol{h}_{q,c}}, \widetilde{\boldsymbol{s}_{q,b}}, \widetilde{\boldsymbol{s}_{q,w}}\right\}} \text{BCELoss}(h_r, y_r) \quad (2)$$

  where λ is the regularization weight, $y_r \in \{0, 1\}$ is the label for each node-pair relation involved in the quadruple evaluation, and $\text{BCELoss}(\cdot, \cdot)$ is the binary cross-entropy loss for the mini-batch. When $\lambda = 0$, the model becomes the original QEN.

We change the value of λ and report the MR, MRR and R@10 on *WordNet-Verb* and *SemEval-Food*. The results are shown in Fig. 3.

The result suggests that setting a larger λ can sometimes achieve a higher MRR and R@10 for the *WordNet-Verb* dataset. However, on the contrary, adding auxiliary tasks can result in a much lower MRR and R@10 for the *SemEval-Food* dataset. Thus, using auxiliary tasks does not guarantee a better performance for QEN, and the impact of auxiliary tasks depends on the dataset. Furthermore, when using auxiliary tasks, the result does not have a consistent trend when tuning the regularization weight λ. In practice, we would suggest testing with $\lambda = 0$ (the original QEN) or tuning λ for each new dataset for the best performance.

6 CONCLUSION

In this paper, we presented the Quadruple Evaluation Network (QEN), a novel architecture for the taxonomy completion task. It utilizes text descriptions with pretrained language models for learning term representations, and evaluates both parent-child relations and sibling relations for matching the query term with each positions in the taxonomy. To make it applicable for recognizing the latest concepts, we obtained a semantically rich term representation with easily accessible term descriptions, and achieved fast and accurate online inference by pretrained language model and the code attention module. To prevent the relational noise brought by the pseudo-leaves, we proposed multilevel position matching and the quadruple evaluation that evaluates full taxonomic relations. Extensive experiments on real-world datasets of different sizes in different domains demonstrated the effectiveness and robustness of QEN, especially for distinguishing leaf and non-leaf nodes. The results showed that performing quadruple evaluation on both parent-child and sibling relations is an effective way to reduce pseudo-leaf's relational noise, and the rich description based on term descriptions can be more robust and better-performing with a more accessible input. Potential future works include designing a better paradigm for the taxonomy completion task that bypasses the usage of null pseudo-leaf nodes, and incorporating taxonomy completion with taxonomy-based downstream tasks.

ACKNOWLEDGMENTS

This research was partially supported by the Canada CIFAR AI Chair Program and the NSERC Discovery Grant RGPIN-2021-03115.

REFERENCES

[1] Georgeta Bordea, Paul Buitelaar, Stefano Faralli, and Roberto Navigli. 2015. SemEval-2015 Task 17: Taxonomy Extraction Evaluation (TExEval). In *Proceedings of the 9th International Workshop on Semantic Evaluation*. Denver, Colorado, 902–910.

[2] Anita Burgun and Olivier Bodenreider. 2001. Aspects of the Taxonomic Relation in the Biomedical Domain. In *Proceedings of the International Conference on Formal Ontology in Information Systems-Volume*. 222–233.

[3] Anne Cocos, Marianna Apidianaki, and Chris Callison-Burch. 2018. Comparing Constraints for Taxonomic Organization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana, 323–333.

[4] Sarthak Dash, Md. Faisal Mahbub Chowdhury, Alfio Gliozzo, Nandana Mihindukulasooriya, and Nicolas Rodolfo Fauceglia. 2020. Hypernym Detection Using Strict Partial Order Networks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*. 7626–7633.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*. Minneapolis, Minnesota, 4171–4186.

[6] Jingyue Gao, Yuanduo He, Yasha Wang, Xiting Wang, Jiangtao Wang, Guangju Peng, and Xu Chu. 2019. STAR: Spatio-Temporal Taxonomy-Aware Tag Recommendation for Citizen Complaints. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. Beijing, China, 1903–1912.

[7] Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *The 14th International Conference on Computational Linguistics*.

[8] Jin Huang, Zhaochun Ren, Wayne Xin Zhao, Gaole He, Ji-Rong Wen, and Daxiang Dong. 2019. Taxonomy-Aware Multi-Hop Reasoning Networks for Sequential Recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. Melbourne, VIC, Australia, 573–581.

[9] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: Architectures and Pre-Training Strategies for Fast and Accurate Multi-sentence Scoring. In *8th International Conference on Learning Representations*. Addis Ababa, Ethiopia.

[10] 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. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Halifax, NS, Canada, 877–886.

[11] David Jurgens and Mohammad Taher Pilehvar. 2016. SemEval-2016 Task 14: Semantic Taxonomy Enrichment. In *Proceedings of the 10th International Workshop on Semantic Evaluation*. San Diego, California, 1092–1102.

[12] Giannis Karamanolakis, Jun Ma, and Xin Luna Dong. 2020. TXtract: Taxonomy-Aware Knowledge Extraction for Thousands of Product Categories. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online, 8489–8502.

[13] Dekang Lin. 1998. An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*. San Francisco, CA, USA, 296–304.

[14] Carolyn E. Lipscomb. 2000. Medical Subject Headings (MeSH). *Bulletin of the Medical Library Association* 88, 3 (2000), 265–266.

[15] Bang Liu, Weidong Guo, Di Niu, Jinwen Luo, Chaoyue Wang, Zhen Wen, and Yu Xu. 2020. GIANT: Scalable Creation of a Web-Scale Ontology. In *Proceedings of the 2020 International Conference on Management of Data*. 393–409.

[16] Bang Liu, Weidong Guo, Di Niu, Chaoyue Wang, Shunnan Xu, Jinghong Lin, Kunfeng Lai, and Yu Xu. 2019. A User-Centered Concept Mining System for Query and Document Understanding at Tencent. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*. 1831–1841.

[17] Xueqing Liu, Yangqiu Song, Shixia Liu, and Haixun Wang. 2012. Automatic Taxonomy Construction From Keywords. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Beijing, China, 1433–1441.

[18] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations*. New Orleans, LA, USA.

[19] Mingyu Derek Ma, Muhao Chen, Te-Lin Wu, and Nanyun Peng. 2021. HyperExpan: Taxonomy Expansion with Hyperbolic Representation Learning. *ArXiv preprint* abs/2109.10500.

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

[21] Yuning Mao, Xiang Ren, Jiaming Shen, Xiaotao Gu, and Jiawei Han. 2018. End-to-End Reinforcement Learning for Automatic Taxonomy Induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia, 2462–2472.

[22] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*. 3111–3119.

[23] George A. Miller. 1992. WordNet: A Lexical Database for English. In *Speech and Natural Language: Proceedings of a Workshop*. Harriman, New York, USA.

[24] 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)*. Melbourne, Australia, 358–363.

[25] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. *ArXiv preprint* abs/1910.01108.

[26] Jingbo Shang, Xinyang Zhang, Liyuan Liu, Sha Li, and Jiawei Han. 2020. NetTaxo: Automated Topic Taxonomy Construction from Text-Rich Network. In *The Web Conference 2020*. Taipei, Taiwan, 1908–1919.

[27] 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 *The Web Conference 2020*. Taipei, Taiwan, 486–497.

[28] Jiaming Shen, Zeqiu Wu, Dongming Lei, Chao Zhang, Xiang Ren, Michelle T. Vanni, Brian M. Sadler, and Jiawei Han. 2018. HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. London, UK, 2180–2189.

[29] Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. Semantic Taxonomy Induction from Heterogenous Evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. Sydney, Australia, 801–808.

[30] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26*. 926–934.

[31] 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* abs/2104.03682.

[32] Ilya Sutskever, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. 2009. Modelling Relational Data using Bayesian Clustered Tensor Factorization. In *Advances in Neural Information Processing Systems 22*. 1821–1828.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*. Long Beach, CA, USA, 5998–6008.

[34] Chengyu Wang, Yan Fan, Xiaofeng He, and Aoying Zhou. 2019. A Family of Fuzzy Orthogonal Projection Models for Monolingual and Cross-lingual Hypernymy Prediction. In *The World Wide Web Conference*. San Francisco, CA, USA, 1965–1976.

[35] Suyuchen Wang, Ruihui Zhao, Xi Chen, Yefeng Zheng, and Bang Liu. 2021. Enquire One's Parent and Child Before Decision: Fully Exploit Hierarchical Structure for Self-Supervised Taxonomy Expansion. In *Proceedings of the Web Conference 2021*. New York, NY, USA, 3291–3304.

[36] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Qili Zhu. 2012. Probase: a probabilistic taxonomy for text understanding. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. Scottsdale, AZ, USA, 481–492.

[37] Wenpeng Yin and Dan Roth. 2018. Term Definitions Help Hypernymy Detection. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*. New Orleans, Louisiana, 203–213.

[38] Yue Yu, Yinghao Li, Jiaming Shen, Hao Feng, Jimeng Sun, and Chao Zhang. 2020. STEAM: Self-Supervised Taxonomy Expansion with Mini-Paths. In *The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Virtual Event, CA, USA, 1026–1035.

[39] Qingkai Zeng, Jinfeng Lin, Wenhao Yu, Jane Cleland-Huang, and Meng Jiang. 2021. Enhancing Taxonomy Completion with Concept Generation via Fusing Relational Representations. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. New York, NY, USA, 2104–2113.

[40] Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian M. Sadler, Michelle Vanni, and Jiawei Han. 2018. TaxoGen: Unsupervised Topic Taxonomy Construction by Adaptive Term Embedding and Clustering. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. London, UK, 2701–2709.

[41] Jieyu Zhang, Xiangchen Song, Ying Zeng, Jiaze Chen, Jiaming Shen, Yuning Mao, and Lei Li. 2021. Taxonomy Completion via Triplet Matching Network. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 5 (2021), 4662–4670.

[42] Yijia Zhang, Qingyu Chen, Zhihao Yang, Hongfei Lin, and Zhiyong Lu. 2019. BioWordVec, Improving Biomedical Word Embeddings With Subword Information and MeSH. *Scientific Data* 6, 1 (2019), 1–9.

Table 4: Comparison of average online inference time per query for all three datasets. The better results are in bold.

	WordNet-Verb	*MeSH*	*SemEval-Food*
QEN w/ code-attention	**0.87s**	**0.65s**	**0.13s**
QEN w/ cross-attention	1.22s	1.94s	1.19s

A IMPLEMENTATION DETAILS

We use the official distilbert-base-uncased [25] as the sequence encoding module for lower memory usage. For QEN, we train the model by AdamW optimizer [18] using a cosine learning rate scheduler, with the learning rate linearly warm-up from 0 to 5×10^{-5} in the first 10% training steps. The negative sampling size is 15, and the mini-batch size is 8 with gradient accumulation for every 16 mini-batches. For the code attention module, it outputs 16 256-dimensional representations for a single input. The dropout rate is set to 0.1 for all modules. We train 50 epochs for *SemEval-Food* due to the small dataset size, and 30 epochs for the other two datasets. The hyperparameters for QEN are tuned on *WordNet-Verb* and are used across all datasets. For the baselines, we adopt the recommended hyperparameters given by [41], except for *SemEval-Food* that we train an extra 50 epochs for fairness. All the experiments are completed on a single NVIDIA RTX 3090 24GB or on a single NVIDIA Tesla V100 32GB.

B INFERENCE TIME

In this section, we compare the online inference time per query with and without the code attention module introduced in Section 4.2 using a single NVIDIA RTX 3090 24GB, to show how the overall design of QEN can reduce the online computation cost. The results are shown in Table 4. In this table, "QEN w/ code-attention" is the original QEN architecture, while in "QEN w/ cross-attention", the relation representations $\boldsymbol{h}_{p,q}, \boldsymbol{h}_{q,c}, \boldsymbol{s}_{q,b}$ and $\boldsymbol{s}_{q,w}$ are the output representation corresponding to the *CLS* token from DistilBERT, whose input is the concatenation of the two corresponding term descriptions. In the "QEN w/ cross-attention" setting, the code attention module, parent detector and sibling detector are all discarded since the relation representations can be directly obtained from PLM.

Although "QEN w/ cross-attention" is a simpler architecture compared to QEN, the cross attention in PLM involving both seed nodes and the query makes it hard to precompute any term representations offline. Therefore, the code-attention module is effective to make QEN more applicable, especially for larger taxonomies.

In addition, notice that the inference time reduction for *SemEval-Food* and *MeSH* is larger compared to that of *WordNet-Verb*. Because of the quadratic computation time of Transformer, datasets that have longer description text would benefit more from QEN's design.