CodeTaxo: Enhancing Taxonomy Expansion with Limited Examples via Code Language Prompts

Qingkai Zeng[1]*, **Yuyang Bai**[1], **Zhaoxuan Tan**[1], **Zhenyu Wu**[1], **Shangbin Feng**[2], **Meng Jiang**[1]
[1]University of Notre Dame [2] University of Washington
{qzeng, ybai3, ztan3, zwu23, mjiang2}@nd.edu, shangbin@cs.washington.edu

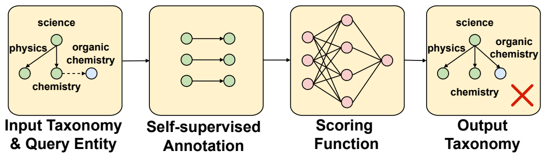
Abstract

Taxonomies provide structural representations of knowledge and are crucial in various applications. The task of taxonomy expansion involves integrating emerging entities into existing taxonomies by identifying appropriate parent entities for these new query entities. Previous methods rely on self-supervised techniques that generate annotation data from existing taxonomies but are less effective with small taxonomies (fewer than 100 entities). In this work, we introduce CODETAXO, a novel approach that leverages large language models through code language prompts to capture the taxonomic structure. Extensive experiments on five real-world benchmarks from different domains demonstrate that CODETAXO consistently achieves superior performance across all evaluation metrics, significantly outperforming previous state-of-the-art methods. The code and data are available at `https://github.com/QingkaiZeng/CodeTaxo-official`.

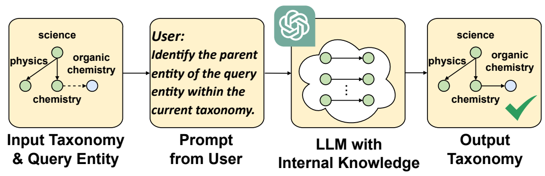
1 Introduction

Taxonomies are hierarchical structures encoding hypernym–hyponym (i.e., "is-A") relations between concepts or entities. Relational knowledge derived from taxonomies has been widely leveraged to identify semantic relevance for web search (Yin and Shah, 2010; Liu et al., 2020; Kang et al., 2024), personalized recommendation (Zhang et al., 2014; Tan et al., 2022; Huang et al., 2019), and question answering (Yang et al., 2017). However, existing taxonomies are mainly constructed by experts or through crowd-sourcing, making the process time-consuming, labor-intensive, and restricted in coverage (Bordea et al., 2016; Jurgens and Pilehvar, 2016). As new entities emerge, continually enriching taxonomies with these additions becomes vital. To address these challenges, taxonomy expansion aims to integrate new entities into existing taxonomies automatically.

*Corresponding author.



(a) **Discriminative Methods:** A trained scoring function selects the most appropriate parent entity from the taxonomy for a given query entity.



(b) **Generative Methods:** LLMs generate the parent entity from the taxonomy based on the query entity.

Figure 1: Two Types Taxonomy Expansion Methods

As shown in Figure 1a, recent taxonomy expansion methods mainly rely on discriminative methods that model hierarchical structures through techniques like Egonets (Shen et al., 2020), mini-paths (Yu et al., 2020), and Ego-Trees (Wang et al., 2021). Although pre-trained language models (PLMs) enhance these methods by encoding entities' textual descriptions (Wang et al., 2021, 2022; Liu et al., 2021b; Xu et al., 2022), their reliance on limited self-supervised annotations often restricts performance. In contrast, Generative Large Language Models (LLMs) such as GPT-4 (Achiam et al., 2023) and Llama family (Touvron et al., 2023; Dubey et al., 2024) have recently shown remarkable capabilities in text comprehension and generation, making them highly effective for tasks aimed at generating structural knowledge (Ye et al., 2022; Bi et al., 2024; Sun et al., 2024a,b). Increasing LLM parameters boosts generalization, surpassing smaller models and enabling superior few-shot or zero-shot performance. Even with limited annotations, LLMs effectively leverage extensive knowledge embedded within their parameters, acquired from large-scale pre-training corpora. In Figure 1b, we illustrate the pipeline to how generative methods are applied to the taxonomy expansion task.

Findings of the Association for Computational Linguistics: ACL 2025, pages 4131–4144
July 27 - August 1, 2025

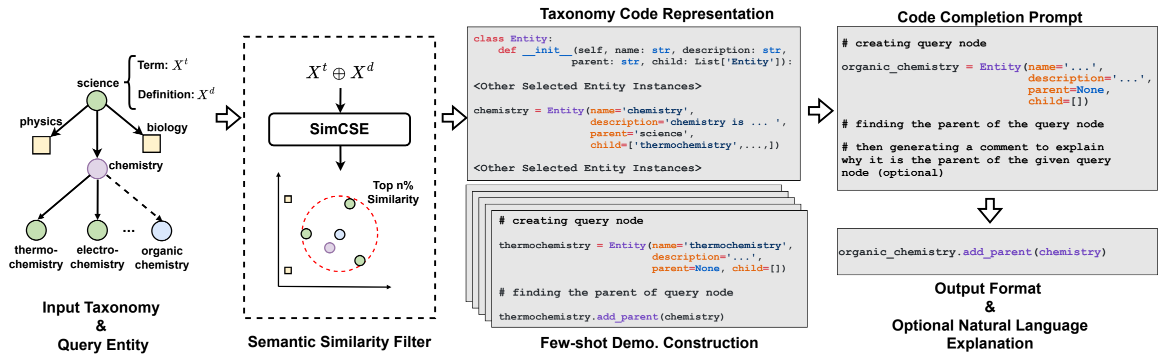


Figure 2: The overview of the pipeline for CODETAXO: CODETAXO reformulates the task of integrating a query entity q into an existing taxonomy $\mathcal{T}_0$ as a code completion task using code-based prompts for LLMs.

Two key challenges arise when applying LLMs to taxonomy expansion. First, unlike traditional text-to-text NLP tasks such as question answering and machine translation, representing the taxonomic structure for this task in natural language is inherently challenging. Specifically, the process requires "flattening" the taxonomy into a sequence of parent-child entity pairs (Madaan et al., 2022), effectively serializing a hierarchical structure into linear text. This serialized format is notably different from the unstructured text that LLMs primarily encounter during pre-training. Furthermore, while semantically related words in natural language are usually located near each other, linearizing a taxonomy can separate conceptually related entities by significant distances within the sequence. This disparity adds to the difficulty of aligning LLM outputs with the desired structured representation. Second, scaling to large taxonomies amplifies the problem, as including every entity from the existing taxonomy in the prompt is infeasible. The limited contextual window size of current LLMs and the associated computational overhead imposes strict constraints. Even if it were possible to include thousands of entities within a prompt, the resulting structural information loss would impair the clarity of entity-specific distinctions, reducing the model's capacity to effectively utilize the taxonomy.

To overcome these challenges, we propose CODETAXO, a novel taxonomy expansion approach that leverages code language as prompts. Code-based representations have shown promise in structure prediction tasks (Madaan et al., 2022; Li et al., 2023; Wang et al., 2023; Li et al., 2024; Bi et al., 2024), as code languages provide a more natural format for structural data. In CODETAXO, we frame taxonomy expansion as a code completion task. We introduce a base `Entity` class to store entity surface names, definitions, parent references, and child lists, along with two methods for modifying the taxonomic relations between entities. Each existing taxonomy entity is instantiated as a corresponding `Entity` object. Due to constraints of contextual window size, we apply a similarity-based filter, using SimCSE (Gao et al., 2021) to encode textual description for entities, to include only the most relevant entities in the prompt

We evaluate CODETAXO through extensive experiments on two sets of small-scale WordNet and Graphine sub-taxonomies (Bansal et al., 2014; Liu et al., 2021a), as well as three large-scale SemEval-2016 taxonomies (Bordea et al., 2016). Our one-shot CODETAXO surpasses all self-supervised baselines trained on large-scale SemEval-2016 annotations, achieving relative accuracy improvements of 10.26%, 8.89%, and 9.21% on SemEval-Sci, SemEval-Env, and SemEval-Food, respectively. Additionally, we evaluated CODETAXO using various open-source LLMs, revealing several interesting observations discussed in this work.

In summary, our main contributions include:

- We introduce CODETAXO, an innovative in-context learning method that utilizes code language prompts to represent taxonomic relationships between entities, thereby improving the effectiveness of taxonomy expansion.
- We develop a similarity-based filter, which employs a small pre-trained model to encode the textual descriptions of entities, ensuring that only highly relevant entities are included in the prompt concerning the query entity.
- Extensive experiments demonstrate that CODETAXO significantly enhances the performance of taxonomy expansion across two sets of small-scale sub-taxonomies and three large-scale taxonomies.

Task Instruction

Complete the next line of code according to the comments and the given code snippet. You need to find the parent of the query node in the given current taxonomy and use the add_parent function. (a)

The parent of the given query node always exists in the given current taxonomy, so do NOT generate a node that is NOT in the given current taxonomy. Note that you only need to complete the next ONE line of code a one-line explanation to explain why it is the parent node of the given query node, DO NOT generate any additional content. (b)

Taxonomy Code Representation

<Entity class defined in Section 3.2>

FOR EACH $e \in \mathcal{E}_0$:

e = Entity(name=X_e^t, description=X_e^d, parent=p_e, child=$\mathcal{C}_e$)

where the textual description of e is $\mathcal{X}_e = [X_e^t, X_e^d]$, p_e is the parent of e, and $\mathcal{C}_e = [c_e^1, c_e^2, \ldots, c_e^n]$ is the list of children of e.

Few-shot Demo.

Demo. $\mathcal{D} = [d_1, \cdots, d_n]$

Code Completion Prompt & Output

 # creating query node

q = Entity(name= X_q^t, description= X_q^d, parent=None, child=[])

 # Finding the parent of query node

q.add_parent(a)

Figure 3: Prompt Overview of CODETAXO

2 Problem Definition

Definition 1 (Taxonomy) *We follow the definition of taxonomy in (Jiang et al., 2023). A taxonomy $\mathcal{T} = (\mathcal{E}, \mathcal{H})$ is a tree-like structure, where each entity $e \in \mathcal{E}$ is a conceptual entity, and each edge $h \in \mathcal{H}$ represents the hypernymy-hyponymy relation between the two entities connected by it. Each entity e is associated with a set of textual description $\mathcal{X}_e = \{X_e^t, X_e^d\}$, where X_e^t is its term and X_e^d is its definition. Meanwhile, each directed edge $h = \langle p, c \rangle \in \mathcal{H}$ represents a parent-child relationship that points to a child entity c from its most exact hypernymy entity p.*

Definition 2 (Taxonomy Expansion) *Given a set of emerging conceptual entities $\mathcal{E}'$, taxonomy expansion aims to incorporate these entities into an existing seed taxonomy $\mathcal{T}_0 = (\mathcal{E}_0, \mathcal{H}_0)$. The goal is to expand $\mathcal{T}_0$ to be a larger taxonomy $\mathcal{T} = (\mathcal{E}_0 \cup \mathcal{E}', \mathcal{H}')$. To insert each query entity $q \in \mathcal{E}'$, we identify an appropriate anchor entity $a \in \mathcal{E}_0$, and introduce a new edge $\langle q, a \rangle$. Consequently, the updated edge set is $\mathcal{H}' = \mathcal{H}_0 \cup_{q \in \mathcal{E}'} \{\langle q, a \rangle\}$.*

3 Methodology

In this section, we provide a comprehensive overview of our proposed CODETAXO designed for addressing the taxonomy expansion task. Specifically, CODETAXO expands the existing taxonomy by prompting LLMs with code language. The pipeline of CODETAXO is shown in Figure 2. Our CODETAXO consists of three parts: Task Instruction, Taxonomy Code Representation, and Few-shot Demonstrations Construction.

```
from typing import List

class Entity:
    def __init__(
                self,
                name: str,
                description: str,
                parent: 'Entity',
                child: List['Entity']
    ):
        self.name = name
        self.description = description
        self.parent = parent
        self.child = child

    def add_parent(self, parent: 'Entity'):
        self.parent = parent
        parent.add_child(self)

    def add_child(self, child: 'Entity'):
        self.child.append(child)
```

Figure 4: Python code in CODETAXO defining a `Entity` class for managing parent-child relations.

3.1 Task Instruction

To enhance the effectiveness and accuracy of LLMs in completing the taxonomy expansion task, we propose a detailed task description along with a set of fundamental rules, denoted as $\mathcal{R}$, for expanding the existing taxonomy via the query entity. As illustrated in Figure 3, component (a) outlines the objectives of the taxonomy expansion task, framing it as a code completion task and specifying `add_parent` function should be employed. In component (b), we emphasize a set of fundamental rules $\mathcal{R}$ for the taxonomy expansion task. These rules include the following: 1. Do not use entities that are not covered in the existing taxonomy $\mathcal{T}_0 = (\mathcal{E}_0, \mathcal{H}_0)$ ($\mathbf{r}_1$); 2. Maintain the output generation format by LLMs, consisting of one line of code followed by one line explaining why the model made that prediction ($\mathbf{r}_2$); 3. Refrain from generating additional content ($\mathbf{r}_3$). Additionally, the rule for generating an explanation for the prediction in $\mathbf{r}_2$ is optional for future analysis. In CODETAXO, this rule is omitted as generating explanations is not required.

3.2 Taxonomy Code Representation

To represent the existing taxonomy $\mathcal{T}_0 = (\mathcal{V}_0, \mathcal{E}_0)$ as code language, we concatenate the entity class definition, representation of existing taxonomic relations, and the code completion prompt. We use Python as the programming language for the code prompt due to its widespread popularity.

3.2.1 Entity Class Definition

First, we define a base type `Entity` to be inherited by each entity mentioned in the taxonomy expansion. In Figure 4, we define a Python class named `Entity` that models a taxonomic structure with parent-child relations. The first line imports the `List` type from the `typing` module, which is used for type hinting. This allows the `child` attribute to be explicitly declared as a list of `Entity` objects.

The `Entity` class encapsulates the attributes and methods for managing hierarchical entities. The `__init__` method initializes an instance of the `Entity` class with the following parameters:

- `name`: A string storing the term of the entity.
- `description`: A string storing the textual description of the entity
- `parent`: An instance of the `Entity` class, denoting the parent entity within the taxonomy.
- `child`: A list of entities, each an instance of the `Entity` class, storing the entity's children.

These instance attributes are assigned as follows: `self.name`, `self.description`, `self.parent`, and `self.child`. Additionally, since we consider that each entity in the taxonomy should only have one parent entity, we do not use the `List` type for the `parent` attribute, unlike the `child` attribute.

The `Entity` class includes two methods for modifying the parent-child relations between entities. The first method, `add_parent`, assigns a parent entity to the current entity. It takes one parameter, `parent`, which is an instance of the `Entity` class. The second method, `add_child`, appends the child entity to the `self.child` list of the current entity. This method also requires one parameter, `child`, which is an instance of the `Entity` class.

3.2.2 Representing the Existing Taxonomy

To facilitate the taxonomy expansion, the initial taxonomy $\mathcal{T}_0$ is encoded using a programming language. Instances of the `Entity` class, as defined in Section 3.2.1, are created for each entity e in the set $\mathcal{E}_0$ of $\mathcal{T}_0$. The taxonomy $\mathcal{T}_0$ is traversed from top to bottom, and for each entry, an entity $e \in \mathcal{E}_0$ is instantiated as follows:

$$e = \text{Entity}(\text{name} = X_e^t, \text{description} = X_e^d,\\ \text{parent} = p_e, \text{child} = \mathcal{C}_e)$$

where p_e is the parent entity of e, and $\mathcal{C}_e = [c_e^1, c_e^2, \dots, c_e^n]$ is the list of its child entities.

3.2.3 Semantic Similarity Filter

Including all entity $e \in \mathcal{E}_0$ to represent the existing taxonomy $\mathcal{T}_0$ presents two problems. First, large-scale taxonomies overload the LLM's limited context window. Second, it unnecessarily expands the search space, introducing irrelevant entities and redundant information. To mitigate these issues, we propose a Semantic Similarity Filter that selects only entities relevant to the query q for inclusion in the prompt context.

To compute the similarity between a query entity q with its descriptive text $\mathcal{X}q = \{X_q^t, X_q^d\}$ and an entity $e_i \in \mathcal{E}0$ with its descriptive text $\mathcal{X}e_i = \{X_{e_i}^t, X_{e_i}^d\}$, we employ the pre-trained language model (PLM) as textual encoder. We concatenate the query entity q and the i-th entity e_i with special tokens [CLS] and [SEP], then encode the sequence using a pre-trained SimCSE model (Gao et al., 2021). SimCSE converts them into m-dimensional representation $\mathbf{q}, \mathbf{e}_i \in \mathbb{R}^m$:

$$\mathbf{q} = \text{PLM}([\text{CLS}] \oplus X_q^t \oplus X_q^d \oplus [\text{SEP}])$$
$$\mathbf{e}_i = \text{PLM}([\text{CLS}] \oplus X_{e_i}^t \oplus X_{e_i}^d \oplus [\text{SEP}])$$

The semantic relevance is calculated using cosine similarity between $\{\mathbf{e}_i\}_{i=1}^n$ and $\mathbf{q}$. We select the Top-k entities most similar to query entity q from the entity set $\mathcal{E}_0$ in $\mathcal{T}_0$ as follow:

$$\mathcal{I} = \underset{\substack{\mathcal{I} \subseteq \{1,2,\dots,n\},\\ |\mathcal{I}|=k}}{\text{argmax}} \sum_{i \in \mathcal{I}} \text{cos\_sim}(\mathbf{e}_i, \mathbf{q})$$

where $\mathcal{I}$ is the index set of the selected entities $\mathcal{E}_{sel} = \{e_i | i \in \mathcal{I}\}$ that represents the existing taxonomy. k is set to 50% of the entities in $\mathcal{E}_0$.

3.2.4 Code Completion Prompt.

The code completion prompt involves the instantiation of a query entity q as an instance of the `Entity` class, as defined in Section 3.2.1. Since the query entity q lacks information about its parent and child entities, it is instantiated as follows:

$$q = \text{Entity}(\text{name} = X_q^t, \text{description} = X_q^d,\\ \text{parent} = \text{None}, \text{child} = [])$$

Here, X_q^t and X_q^d define the query's name and description, while `None` and `[]` indicate the absence of parent and child entities.

We include the requirement "*Find the parent of the query node*" as a comment to guide LLMs in selecting an anchor entity $a \in \mathcal{E}_{sel}$ as the parent entity for entity q. The output is the query q, an instance of the `Entity` class, which invokes the predefined method `add_parent()` to assign a as its parent entity like `q.add_parent(a)`.

Dataset	SemEval-Sci		SemEval-Env		SemEval-Food		WordNet		Graphine	
Metric	**Acc**	**Wu&P**	**Acc**	**Wu&P**	**Acc**	**Wu&P**	**Acc**	**Wu&P**	**Acc**	**Wu&P**
Self-supervised Setting										
TaxoExpan	27.8	57.6	11.1	54.8	27.6	54.2	19.8	64.8	24.5	65.9
STEAM	36.5	68.2	36.1	69.6	34.2	67.0	23.2	62.4	20.3	63.1
HEF	53.6	75.6	55.3	71.4	47.9	73.5	16.4	60.3	25.5	66.5
Musubu	44.9	76.2	45.3	65.4	42.3	72.4	28.5	64.0	35.4	75.2
TEMP	57.8	85.3	49.2	77.7	47.6	81.0	29.4	65.7	35.9	73.8
BoxTaxo	31.8	64.7	38.1	75.4	31.4	66.8	26.4	63.9	29.2	68.2
TaxoPrompt	61.4	85.6	57.4	83.6	53.2	83.1	40.3	71.5	33.9	74.4
TaxoInstruct	45.9	76.2	48.8	77.2	34.3	70.2	43.3	71.8	31.8	69.0
1-shot Setting										
NL (GPT-4o)	54.8	88.3	52.5	81.3	55.5	85.6	72.2	90.7	69.8	89.1
CODETAXO (GPT-4o)	**67.7**	**89.2**	**62.5**	**86.1**	**58.1**	85.3	**74.5**	**91.3**	**72.9**	**91.0**
NL (GPT-4o-mini)	50.0	83.0	35.0	76.1	55.1	**87.2**	60.1	86.0	58.3	85.2
CODETAXO (GPT-4o-mini)	58.1	85.6	42.5	76.0	55.9	85.3	68.8	89.2	61.5	85.1
5-shot Setting										
NL (GPT-4o)	56.5	84.3	60.0	85.5	52.5	86.9	72.2	90.1	69.3	90.0
CODETAXO (GPT-4o)	**66.1**	**88.0**	**67.5**	**87.0**	**60.2**	85.7	**76.5**	**91.9**	**77.6**	**93.4**
NL (GPT-4o-mini)	53.2	84.8	42.5	80.2	57.2	87.6	63.4	87.3	63.5	88.6
CODETAXO (GPT-4o-mini)	59.7	84.8	47.5	78.3	58.9	**87.9**	66.8	88.6	70.3	89.1

Table 1: Performance on taxonomy expansion across two small-scale taxonomies (WordNet and Graphine) and three large-scale taxonomies (SemEval2016: science, environment, food). Bold indicates the highest score; underlined indicates the second-highest. All metrics are in percentages (%).

We propose incorporating an optional feature in the code completion prompt: "*then generating a comment to explain why it is the parent of the given query node*". This feature allows the LLM to simultaneously generate both the prediction and its rationale, improving explainability and revealing interesting insights, as discussed in Section 4.5.

3.3 Few-shot Demonstration Construction

To enhance LLMs' ability to expand our existing taxonomy, we propose a method for constructing demonstrations using the initial taxonomy $\mathcal{T}_0$. Our demonstration selection strategy focuses on the semantic similarity between the query entity q and entities $e \in \mathcal{E}_0$ in the existing taxonomy. Specifically, we use SimCSE encoding to calculate these similarities, selecting the top-5 entities from the existing set $\mathcal{E}_0$ based on their similarity to q:

$$\mathcal{I}_d = \underset{\substack{\mathcal{I}_d \subseteq \{1,2,\dots,n\}, \\ |\mathcal{I}_d|=5}}{\operatorname{argmax}} \sum_{i \in \mathcal{I}_d} \cos\_\mathrm{sim}(\mathbf{e}_i, \mathbf{q})$$

Here, $\mathcal{I}_d$ represents the indices of entities selected for the demonstration set $\mathcal{E}_{demo} = \{e_i | i \in \mathcal{I}_d\}$. For each demonstration d_i, we treat each entity $e_i \in \mathcal{E}_{demo}$ as a query entity and, following the procedure outlined in Section 3.2.4, add its parent entity using the `add_parent` method.

4 Experiments

4.1 Experimental Settings

Datasets. We evaluate taxonomy expansion on small-scale WordNet sub-taxonomies (Bansal et al., 2014) and Graphine taxonomies (Liu et al., 2021a). Additionally, we evaluate three large-scale taxonomies from SemEval-2016 (Bordea et al., 2016) across science, environment, and food domains. For all benchmarks, 20% of leaf entities are reserved for testing, with the remaining entities used for training. See App. A.1 for details.

Baselines. We evaluate CODETAXO, using both GPT-4O and GPT-4O-MINI, against self-supervised baselines including TaxoExpan (Shen et al., 2020), STEAM (Yu et al., 2020), HEF (Wang et al., 2022), Musubu (Takeoka et al., 2021), TEMP (Liu et al., 2021b), BoxTaxo (Jiang et al., 2023), TaxoPrompt (Xu et al., 2022), and TaxoInstruct (Shen et al., 2024), and prompting LLMs through natural language. See details in App. A.2.

Evaluation Metrics: We use two Accuracy (ACC) and Wu & Palmer similarity (Wu&P) to evaluate the performance of CODETAXO and baselines. See details in App. A.3

Method	Def.	SemEval-Sci		SemEval-Env		SemEval-Food		WordNet		Graphine	
		Acc	Wu&P	Acc	Wu&P	Acc	Wu&P	Acc	Wu&P	Acc	Wu&P
1-shot Setting											
NL (GPT-4o)	✓	54.8	88.3	52.5	81.3	55.5	85.6	72.2	90.7	69.8	89.1
	×	59.7	89.0	57.5	82.8	56.4	87.0	68.1	89.1	68.8	90.1
CODETAXO (GPT-4o)	✓	**67.7**	**89.2**	**62.5**	**86.1**	**58.1**	85.3	**74.5**	**91.3**	**72.9**	**91.0**
	×	56.5	84.5	55.0	85.1	56.8	86.1	66.4	88.4	69.8	88.8
5-shot Setting											
NL (GPT-4o)	✓	56.5	84.3	60.0	85.5	52.5	86.9	72.2	90.1	69.3	90.0
	×	59.7	89.6	50.0	79.3	55.5	87.6	70.5	89.9	68.8	88.9
CODETAXO (GPT-4o)	✓	**66.1**	**88.0**	**67.5**	**87.0**	**60.2**	85.7	**76.5**	**91.9**	**77.6**	**93.4**
	×	51.6	80.6	65.0	86.7	57.6	86.1	67.8	88.8	68.8	89.7

Table 2: Impact of Definition Sentences (Def.) on CODETAXO and NL Performance in 1-Shot and 5-Shot Settings.

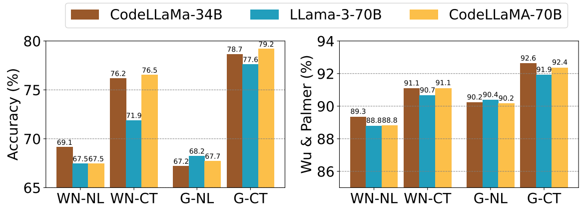


Figure 5: Performance comparison of NL and CODETAXO (CT) across Llama trained on Code and Natural Language domains. Due to limited contextual window sizes, evaluations were conducted on small-scale subtaxonomies from WordNet (WN) and Graphine (G).

4.2 Experimental Results

4.2.1 Can CODETAXO expand taxonomy better than other baselines?

We evaluate CodeTaxo against baseline methods for taxonomy expansion in Table 1, including self-supervised and in-context learning approaches. On WordNet and Graphine, both NL and CODETAXO significantly outperform self-supervised baselines. In one-shot settings, CODETAXO improves accuracy by 72.06% and 103.06% over the best self-supervised methods, demonstrating that minimal annotated data effectively unlocks LLMs' internal knowledge, while self-supervised methods struggle with limited-scale taxonomies. On large-scale SemEval-2016 taxonomies, CODETAXO surpasses the best self-supervised baseline, TaxoPrompt, by 10.26%, 8.89%, and 9.21% on SemEval-Sci, SemEval-Env, and SemEval-Food, respectively. While the NL prompt underperforms TaxoPrompt on SemEval-Sci and SemEval-Env, it exceeds TaxoPrompt on SemEval-Food but still trails CodeTaxo by 4.68%, highlighting CODETAXO's superior ability to capture taxonomic structures. Performance depends on LLM capability and demonstration count, with GPT-4o outperforming GPT-4o-mini and more demonstrations improving accuracy across benchmarks, underscoring the value of high-quality demonstrations for taxonomy expansion.

Setting	Config.		SemEval-Sci		SemEval-Env		SemEval-Food	
	Demo.	Filter	Acc	Wu&P	Acc	Wu&P	Acc	Wu&P
1-shot	×	×	50.0	84.0	47.5	81.1	56.4	85.3
	×	✓	61.3	84.4	55.0	83.2	54.2	84.6
	✓	×	61.3	85.9	47.5	79.0	57.2	86.7
	✓	✓	67.7	89.2	62.5	86.1	58.1	85.3
5-shot	×	×	58.1	86.0	55.0	82.8	56.4	86.5
	×	✓	59.7	84.7	57.5	83.7	58.5	85.8
	✓	×	61.3	88.5	55.0	85.3	57.6	86.5
	✓	✓	66.1	88.0	67.5	87.02	60.2	85.7

Table 3: Ablation Study of two major modules in the CODETAXO: All metrics are presented in percentages (%). Configurations indicate whether Demonstration Selection (Demo.) and Semantic Similarity Filter (Filter) were employed.

4.2.2 How does CODETAXO perform across different large language models?

We evaluate CODETAXO, a prompting method specifically designed for programming languages, by comparing its effectiveness against natural language prompting on both general-purpose LLMs and Code-LLMs. We used Llama-family models, including LLaMa-3-70B-instruct, CodeLLaMA-70B-instruct, and the smaller CodeLLaMA-34B-instruct, to evaluate how model size affects performance. Given the limited contextual capacity of these models, we focused our evaluation on WordNet and Graphine, as shown in Figure 5. The results highlight CODETAXO's superior accuracy and Wu&P scores across all tested models, outperforming natural language prompts in representing taxonomic structures for black-box LLMs like GPT-4 and open-source LLMs. The analysis further reveals that CODETAXO benefits more significantly from Code-LLMs, with a 13.33% accuracy improvement on WordNet compared to 6.51% for natural language prompts when transitioning to

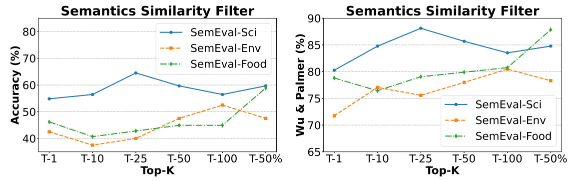


Figure 6: Effect of Top-K relevant entities selected through SimCSE-based Semantic Similarity Filter.

code language prompting. Notably, CodeLLaMA-34B-instruct, despite being smaller, showed better performance on WordNet and Graphine, emphasizing CODETAXO's efficiency and robustness.

4.3 Hyperparmeter Analysis of CODETAXO

This section explores the impact of selecting Top-K entities using the Semantic Similarity Filter on model performance, with experiments conducted using GPT-4o-mini across three SemEval taxonomies. As shown in Figure 6, increasing the number of Top-K entities generally enhances performance by retaining more entities, thereby reducing the likelihood of filtering out the ground truth and boosting prediction accuracy. However, this improvement involves a trade-off: a smaller search space sharpens the model's focus but increases the risk of excluding the ground truth. For instance, in the SemEval-Sci taxonomy, the model achieved optimal performance with a Hit@25 score of 78% by retaining the top 25 entities, demonstrating the filter's ability to balance search space and coverage. To further refine this balance, we retained the top 50% of entities in our experiments, ensuring that Hit@n exceeded 90% across all benchmarks.

4.4 Ablation Study

4.4.1 Insight of Definition Sentences

We performed an ablation study on definition sentences, a vital data source for taxonomy expansion tasks, using two prompting methods: NL and CODETAXO. Our results in Table 2 show that without definition sentences, CODETAXO suffers a substantial drop in accuracy and Wu&P across all benchmarks in both 1-shot and 5-shot settings, highlighting its reliance on semantic information from definitions to establish taxonomic relationships. Interestingly, NL performed better without definition sentences in specific benchmarks (SemEval-Sci, SemEval-Env, SemEval-Food) in the 1-shot setting, and in SemEval-Sci and SemEval-Food in the 5-shot setting. This suggests that NL struggles to process definition information effectively, potentially leading to incorrect predictions when overloaded with definitional content.

4.4.2 Effevtiveness of Demo. Selection and Semantic Similarity Filter

We performed an ablation study on the three SemEval2016 benchmarks mentioned above to assess the effectiveness of the two primary modules in CODETAXO: Demonstration Selection (Demo.) and the Semantic Similarity Filter (Filter). Due to the relatively small size of the taxonomies in WordNet and Graphine, filtering redundant entities from the existing taxonomies was unnecessary. The results, presented in Table 3, indicate that selecting demonstrations related to the query entity and filtering out unrelated entities in the existing taxonomy significantly improves taxonomy expansion. This finding suggests that incorporating more relevant contextual information and reducing redundant information to narrow the search space is beneficial for both accuracy and the Wu&P score across all SemEval2016 benchmarks.

4.5 Case Study

This section presents a case study demonstrating the effectiveness of our CODETAXO framework by comparing its outputs to those of the natural language (NL) prompt, alongside model predictions and corresponding definitions in Table 4. Notably, CODETAXO aligns closely with the ground truth and generates explanations using the prompt from Section 3.2.4 to facilitate insightful discussions. For instance, in the query *functional genomics*, CODETAXO accurately classifies it under genomics, emphasizing its focus on the "functions and interactions of genes within the genome", whereas the NL model incorrectly selects the broader term *genetics*. Similarly, CODETAXO identifies *geopolitics* as the parent entity of *geostrategy*, highlighting its emphasis on *geographic factors*, while the NL model selects the more general category of politics. These cases showcase CODETAXO's ability to leverage definition information for a comprehensive understanding of taxonomy structures, resulting in more precise predictions.

5 Related Works

5.1 Taxonomy Expansion

In taxonomy expansion, various approaches have been developed to integrate emerging entities into

Query	NL Output	CodeTaxo Output	CodeTaxo Exp.
functional genomics *Def.* functional genomics is a field of molecular biology that attempts to describe gene functions and interactions	**genetics** *Def.* genetics is a branch of biology concerned with the study of genes, genetic variation, and heredity in organisms.	**genomics** *Def.* genomics is an interdisciplinary field of biology focusing on the structure, function, evolution, mapping, and editing of genomes	**genomics** *Exp.* Functional genomics is a subfield of genomics that focuses on understanding the functions and interactions of genes within the genome.
geostrategy *Def.* geostrategy, a subfield of geopolitics, is a type of foreign policy guided principally by geographical factors as they inform, constrain, or affect political and military planning	**politics** *Def.* politics is the set of activities that are associated with making decisions in groups, or other forms of power relations between individuals, such as the distribution of resources or status	**geopolitics** *Def.* geopolitics on politics and international relations	**geopolitics** *Exp.* : Geostrategy is a subfield of geopolitics, which focuses on geographic factors influencing political and military planning.

Table 4: Case study comparing the outputs of the CodeTaxo and NL prompt using the SemEval-Sci benchmarks. The table presents the definitions (Def.) of each model's prediction, additionally with the CodeTaxo explanations (Exp.) provided in the last column (CodeTaxo Exp.). Yellow highlights emphasize the specific focus of the query within its definition, as correctly captured by CodeTaxo, while green highlights indicate broader, less precise concepts used by the NL model.

existing taxonomies. Aly *et al.* (Aly et al., 2019) and Ma *et al.* utilized hyperbolic embeddings to capture taxonomic relations, while Jiang *et al.* (Jiang et al., 2023) and Xu *et al.* (Xu et al., 2024) employed box embedding and fuse embedding instead of single vector embedding to encode taxonomic relations respectively. Manzoor *et al.* (Manzoor et al., 2020) introduced implicit edge semantics to enhance entity representations. Self-supervised methods, such as Egonet (Shen et al., 2020), mini-path (Yu et al., 2020), and Ego-Tree (Wang et al., 2021), have also been explored to model structural information within taxonomies. To leverage more semantic information from the textural description of entities, Liu *et al.* (Liu et al., 2021b), Takeoka *et al.* (Takeoka et al., 2021) and Xu *et al.* (Xu et al., 2022) fine-tuned BERT-based models to leverage textual descriptions of entities. Zhu *et al.* (Zhu et al., 2023) integrates textual and visual semantics to capture the hierarchical relation between entities. Shen *et al.* (Shen et al., 2024) and Moskvoretskii *et al.* (Moskvoretskii et al., 2024) unified framework combining various taxonomy construction tasks for instruction tuning. To our knowledge, CodeTaxo is the first work to perform taxonomy expansion via prompting LLMs.

5.2 Code-LLMs for Structured Tasks

Recent studies have demonstrated the strong performance of Code-LLMs in complex reasoning tasks (Yang et al.; MA et al.), including symbolic reasoning (Madaan et al., 2022; Cheng et al.), graph reasoning (Cai et al., 2024), event structure prediction (Wang et al., 2023; Chen et al., 2023), mathematical reasoning (Gao et al., 2023), and knowledge graph construction (Li et al., 2023; Bi et al., 2024). These works highlight Code-LLMs' ability to transform unstructured text into structured representations, enabling advanced reasoning tasks. In this paper, we focus on enhancing Code-LLMs' ability to comprehend and expand existing taxonomies through emerging query entities.

6 Conclusion

In this paper, we introduce CodeTaxo, a novel approach to taxonomy expansion that leverages code-based prompts to effectively utilize the inherent knowledge within LLMs. Our method addresses key challenges in taxonomy expansion by reformulating the task as a code completion problem and employing a Semantic Similarity Filtering mechanism to optimize the use of LLMs' contextual capacity. Extensive experiments on small-scale and large-scale taxonomies demonstrate that CodeTaxo achieves state-of-the-art performance in both one-shot settings and five-shot settings. We envision CodeTaxo as a powerful framework for integrating emerging entities into existing taxonomies by accurately identifying appropriate parent entities and also providing new insights for leveraging LLMs in structured knowledge tasks.

Limitations

This study represents an initial effort to utilize LLMs for taxonomy expansion. Our primary objective is to identify an effective in-context learning strategy to leverage the potential of LLMs. We acknowledge that the performance and scalability of CODETAXO are constrained by the inherent knowledge of LLMs and the limitations of their context window size. While this paper does not address the challenges of expanding LLM knowledge or increasing context window size, we hope that our work will inspire further research in these areas.

Ethics Statement

Our research addresses taxonomy expansion within general knowledge domains, leveraging our proposed method, CODETAXO, which uses large language models (LLMs) to generate structured knowledge and overcome the limitations of traditional manual taxonomy construction. We exclusively utilize publicly available datasets and benchmarks, avoiding user-generated, private, or sensitive data to ensure compliance with privacy and ethical standards. While our datasets do not engage directly with ethically sensitive content, LLMs inherently carry biases from their pre-training data, which may influence the structure and content of the expanded taxonomies. To address this, we integrate mechanisms for generating explanatory outputs, enabling detailed scrutiny of the model's reasoning and identifying potential biases. Additionally, we recognize the risks of applying similar methodologies to subjective or sensitive domains, which could lead to misrepresentation or bias. To mitigate such risks, we emphasize collaboration with domain experts and advocate for responsible application of our methodologies across diverse fields, aiming to promote fairness, accuracy, and ethical research practices.

Acknowledgements

This work was supported by NSF IIS-2119531, IIS-2137396, IIS-2142827, IIS-2234058, CCF-1901059, and ONR N00014-22-1-2507. The authors also thank the OpenAI Researcher Access Program (ID: 0000003889) for its support.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Rami Aly, Shantanu Acharya, Alexander Ossa, Arne Köhn, Chris Biemann, and Alexander Panchenko. 2019. Every child should have parents: A taxonomy refinement algorithm based on hyperbolic term embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4811–4817.

Mohit Bansal, David Burkett, Gerard De Melo, and Dan Klein. 2014. Structured learning for taxonomy induction with belief propagation. In *ACL*, pages 1041–1051.

Zhen Bi, Jing Chen, Yinuo Jiang, Feiyu Xiong, Wei Guo, Huajun Chen, and Ningyu Zhang. 2024. Codekgc: Code language model for generative knowledge graph construction. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 23(3):1–16.

Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. Semeval-2016 task 13: Taxonomy extraction evaluation (texeval-2). In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)*, pages 1081–1091.

Qiaolong Cai, Zhaowei Wang, Shizhe Diao, James Kwok, and Yangqiu Song. 2024. Codegraph: Enhancing graph reasoning of llms with code. *arXiv preprint arXiv:2408.13863*.

Yangyi Chen, Xingyao Wang, Manling Li, Derek Hoiem, and Heng Ji. 2023. Vistruct: Visual structural knowledge extraction via curriculum guided code-vision representation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13342–13357.

Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, et al. Binding language models in symbolic languages. In *The Eleventh International Conference on Learning Representations*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

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*, pages 573–581.

Song Jiang, Qiyue Yao, Qifan Wang, and Yizhou Sun. 2023. A single vector is not enough: Taxonomy expansion via box embeddings. In *Proceedings of the ACM Web Conference 2023*, pages 2467–2476.

David Jurgens and Mohammad Taher Pilehvar. 2016. Semeval-2016 task 14: Semantic taxonomy enrichment. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 1092–1102.

SeongKu Kang, Shivam Agarwal, Bowen Jin, Dongha Lee, Hwanjo Yu, and Jiawei Han. 2024. Improving retrieval in theme-specific applications using a corpus topical taxonomy. In *Proceedings of the ACM on Web Conference 2024*, pages 1497–1508.

Peng Li, Tianxiang Sun, Qiong Tang, Hang Yan, Yuanbin Wu, Xuan-Jing Huang, and Xipeng Qiu. 2023. Codeie: Large code generation models are better few-shot information extractors. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15339–15353.

Zixuan Li, Yutao Zeng, Yuxin Zuo, Weicheng Ren, Wenxuan Liu, Miao Su, Yucan Guo, Yantao Liu, Xiang Li, Zhilei Hu, et al. 2024. Knowcoder: Coding structured knowledge into llms for universal information extraction. *arXiv preprint arXiv:2403.07969*.

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 ACM SIGMOD International Conference on Management of Data*, pages 393–409.

Zequn Liu, Shukai Wang, Yiyang Gu, Ruiyi Zhang, Ming Zhang, and Sheng Wang. 2021a. Graphine: A dataset for graph-aware terminology definition generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3453–3463.

Zichen Liu, Hongyuan Xu, Yanlong Wen, Ning Jiang, Haiying Wu, and Xiaojie Yuan. 2021b. Temp: Taxonomy expansion with dynamic margin loss through taxonomy-paths. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3854–3863.

YINGWEI MA, Yue Liu, Yue Yu, Yuanliang Zhang, Yu Jiang, Changjian Wang, and Shanshan Li. At which training stage does code data help llms reasoning? In *The Twelfth International Conference on Learning Representations*.

Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. 2022. Language models of code are few-shot commonsense learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1384–1403.

Emaad Manzoor, Rui Li, Dhananjay Shrouty, and Jure Leskovec. 2020. Expanding taxonomies with implicit edge semantics. In *Proceedings of The Web Conference 2020*, pages 2044–2054.

Viktor Moskvoretskii, Ekaterina Neminova, Alina Lobanova, Alexander Panchenko, and Irina Nikishina. 2024. Taxollama: Wordnet-based model for solving multiple lexical sematic tasks. *arXiv preprint arXiv:2403.09207*.

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*, pages 486–497.

Yanzhen Shen, Yu Zhang, Yunyi Zhang, and Jiawei Han. 2024. A unified taxonomy-guided instruction tuning framework for entity set expansion and taxonomy expansion. *arXiv preprint arXiv:2402.13405*.

Kai Sun, Yifan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2024a. Head-to-tail: How knowledgeable are large language models (llms)? aka will llms replace knowledge graphs? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 311–325.

Yushi Sun, Hao Xin, Kai Sun, Yifan Ethan Xu, Xiao Yang, Xin Luna Dong, Nan Tang, and Lei Chen. 2024b. Are large language models a good replacement of taxonomies? *arXiv preprint arXiv:2406.11131*.

Kunihiro Takeoka, Kosuke Akimoto, and Masafumi Oyamada. 2021. Low-resource taxonomy enrichment with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2747–2758.

Yanchao Tan, Carl Yang, Xiangyu Wei, Chaochao Chen, Longfei Li, and Xiaolin Zheng. 2022. Enhancing recommendation with automated tag taxonomy construction in hyperbolic space. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 1180–1192. IEEE.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti

Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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*, pages 3291–3304.

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*, pages 1008–1017.

Xingyao Wang, Sha Li, and Heng Ji. 2023. Code4struct: Code generation for few-shot event structure prediction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3640–3663.

Fred Xu, Song Jiang, Zijie Huang, Xiao Luo, Shichang Zhang, Yuanzhou Chen, and Yizhou Sun. 2024. Fuse: Measure-theoretic compact fuzzy set representation for taxonomy expansion. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 2707–2720.

Hongyuan Xu, Yunong Chen, Zichen Liu, Yanlong Wen, and Xiaojie Yuan. 2022. Taxoprompt: A prompt-based generation method with taxonomic context for self-supervised taxonomy expansion. In *IJCAI*, pages 4432–4438.

Ke Yang, Jiateng Liu, John Wu, Chaoqi Yang, Yi Fung, Sha Li, Zixuan Huang, Xu Cao, Xingyao Wang, Heng Ji, et al. If llm is the wizard, then code is the wand: A survey on how code empowers large language models to serve as intelligent agents. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Shuo Yang, Lei Zou, Zhongyuan Wang, Jun Yan, and Ji-Rong Wen. 2017. Efficiently answering technical questions—a knowledge graph approach. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Hongbin Ye, Ningyu Zhang, Hui Chen, and Huajun Chen. 2022. Generative knowledge graph construction: A review. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1–17.

Xiaoxin Yin and Sarthak Shah. 2010. Building taxonomy of web search intents for name entity queries. In *Proceedings of the 19th international conference on World wide web*, pages 1001–1010.

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*, pages 1026–1035.

Yuchen Zhang, Amr Ahmed, Vanja Josifovski, and Alexander Smola. 2014. Taxonomy discovery for personalized recommendation. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 243–252.

Tinghui Zhu, Jingping Liu, Jiaqing Liang, Haiyun Jiang, Yanghua Xiao, Zongyu Wang, Rui Xie, and Yunsen Xian. 2023. Towards visual taxonomy expansion. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 6481–6490.

A Appendix

A.1 Datasets

We evaluate the performance of taxonomy expansion methods on small-scale taxonomies using WordNet Sub-taxonomies from (Bansal et al., 2014), and Graphine taxonomies from (Liu et al., 2021a). Specifically, we use 35 Graphine taxonomies with fewer than 100 entities, selected from a total of 227 taxonomies. For the Graphine dataset, we selected 35 taxonomies with fewer than 100 entities out of 227 total taxonomies. In our experiment with WordNet, we utilized 114 sub-taxonomies from the test sets. Additionally, we evaluate three large-scale taxonomies from SemEval-2016 (Bordea et al., 2016) across science, environment, and food domains. Table 5 presents the statistics of these taxonomies, all of which contain entities and definitions curated by human experts. For all benchmarks, 20% of leaf entities are reserved for testing, with the remaining entities used for training.

	#Concepts	#Edges	Depth	License
WordNet	20.5	19.5	3.0	WordNet
Graphine	48.2	48.2	4.6	None
SemEval-Sci	429.0	451.0	8.0	None
SemEval-Env	261.0	261.0	6.0	None
SemEval-Food	1,486.0	1,576.0	8.0	None

Table 5: Statistics of five taxonomy benchmarks. For WordNet and Graphine, we report the average for taxonomies included in these two benchmarks.

A.2 Baselines

We compare our method with the following baselines for taxonomy expansion, all experiments are implemented in a server with three NVIDIA A6000 GPUs:

- **TaxoExpan (Shen et al., 2020):** adopts GNNs to encode local ego-graphs in taxonomy to enhance entity representation.

- **STEAM (Yu et al., 2020):** utilizes the mini-path information to capture the global structure of the taxonomy.
- **HEF (Wang et al., 2022):** represents taxonomies as ego-trees to capture hierarchy, fully leveraging the hierarchical structure to improve taxonomy coherence.
- **Musubu (Takeoka et al., 2021):** leverages pre-trained models and fine-tunes them as sentence classifiers using queries generated from Hearst patterns.
- **TEMP (Liu et al., 2021b):** utilizes a pre-trained model to encode text descriptions of each concept in the taxonomy. It incorporates taxonomic structure information through taxonomy paths.
- **BoxTaxo (Jiang et al., 2023):** represent the entities via box embeddings instead of single vector embeddings to capture the hierarchical relation between entities.
- **TaxoPrompt (Xu et al., 2022):** adopt prompt tuning on the BERT-based encoder model to capture the taxonomic structure.
- **TaxoInstruct (Shen et al., 2024):** a unified framework for taxonomy-related tasks using instruction tuning, focused solely on taxonomy expansion for fair comparison.

To the best of our knowledge, CODETAXO represents the first work to address taxonomy expansion using an in-context learning approach. To validate the effectiveness of the code language based prompt design, we additionally propose a prompting method based on natural language prompts. The results obtained using the natural language prompt (NL) are presented in Table 1. To ensure that the natural language prompt communicates the same information as code language prompt in CODETAXO, we represent each entity using natural language to describe its surface name, definition, parent, and children list. The details of the NL prompt are provided in Table 7. For a more direct comparison, we also demonstrate CODETAXO's predictions on the same example in Table 8.

A.3 Evaluation Metrics.

The performance of CODETAXO and the baseline models for taxonomy expansion tasks is evaluated using commonly adopted metrics, including accuracy (Acc) and Wu & Palmer similarity (Wu&P), as established in prior work (Yu et al., 2020; Liu et al., 2021b; Wang et al., 2021). Since CODETAXO is a generation-based method rather than a ranking-based one, the mean reciprocal rank (MRR) used in the baselines is not applicable to CODETAXO.

Dataset	1-shot		5-shot	
	NL	CodeTaxo	NL	CodeTaxo
SemEval-Sci	15737.2	9701.4	16095.1	10342.6
SemEval-Env	8965.7	5693.6	9325.0	6321.1
SemEval-Food	48908.1	30536.5	49266.4	31176.3
WordNet	948.9	1369.2	1306.4	1962.3
Graphine	2486.0	3223.9	2855.2	3893.3

Table 6: Comparison of average tokens used by NL and CODETAXO across 5 benchmarks in 1-shot and 5-shot settings.

A.4 Efficiency Analysis of CODETAXO

Token Consumption Table 6 compares average token usage across benchmarks and prompt types (CODETAXO vs. NL) in 1-shot and 5-shot settings. The findings highlight CODETAXO's efficiency in reducing token usage while maintaining effectiveness. Notably, in SemEval2016, CODETAXO cuts token usage by approximately 37.6% in the SemEval-Food task compared to natural language prompts. However, in the WordNet and Graphine datasets, CODETAXO uses slightly more tokens due to the need to define Entity classes and methods. Overall, the significant reduction in token usage in SemEval2016 underscores CODETAXO's efficiency, especially in contexts with limited token windows.

Natural Language Prompt
User: Given the current taxonomy, find the parent of the query node. Please note that the query node may be a new node not in the current taxonomy. The parent of given query node always exists, so do not generate 'none' or 'not found'. You only need to answer the entity name and do not generate any additional content or comments. lunacy: obsolete terms for legal insanity; parent: insanity; children: []. irrationality: the state of being irrational; lacking powers of understanding; parent: insanity; children: []. dementia: mental deterioration of organic or functional origin; parent: insanity; children: ['presenile dementia', 'alcoholic dementia', 'senile dementia']. alcoholic dementia: dementia observed during the last stages of severe chronic alcoholism; involves loss of memory for recent events although long term memory is intact; parent: dementia; children: []. Pick's disease: a progressive form of presenile dementia found most often in middle-aged and elderly women and characterized by degeneration of the frontal and temporal lobes with loss of intellectual ability and transitory aphasia; parent: presenile dementia; children: []. derangement: a state of mental disturbance and disorientation; parent: insanity; children: []. craziness: informal terms for insanity; parent: insanity; children: []. presenile dementia: dementia with onset before the age of 65; parent: dementia; children: ["Pick's disease"]. senile dementia: dementia of the aged; results from degeneration of the brain in the absence of cerebrovascular disease; parent: dementia; children: []. insanity: relatively permanent disorder of the mind; parent: None; children: ['irrationality', 'dementia', 'craziness', 'derangement', 'lunacy']. Query node: Alzheimer's disease The parent of query node: **Assistant:** dementia **Ground Truth:** presenile dementia

Table 7: Example of Natural Language (NL) Prompt.

Code Prompt

User: Complete the next line of code according to the comments and the given code snippet. You need to find the parent of the query node in the given current taxonomy and use the add_parent function. The parent of given query node always exists in the given current taxonomy, so do NOT generate node that is NOT in the given current taxonomy. Note that you only need to complete the next ONE line of code, do not generate any additional content or comments.

```
from typing import List

class Entity:
  def __init__(self, name: str, description: str, parent: str, child: List['Entity']):
    self.name = name
    self.description = description
    self.parent = parent
    self.child = child
  def add_parent(self, parent: 'Entity'):
    self.parent = parent.name
    parent.add_child(self)
  def add_child(self, child: 'Entity'):
    self.child.append(child)

# Creating entities and establishing parent-child relationship
lunacy = Entity(name='lunacy', description='obsolete terms for legal insanity', parent=insanity, child=[])
irrationality = Entity(name='irrationality', description='the state of being irrational; lacking powers of understanding', parent=insanity, child=[])
dementia = Entity(name='dementia', description='mental deterioration of organic or functional origin', parent=insanity, child=['presenile dementia', 'alcoholic dementia', 'senile dementia'])
alcoholic_dementia = Entity(name='alcoholic dementia', description='dementia observed during the last stages of severe chronic alcoholism; involves loss of memory for recent events although long term memory is intact', parent=dementia, child=[])
Pick's_disease = Entity(name='Pick's disease', description='a progressive form of presenile dementia found most often in middle-aged and elderly women and characterized by degeneration of the frontal and temporal lobes with loss of intellectual ability and transitory aphasia', parent=presenile dementia, child=[])
derangement = Entity(name='derangement', description='a state of mental disturbance and disorientation', parent=insanity, child=[])
craziness = Entity(name='craziness', description='informal terms for insanity', parent=insanity, child=[])
presenile_dementia = Entity(name='presenile dementia', description='dementia with onset before the age of 65', parent=dementia, child=["Pick's disease"])
senile_dementia = Entity(name='senile dementia', description='dementia of the aged; results from degeneration of the brain in the absence of cerebrovascular disease', parent=dementia, child=[])
insanity = Entity(name='insanity', description='relatively permanent disorder of the mind', parent=None, child=['irrationality', 'dementia', 'craziness', 'derangement', 'lunacy'])

# creating query node
Alzheimer's_disease = Entity(name='Alzheimer's disease', description='a progressive form of presenile dementia that is similar to senile dementia except that it usually starts in the 40s or 50s; first symptoms are impaired memory which is followed by impaired thought and speech and finally complete helplessness', parent=None, child=[])

# Finding the parent of query node
```

Assistant: Alzheimer's_disease.add_parent(presenile_dementia)

Ground Truth: presenile dementia

Table 8: Example of Code-based Prompt.