HierPrompt: Zero-Shot Hierarchical Text Classification with LLM-Enhanced Prototypes

Qian Zhang[1], Qinliang Su*[1,2], Wei Zhu[3], Yachun Pang[3]
[1] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China,
[2]Guangdong Key Laboratory of Big Data Analysis and Processing,Guangzhou, China,
[3]China Mobile Internet Company Ltd.
zhangq637@mail2.sysu.edu.cn, suqliang@mail.sysu.edu.cn,
{zhuwei6,pangyachun}@cmic.chinamobile.com

Abstract

Hierarchical Text Classification is a challenging task which classifies texts into categories arranged in a hierarchy. Zero-Shot Hierarchical Text Classification (ZS-HTC) further assumes only the availability of hierarchical taxonomy, without any training data. Existing works of ZS-HTC are typically built on the prototype-based framework by embedding the category names into prototypes, which, however, do not perform very well due to the ambiguity and impreciseness of category names. In this paper, we propose HierPrompt, a method that leverages hierarchy-aware prompts to instruct LLM to produce more representative and informative prototypes. Specifically, we first introduce Example Text Prototype (ETP), in conjunction with Category Name Prototype (CNP), to enrich the information contained in hierarchical prototypes. A Maximum Similarity Propagation (MSP) technique is also proposed to consider the hierarchy in similarity calculation. Then, the hierarchical prototype refinement module is utilized to (i) contextualize the category names for more accurate CNPs and (ii) produce detailed example texts for each leaf category to form ETPs. Experiments on three benchmark datasets demonstrate that HierPrompt substantially outperforms existing ZS-HTC methods.

1 Introduction

Hierarchical Text Classification (HTC) (Sun and Lim, 2001; Song and Roth, 2014), which assigns text data to categories within a hierarchical taxonomy, is a fundamental task in natural language processing. It plays a crucial role in a wide range of real-world applications, such as product categorization in e-commerce (Cevahir and Murakami, 2016), document organization (Li et al., 2019), web content classification (Dumais and Chen, 2000) and so on. Most of the existing HTC methods rely on labeled data, which is expensive and time-consuming. Moreover, as the hierarchical taxonomies may vary considerably due to evolving categorization standards, it's unrealistic to relabel the data every time. Therefore, interest in Zero-Shot Hierarchical Text Classification (ZS-HTC) arises, which aims to classify texts into hierarchical categories using only the provided hierarchical taxonomy—without any labeled or unlabeled training data (Bongiovanni et al., 2023; Paletto et al., 2024).

A common approach for Zero-Shot Classification (ZSC) is to solve a 1-nearest neighbor problem (Snell et al., 2017; Liu et al., 2022), where the category names are embedded as prototypes and the input text is projected to the same embedding space as prototypes. The similarity between the text embedding and the prototypes is then calculated to determine the category of the text. To leverage the hierarchical information of category names in HTC, USP (Bongiovanni et al., 2023) proposes a similarity propagation technique, which transmits similarity scores from leaf nodes upward through the hierarchy. More recently, HiLA (Paletto et al., 2024) proposes to use Large Language Models (LLM) to generate more fine-grained categories for the leaf categories and then use the generated categories to estimate the similarities for better accuracy. Despite the effectiveness of these methods, they are all built on the assumption of high-quality prototypes, which, however, are not easy to achieve in current methods for the following reasons. 1) Lack of representativeness: The category names defined in the taxonomy are often overly general and imprecise, leading to ambiguous meaning. For instance, 'work' can be interpreted as either 'job' or 'creative content' without providing contextual information. As a result, directly encoding these decontextualized and imprecise category names leads to unrepresentative prototypes. 2) Lack of stylistic information: Category names are usually overly

* Corresponding author

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 3846–3859
November 4-9, 2025 ©2025 Association for Computational Linguistics

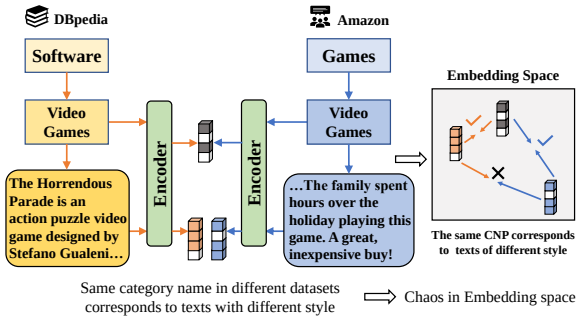

Figure 1: Category name lacks detailed and stylistic information, leading to chaos in embedding space.

simplified phrases that fail to capture fine-grained stylistic information of texts. As illustrated in Fig. 1, the category of 'video games' is included in two datasets. In one dataset, the category is mainly composed of texts describing the features of video games, while in the other, it mostly consists of people's feelings on the game. The prototypes without considering stylistic information are limited in reflecting the true meaning of a category.

To address these problems, we propose **HierPrompt**, a method that makes use of the hierarchical structure of category names to construct **Hier**archy-aware **Prompts** and then use them to instruct LLMs to help produce more representative and informative prototypes. Specifically, we first generalize the proptotype-based framework by introducing the Example Text Prototype (ETP), which is used together with the Category Name Prototype (CNP) to improve the representativeness and preciseness of prototypes. A Maximum Similarity Propagation (MSP) technique is also developed to take the hierarchical structure into account when calculating text-prototype similarity. Then, we prompt the LLMs to contextualize the category names from every level by having them perceive their parents, siblings and children based on the hierarchical taxonomy information. In addition, we also prompt LLMs to generate concrete exemplar texts for every leaf category. The contextualized category names improve the representativeness of CNP, while the generated exemplar texts form the ETP, which supplements the stylistic information under the absence of training data. Experimental results on three public benchmarks with hierarchical labels demonstrate the superiority of HierPrompt.

2 Related Work

Zero-shot Text Classification Considering the high cost of labeled data, many researchers focus on zero-Shot Classification (ZSC). Gera et al. (2022) utilized self-training for ZSC. However, the ZSC scenario they defined is similar to unsupervised scenarios: the text classifier is trained with unlabeled training data. Song and Roth (2014) strictly define the ZSC as a highly constrained scenario where no training data (even unlabeled data) is provided to the system. In this scenario, some studies leveraged in-context learning (Edwards and Camacho-Collados, 2024; Lepagnol et al., 2024), others proposed to solve the textual entailment problem with LLM to obtain labels (Williams et al., 2018; Pàmies et al., 2023). Another approach is to treat the representation of category names as prototypes and transfer the text classification task into the nearest neighbor problem (Snell et al., 2017; Liu et al., 2022). This solution projects the category name and the text into the same representation space with the same encoder, after which calculates the similarity score between the text and all prototypes to determine the category of the text according to the similarity.

Hierarchical Text Classification HTC (Sun and Lim, 2001) assigns multiple labels to a given text, where each label belongs to a level of hierarchical taxonomy. Earlier works (Kowsari et al., 2017; Huang et al., 2019) are highly reliant to supervised information. Other works (Meng et al., 2019; Shen et al., 2021) weaken the supervision to build a hierarchical text classifier with a small number of labeled texts and keywords of leaf categories. Some works (Zhang et al., 2025) use only label hierarchy and unlabelled data to train classifier, but only a little work focused on strict Zero-Shot Hierarchical Text Classification (ZS-HTC). Bongiovanni et al. (2023) proposed a method called Upward Score propagation (USP), which accumulates the similarity scores of children categories according to a certain rule to calculate the parent similarity. The similarity is propagated from the leaf category to the root in the hierarchy. Despite its good performance, it fails to provide hierarchical information for the most leaf categories. Recent advances in Large Language Models (LLMs) offer promising tools to address this challenge. With their strong capabilities in language understanding(Novack et al., 2023; Menon and Vondrick, 2023; Ren et al., 2023), summarization(De Raedt et al., 2023), and generation(Ye et al., 2022; Peng et al., 2024), LLMs have been successfully applied to a wide range of tasks. Paletto et al. (2024) exploited LLM to improve USP: they generate new subclasses for the

original leaf categories with LLM. The enriched hierarchy is then used to perform ZS-HTC with USP technique. However, the aforementioned works of ZS-HTC ignore the potential problems of low quality and diversity of prototypes.

3 Method

In this section, we first introduce the common approach for ZS-HTC, which embeds category names as prototypes for the text 1-nearest neighbor problem. Subsequently, our refined framework with additional prototypes and different similarity calculation technique is illustrated. Finally, we leverage LLMs and introduce hierarchical prototype refinement module, including category name extension and example text generation, with hierarchy-aware prompts.

We follow the notation defined by Paletto et al. (2024), as is described in Table 1. Since the tree-structure hierarchy is considered, for a non-root node ($l \neq 1$), its parent $\uparrow c_j^l$ contains only one category. For a leaf category ($l = L$), its child and descendant categories are both empty set, that is, $\downarrow c_j^l = \emptyset, \Downarrow c_j^l = \emptyset$.

Symbol	Meaning
l	A level in hierarchy, $l = 0, \cdots, L$
c_j^l	the j^{th} category in level l
N_l	The number of categories in level l
$\uparrow c_j^l$	Parent of c_j^l
$\Uparrow c_j^l$	The set of ancestors of c_j^l
$\downarrow c_j^l$	The set of children of c_j^l
$\Downarrow c_j^l$	The set of descendants of c_j^l

Table 1: Notation of Hierarchical taxonomy

3.1 Framework of ZS-HTC

In the ZS-HTC scenario, the hierarchical taxonomy, which contains all category names $\left\{\left\{\mathrm{c}_i^l\right\}_{i=1}^{N_l}\right\}_{l=1}^{L}$ as well as the hierarchical relationship, is provided. By encoding category names at different levels, hierarchical Category Name Prototypes (CNP) can be obtained: $\left\{\left\{\mathrm{CNP}_i^l\right\}_{i=1}^{N_l}\right\}_{l=1}^{L}$. Then, the hierarchical classification problem is transformed into solving the nearest neighbor problem between the text x and the set of hierarchical prototypes:

$$\hat{c}_i^l = \arg\max_{c_j^l} \left(S\left(x, c_j^l\right)\right), \tag{1}$$

where $S\left(x, c_j^l\right)$ denotes cosine similarity between $E(x)$ and CNP_i^l.

Example Text Prototype Since the category name merely comprises the general information of a category, the detailed information and the style of the texts are not included. The ignorance can cause chaos in embedding space where the same CNP should represent text data of total different style. As is depicted in Figure 1, there is a category of "video game" in both DBpedia(Lehmann et al., 2015) and Amazon(Kashnitsky, 2020). But the texts corresponding to video game in DBpedia are neutral and objectively descriptive texts about different games that may include the content, publisher, related history, while texts in Amazon are usually informal and highly subjective reviews of the experience of a purchased game.

To complement the general information of category provided by CNP, it is necessary to introduce additional prototype that capture the stylistic and detailed characteristics of the texts. To this end, we propose to construct prototypes based on example texts—i.e., texts that belong to a given category and reflect the writing style and content typical of the dataset. Compared to category names, example texts offer richer contextual information, which better align with the linguistic style and details of the actual data.

By encoding the example text of each leaf category with $E(\cdot)$, Example Text Prototype (ETP) is obtained:

$$\mathrm{ETP}_i^L = E\left(\mathrm{text}_i^L\right) \tag{2}$$

Prototypes of categories in higher layers ($l < L$) are further obtained by averaging the prototypes of corresponding children:

$$\mathrm{ETP}_i^l = \frac{\sum_{j \in \downarrow c_i^l} \mathrm{ETP}_j}{|\downarrow c_i^l|} \tag{3}$$

With $\left\{\left\{\mathrm{CNP}_i^l\right\}_{i=1}^{N_l}\right\}_{l=1}^{L}$ providing general semantics and $\left\{\left\{\mathrm{ETP}_i^l\right\}_{i=1}^{N_l}\right\}_{l=1}^{L}$ providing detailed semantics, more accurate similarity score can be calculated:

$$S\left(x, c_j^l\right) = S_{CN}\left(x, c_j^l\right) + S_{ET}\left(x, c_j^l\right), \tag{4}$$

where $S_{CN}\left(x, c^l\right), S_{ET}\left(x, c^l\right)$ denotes cosine similarity between $\mathrm{CNP}_i^l/\mathrm{ETP}_i^l$ and $E(x)$.

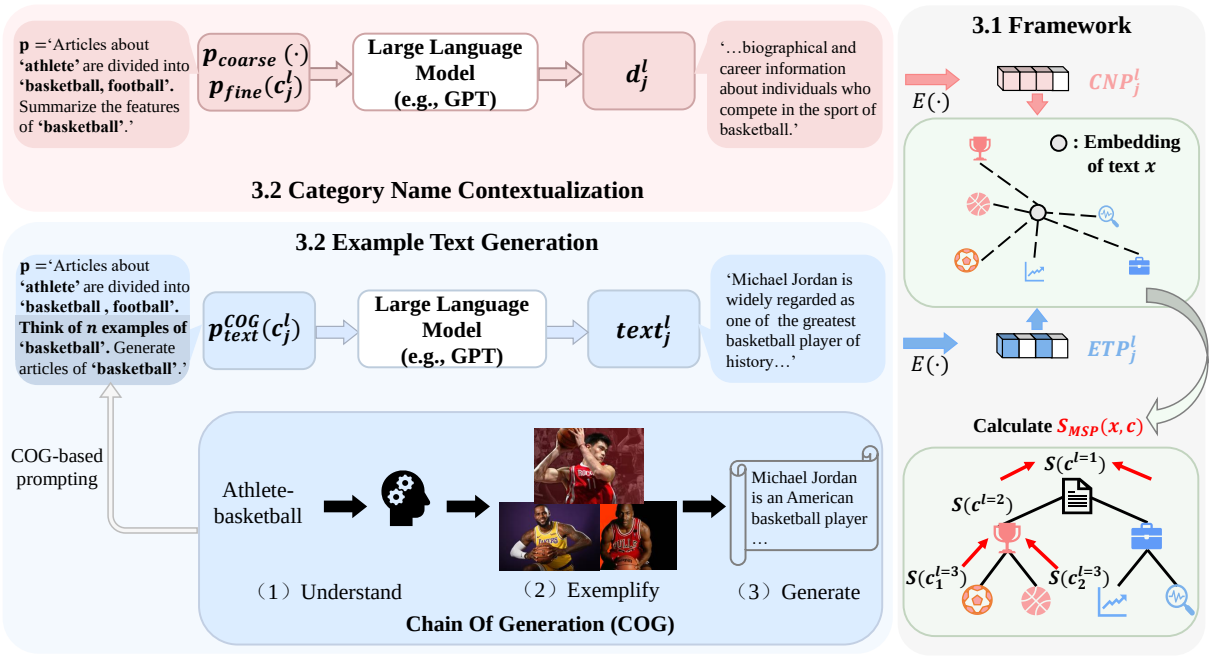

Figure 2: Overview of the proposed HierPrompt. First, the hierarchy-aware prompts are designed Then the prompts serve as input of LLM for Category Name Contextualization and Example Text Generation (Sect.3.2). Finally, the refined prototypes are used to perform classification according to the Maximum Score Propagation technique (Sect. 3.1).

Maximum Similarity Propagation Although the similarity calculation in Equation (4) considers both general and detailed information with CNP and ETP, it completely ignores the hierarchical relationships among categories. As a result, the hierarchical text classification (HTC) task is reduced to a set of independent flat classification problems at each level, thereby failing to exploit the structural dependencies within the hierarchy.

Notably, the hierarchical structure exhibits a form of transitivity: if a text x is semantically related to a lower-level category c_j^l, it is likely to also be related to its parent category $\uparrow c_j^l$. Therefore, we propose Maximum Similarity Propagation(MSP) technique:

$$S_{MSP}\left(x, c_j^l\right)=\begin{cases} S\left(x, c_j^l\right) & l=L \\ S\left(x, c_j^l\right)+S^{\max}\left(\downarrow c_j^l\right) & l<L \end{cases} \tag{5}$$

where $S^{\max}\left(\downarrow c_j^l\right)$ is calculated as:

$$\max_{c \in \downarrow c_j^l}\left(S_{CN}(x, c)\right)+\max_{c \in \downarrow c_j^l}\left(S_{ET}(x, c)\right), \tag{6}$$

which makes $S^{\max}\left(\downarrow c_j^l\right)$ absorb the maximum similarity score of x and the descendant CNP/ETP of c_j^i.

3.2 Hierarchical Prototype Refinement With Hierarchy-Aware Prompts

In zero-shot scenarios, training data is inaccessible, making it impossible to obtain real example data of categories. What's more, the introduction of ETP can't remedy CNP's lack of representativeness, which is caused by : **(1) Inaccurate or over-general category name; (2) Lack of context for category name.** To solve the problem of non-representativeness of CNP and inaccessibility of ETP, we propose to take advantage of the powerful understanding and generating ability of LLMs by prompting them with hierarchy-aware prompts.

Category Name Contextualization To begin with, we focus on level one (L1), which is the most general level with coarsest categories. Intuitively, coarse-level classification should be more accurate than finer one. However, as is demonstrated in Figure 3, the macro-F1 for L1 is 31.6% while it is 63.60% for L3 in DBpedia dataset.

This counterintuitive result is caused by the over-general category name, which blurs the semantics. For instance, a category named 'work' can be understood as 'job' or 'creative production'. It is scarcely possible to determine the correct meaning with a single word both for human being and the encoder, leading to inaccurate prototypes. However,

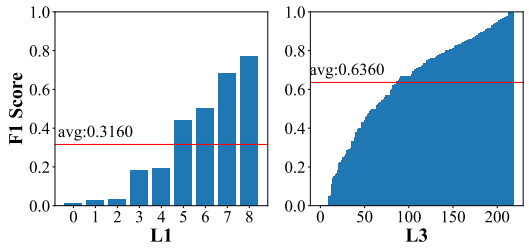

Figure 3: Directly using category name for classification in DBpedia.

with hierarchical taxonomy providing semantic information for category name (*e.g.* "'work' can be subdivided into 'Written Work', 'Periodical Literature','Song', etc."), it's straightforward to conclude the actual meaning of category names (*e.g.* "'work' means 'a range of creative work'"). Therefore, we propose to integrate hierarchical information into the prompt statements to prompt LLM, so that more accurate category name and explanatory context for coarse-grained category can be concluded:

$$P_{\text{coarse}}\left(c_i^1\right) = \text{fill-Template}_{\text{coarse}}\left(\downarrow c_i^1\right), \quad (7)$$

where fill-Template$_{\text{coarse}}\left(\downarrow c_i^1\right)$ = "*A* [*dataset*] *can be classified into* $\left[\downarrow c_i^1\right]$*, please summarize them into one class, give the class name and its corresponding description sentence.*",where [dataset] varies according to different datasets.

For other levels of categories, we also design prompts with corresponding parent and sibling categories to obtain explanatory context:

$$P_{\text{fine}}\left(c_i^l\right) = \text{fill-Template}_{\text{text}}\left(c_i^l, \downarrow\left\{\uparrow c_j^l\right\}\right), \quad (8)$$

where fill-Template$_{\text{text}}\left(c_i^l, \downarrow\left\{\uparrow c_j^l\right\}\right)$ = "[*dataset*] *of* $\left[\uparrow c_j^l\right]$ *can be classified into classes:* $\left[\downarrow\left\{\uparrow c_j^l\right\}\right]$*. Write one sentence to summarize the features of* [*dataset*] *that is classified into the class* c_i^l*.*".

The contextualized hierarchical category names, each of which contains (revised) category name and explanatory context, is obtained by prompting LLM:

$$d_i^l = \begin{cases} LLM(p_{\text{coarse}}(c_i^1)) & l = 1 \\ LLM(p_{\text{fine}}(c_i^l)) & l > 1 \end{cases} \quad (9)$$

which are then encoded into CNP as $\text{CNP}_i^l = E(d_i^l)$.

Example Text Generation In zero-shot scenario, example texts are not available, thus an Example Text Generation module is proposed to generate example texts for ETP. Following the prompt design in Category Name Contextualization module, hierarchical information should be integrated into prompt, which can alleviate the misunderstanding of category names and improve the quality of generation. The prompt for LLM to generate example text of c_i^L can be defined as follows:

$$P_{\text{text}}(c_i^L) = \text{fill-Template}_{\text{text}}(c_i^L \cup \Uparrow c_i^L \cup \downarrow\{\uparrow c_i^L\}), \quad (10)$$

where fill-Template$_{\text{text}}\left(c_i^L \cup \uparrow c_i^L \cup \downarrow\{\uparrow c_i^L\}\right)$ = "*There are* [*dataset*] *about* $\left[\Uparrow c_i^L\right]$*, which can be divided into* $\left[\downarrow\{\uparrow c_i^L\}\right]$*. Please generate a* [*dataset*]*, which can serve as a typical example of the class* $\left[c_i^L\right]$.".

With hierarchical information sufficiently exploited in prompt, the LLM is expected to generate concrete example texts for each category, which shares similar style with the unseen dataset. For example, for the category 'agent-athlete-basketball', an introductory article of a specific basketball athlete should be generated. However, such complicated generation is not an overnight process. The generation of an example text for a category actually contains multiple logical process, forming a Chain Of Generation (COG), as is demonstrated in Figure 2: (1) Understanding category semantics; (2) Exemplifying concrete instance of category; (3) Generating corresponding text of the instance. With the assistance of hierarchical information in prompt as Equation (10), it's possible for LLM to understand category correctly, while there is not any clue to remind LLM to exemplify a concrete instance as step(2) suggests. The break of the logical chain cast a shadow for LLM to generate concrete example text of given category. LLM may even go all the way to generate explanation of category name, which coincides with CNP. To address this problem, we propose to explicitly complemente the COG in the prompt statement , so that LLM can provide desired example texts with concrete examples:

$$P_{\text{text}}^{\text{COG}}(c_i^L) = \text{fill-Template}_{\text{COG}}(c_i^L \cup \Uparrow c_i^L \cup \downarrow\{\uparrow c_i^L\}), \quad (11)$$

which changes the template into: fill-Template$_{\text{COG}}$ $\left(c_i^L \cup \Uparrow c_i^L \cup \downarrow\{\uparrow c_i^L\}\right)$ = "*There are* [*dataset*] *about* $\left[\Uparrow c_i^L\right]$*, which can be divided into* $\left[\downarrow\{\uparrow c_i^L\}\right]$*.* ***Please think of*** n ***specific examples of each fine-grained class. Then generate*** [***dataset***] ***for each fine-grained class.***",where n denotes the number of generated examples.

The generated n examples are encoded separately, after which are averaged to obtain the ETP:

$$\mathrm{ETP}_i^L = \frac{\sum_{j=1}^{n} E\left(\mathrm{text}_j^{c_i^L}\right)}{n}, \tag{12}$$

4 Experiment[1]

4.1 Experimental Setups

Datasets Three public datasets with hierarchical labels are selected to evaluate the proposed method, including: (1) NYT(Tao et al., 2018), which contains 13081 news documents. Its taxonomy two levels with 5,26 categories respectively. (2) DBpedia(Lehmann et al., 2015) contains 50,000 Wikipedia articles, which are divided into three levels according to the DBpedia category hierarchy, with 9, 70 and 219 categories respectively.; (3) Amazon(Kashnitsky, 2020) contains 50,000 Amazon reviews, with a three-level taxonomy of 6, 64 and 522 categories respectively. The language of is casual and informal.

Dataset	L1	L2	L3	DocNum	AvgLen
NYT	5	26	nan	13081	648.13
DBpedia	9	70	219	50000	103.37
amazon	6	64	522	50000	90.29

Table 2: Statistics of Datasets

Baseline Models

- base: The baseline of zero-shot text classification. The 'base-label' denotes performing ZSC with raw category names. The 'base-text' randomly select an example text of each leaf category within the dataset to form prototypes for classification. Noted that 'base-text' is not a strictly zero-shot classification, which is for reference only.
- USP(Bongiovanni et al., 2023): USP proposes a technique called **U**pward **S**core **P**ropagation on the basis of the baseline model, which propagates the finer-grained similarity to the coarse-grained level through hierarchical structure according to certain rules.
- HiLA(Paletto et al., 2024): On the basis of the USP, HiLA (**Hi**erarchical **L**abel **A**ugmentation) utilizes LLM (gpt-3.5-turbo) to expand the categories of leaf nodes.

There is another model called TELEClass (Zhang et al., 2025) similar to our proposed HierPrompt, but since the task is different, it is not included as baseline. The differences between these models are detailed in table 3.

Implementation Details For the category name expansion and example text generation, the Bailian API of Alibaba Cloud platform is used. Specifically, qwen-turbo with the least parameters is used for category name expansion as this task is relatively basic. Qwen-plus with moderate number of parameters is used for example text generation.

Different words are filled for the $[dataset]$ in prompts. For NYT, DBpedia and Amazon, 'news report',' dbpedia article' and 'amazon review' are filled in respectively. The number of example text n in fill-Template$_{\mathrm{COG}}$ is set to 4. Following USP and HiLA, encoder mpnet-all[2] (Song et al., 2020) is used as zero-shot text classification encoder. Macro-F1 score is reported in experiment.

4.2 Main Results

In order to comprehensively evaluate the performance of the proposed HierPrompt on ZS-HTC, we conduct fair experiments on three public benchmarks. The main results are summarized in the table 4.

The row 'ours-label' records the F1 score after performing Category Name Extension modules proposed in Section 3.2. Compared with directly using of category names, F1 score improves greatly after the extension. Especially for level 1 with the coarsest-grained categories, it is 15.18%,50.57% and 18.59% higher than 'base-label' on NYT,DBpedia and Amazon datasets, respectively. This is in line with the analysis at the beginning of Section 3.2 : coarse-grained label names that are too general can produce misleading prototypes and adversely affect classification. After the coarse-grained category names modified and expanded by LLM according to the prompt statements designed with hierarchical structure, the accuracy of the category names is greatly increased, and the classification performance is improved.

The row 'ours-text' records F1 score after performing Example Text Generation module (that is, generating sample text according to fill-Template$_{\mathrm{COG}}$). Except for base-text line results which are slightly lower than using real text in L1

[1]Our code is available at https://github.com/Emily-zero/HierPrompt

[2]https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Model	Task	LLM usage	Hierarchy usage
TELEClass	weakly-supervised	generate keywords and pseudo texts	input of LLM
HiLA	zero-shot	expand hierarchy	input and output of LLM
HierPrompt	zero-shot	contextualiza label name and generate pseudo texts	input of LLM

Table 3: Differences between HierPrompt and similar models

method	**NYT**		**DBpedia**			**Amazon**		
	L1	L2	L1	L2	L3	L1	L2	L3
base-label	70.60	66.13	31.60	33.64	63.60	56.35	26.97	14.22
base-text♣	89.05	58.22	62.72	55.33	63.88	79.57	43.92	17.84
USP	N.A.	66.13	64.70	65.60	62.80	71.20	34.80	17.30
HILA	N.A.	N.A.	76.80	66.00	62.90	76.20	39.30	**24.90**
ours-label	85.78	68.71	76.95	66.47	66.20	82.44	43.39	19.45
ours-text	86.48	63.79	**88.17**	65.90	56.09	**82.69**	47.58	22.09
ours	**87.00**	**69.98**	83.56	**71.32**	**65.78**	82.56	**48.67**	22.69

♣ Not strictly zero-shot classification, for reference only

Table 4: Experimental results of zero-shot hierarchical classification on three datasets

layer of NYT and L3 layer of DBpedia, Most of the generated text performs better even than the randomly selected real examples in 'base-text', proving the effectiveness of the proposed Example Text Generation module.

The row 'ours' represents the final HierPrompt which combines the aforementioned two modules. It's obvious that HierHash surpasses existing SOTA HiLA in almost each level of taxonomy of each dataset. For DBpedia, it exceed HiLA by 5.41%, 5.24% and 4.99% respectively, and Amazon by 6.36% and 9.39% in level 1 and level 2.

4.3 Ablation Study

Four groups of ablation experiments are conducted to prove the effectiveness of the proposed (1) Maximum Similarity Propagation (MSP); (2) Category Name Extension with hierarchy-aware Prompt; (3) Example Text Generation with hierarchy-aware Prompt. The variant prompts in ablation experiments are detailed in appendix B.2.

Maximum Similarity Propagation To explore the effect of our proposed MSP technique, several variant models are proposed as follows:

(i) USP: The USP baseline; **(ii) USP-label**: The similarity propagation technique of USP is implemented to propagate the category names expanded by our proposed category extension module; **(iii) direct-label/text**: Directly use the extended category name/generated example text for classification; **(iv) MSP-label/text**: Our proposed MSP technique is implemented to propagate extended category names/generate sample texts for classification.

method	**DBpedia**		**Amazon**	
	L1	L2	L1	L2
USP	64.70	65.60	71.20	34.80
USP-label	68.60	**69.80**	72.94	36.00
direct-label	38.48	35.37	74.10	30.51
MSP-label	**76.95**	66.47	**82.44**	**43.39**
dirct-text	58.11	60.66	75.89	41.59
MSP-text	**79.21**	**67.94**	**77.71**	**44.54**

Table 5: Ablation: Comparison of the similarity propagation technique in hierarchy of USP and our proposed MSP

Since similarity propagation only affects the non-leaf layer, this set of experiments are conducted on DBpedia and Amazon dataset with three levels of categories and the F1-score of the non-leaf layer(*i.e.*L1,L2). The results are displayed in table 5, where the bold in upper and lower parts indicates the optimal results for category name and example text respectively. It's obvious that the maximum similarity propagation greatly improves the performance for both category name and example text, with the DBPedia dataset improved more than 30%. Compared with the USP method, the MSP method also has great advantages. In DBpedia dataset, When propagating to the level two,

the performance of USP is slightly better than that of MSP, while further propagating up to the level one, the performance of MSP method exceeds that of USP by about 8.35%. In Amazon dataset, the F1-score in level one and two outperforms that of USP by about 9.5% and 7.39%, respectively.

Category Name Contextualization with hierarchy-aware Prompting To explore the effect of the hierarchical information in the category name contextualization module, we here conduct ablation experiments on level one and two of three datasets by comparing the F1-score of extension with/without hierarchical information. In order to reduce the influence of other modules, the results presented here is obtained without utilizing the maximum similarity propagation technique.

For coarse-grained category extension in level one, we directly compare the real category name (base-label) with the category name extended with P_{coarse} (ours-hier). As is displayed in 4, compared with base-label, there is a large increase in F1 score for each dataset in ours-hier.

For the fine-grained category extension, we compare the F1-score of category extension in level two obtained by prompting LLM with $P_{\text{fine}}^{\text{direct}}$ (denoted as 'ours-w/hier'), which ignore the hierarchical information, and with P_{fine} (denoted as 'ours-hier').

As is displayed in Figure 4, 'ours-hier' performs better than 'base-label' and 'ours-w/hier', which proves the importance of integrating hierarchical information into prompts.

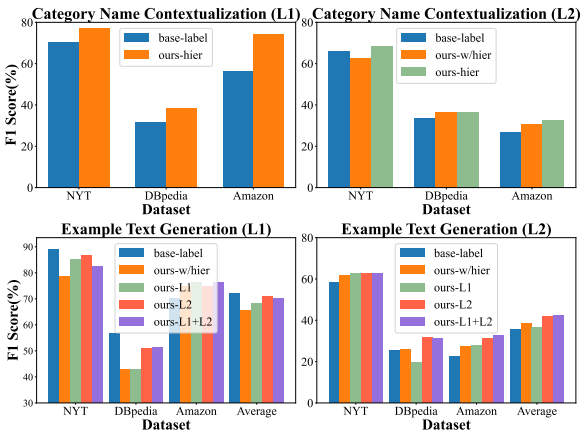

Figure 4: Ablation: The Effect of utilizing hierarchical information in prompts in Category Name Extension (Row 1) and Example Text Generation (Row 2).

Example Text Generation with hierarchy-aware Prompting To explore the effectiveness of integrating hierarchical information in prompts for example text generation module, experiments are conducted in level one and two in three datasets. Five variant models are designed as follows:

(i) base-text: Randomly select a document from the dataset for each category in L2 as its corresponding example text; **(ii)ours-w/hier**: Prompting LLM with non-hierarchical prompts $P_{\text{text}}^{\text{direct}}$ to generate example texts; **(iii)ours-L1**: Prompting LLM with $P_{\text{text}}^{L_1}(c_i^L)$ which integrates coarse-grained information in L1. **(iv)ours-L2**: Prompting LLM with $P_{\text{text}}^{L_2}(c_i^L)$ which integrates fine-grained information in L2. **(v) ours-L1+L2**: Prompting LLM with P_{text}, which includes hierarchical information from all levels.

In second row of Figure 4, the L1 and L2 F1-score of the five variants are demonstrated separately. In general, after considering the hierarchical information in prompts, the example texts generated by using either the coarse-grained information ('ours-L1') or the fine-grained information ('ours-L2') is better than 'ours-w/hier' without considering any hierarchical information. In the meantime, considering more complete hierarchical information ('ours-L1+L2') is generally better than other variants.

Prompt	DBpedia			Amazon		
	L1	L2	L3	L1	L2	L3
P_{text}	64.60	**61.57**	51.19	76.87	43.13	21.91
P'_{text}-avg	59.06	58.06	**57.28**	76.06	41.95	21.89
$P_{\text{text}}^{\text{COG}}$-avg	**67.17**	60.13	56.09	**77.58**	**43.43**	**22.69**
P_{text}*	82.29	65.22	51.19	77.73	45.47	21.91
P'_{text}-avg*	79.70	65.69	**57.28**	76.70	46.03	21.89
$P_{\text{text}}^{\text{COG}}$-avg*	**88.17**	**65.90**	56.09	82.69	**47.58**	**22.69**

* Apply MSP for similarity calculation

Table 6: Ablation: The Effect of COG-prompting for Example Text Generation

Example Text Generation with COG-based Prompting In the ablation experiment in previous paragraph, we demonstrated the importance of incorporating hierarchical information into prompts for example text generation. Here we further demonstrate the importance of integrating Chain-Of-Generation(COG) into prompts. Since the prompts with COG generates multiple example texts, $P'_{\text{text}}(c_i^L)$ that does not consider COG but generates multiple example texts is added in this experiment. A total of three variants are considered: **(i)** $P_{\textbf{text}}$: Generating 1 piece of example text for each leaf category with hierarchy-aware prompt P_{text}; **(ii)** $P'_{\textbf{text}}$: Generating n pieces

of example texts with hierarchy-aware prompt P'_{text}; **(iii)** $P^{\textbf{COG}}_{\textbf{text}}$: Generating n pieces of example text with Hierarchical-structure-and-COG-based prompt $P^{\text{COG}}_{\text{text}}$.

Experiments on two three-level datasets, DBpedia and Amazon, are conducted for the five variants, as is demonstrated in Table 6. In DBpedia dataset, compared with P_{text}, $P^{\text{COG}}_{\text{text}}$has advantages in L1 and L3. But its advantage is not obvious compared with P'_{text}. In Amazon dataset, $P^{\text{COG}}_{\text{text}}$ has a significant performance improvement over the other two methods regardless of whether the MSP technique is implemented. On the whole, the generation with COG has general advantages.

5 Conclusion

In this paper, HierPrompt, a method to leverage hierarchy-aware prompts to instruct LLM for producing more representative and informative prototypes in ZS-HTC task, is proposed. Specifically, the ZSC framework is refined by introducing Example Text Prototype (ETP) on the basis of Category Name Prototype (CNP) and by designing Maximum Similarity Propagation. Then, with the help of LLM and hierarchy-aware prompts, the category names are contextualized and the example texts are generated, forming the refined prototypes. Experiment on three public benchmark demonstrated the superiority in ZS-HTC of HierPompt.

Limitations

The LLM used in our work is assumed to have certain knowledge about the taxonomy. However, it is unclear weather the LLM really has correct knowledge about taxonomy, especially when the taxonomy is highly professional. The hallucinations of LLM can harm the quality of the contextualized category name and the generated example texts, further deteriorating the performance of the model. What's more, HierPrompt mainly refines the hierarchical prototypes in text level: it contextualizes category names and generate example texts, so that the quality of texts encoded into prototypes are improved, but ignores the adjustment of prototypes in embedding space.

Therefore, our future work will focus on:(1) Adding explicit mechanisms (*e.g.*, self-consistency, self-correctness) to harness LLM hallucinations; (2) Improving hierarchical prototypes on both textual space and embedding space.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62276280), Guangzhou Science and Technology Planning Project (No. 2024A04J9967).

References

Lorenzo Bongiovanni, Luca Bruno, Fabrizio Dominici, and Giuseppe Rizzo. 2023. Zero-shot taxonomy mapping for document classification. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, SAC '23, page 911–918, New York, NY, USA. Association for Computing Machinery.

Ali Cevahir and Koji Murakami. 2016. Large-scale multi-class and hierarchical product categorization for an E-commerce giant. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 525–535, Osaka, Japan. The COLING 2016 Organizing Committee.

Maarten De Raedt, Fréderic Godin, Thomas Demeester, and Chris Develder. 2023. IDAS: Intent discovery with abstractive summarization. In *Proceedings of the 5th Workshop on NLP for Conversational AI (NLP4ConvAI 2023)*, pages 71–88, Toronto, Canada. Association for Computational Linguistics.

Susan Dumais and Hao Chen. 2000. Hierarchical classification of web content. *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*.

Aleksandra Edwards and Jose Camacho-Collados. 2024. Language models for text classification: Is in-context learning enough? In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 10058–10072, Torino, Italia. ELRA and ICCL.

Ariel Gera, Alon Halfon, Eyal Shnarch, Yotam Perlitz, Liat Ein-Dor, and Noam Slonim. 2022. Zero-shot text classification with self-training. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1107–1119, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Wei Huang, Enhong Chen, Qi Liu, Yuying Chen, Zai Huang, Yang Liu, Zhou Zhao, Dandan Zhang, and Shijin Wang. 2019. Hierarchical multi-label text classification: An attention-based recurrent network approach. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*.

Yury Kashnitsky. 2020. Hierarchical text classification.

Kamran Kowsari, Donald E. Brown, Mojtaba Heidarysafa, K. Meimandi, Matthew S. Gerber, and Laura E. Barnes. 2017. Hdltex: Hierarchical deep

learning for text classification. *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 364–371.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Pierre Lepagnol, Thomas Gerald, Sahar Ghannay, Christophe Servan, and Sophie Rosset. 2024. Small language models are good too: An empirical study of zero-shot classification. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 14923–14936, Torino, Italia. ELRA and ICCL.

Keqian Li, Shiyang Li, Semih Yavuz, Hanwen Zha, Yu Su, and Xifeng Yan. 2019. Hiercon: Hierarchical organization of technical documents based on concepts. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 379–388.

Wenfu Liu, Jianmin Pang, Nan Li, Feng Yue, and Guangming Liu. 2022. Few-shot short-text classification with language representations and centroid similarity. *Applied Intelligence*, 53.

Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2019. Weakly-supervised hierarchical text classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6826–6833.

Sachit Menon and Carl Vondrick. 2023. Visual classification via description from large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Zachary Novack, Julian McAuley, Zachary Lipton, and Saurabh Garg. 2023. Chils: zero-shot image classification with hierarchical label sets. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Lorenzo Paletto, Valerio Basile, and Roberto Esposito. 2024. Label augmentation for zero-shot hierarchical text classification. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7697–7706, Bangkok, Thailand. Association for Computational Linguistics.

Marc Pàmies, Joan Llop, Francesco Multari, Nicolau Duran-Silva, César Parra-Rojas, Aitor Gonzalez-Agirre, Francesco Alessandro Massucci, and Marta Villegas. 2023. A weakly supervised textual entailment approach to zero-shot text classification. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 286–296, Dubrovnik, Croatia. Association for Computational Linguistics.

Letian Peng, Yi Gu, Chengyu Dong, Zihan Wang, and Jingbo Shang. 2024. Text grafting: Near-distribution weak supervision for minority classes in text classification. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3741–3752, Miami, Florida, USA. Association for Computational Linguistics.

Zhiyuan Ren, Yiyang Su, and Xiaoming Liu. 2023. Chatgpt-powered hierarchical comparisons for image classification. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Jiaming Shen, Wenda Qiu, Yu Meng, Jingbo Shang, Xiang Ren, and Jiawei Han. 2021. TaxoClass: Hierarchical multi-label text classification using only class names. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4239–4249, Online. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4080–4090, Red Hook, NY, USA. Curran Associates Inc.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Preprint*, arXiv:2004.09297.

Yangqiu Song and Dan Roth. 2014. On dataless hierarchical text classification. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, page 1579–1585. AAAI Press.

Aixin Sun and Ee-Peng Lim. 2001. Hierarchical text classification and evaluation. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 521–528.

Fangbo Tao, Chao Zhang, Xiusi Chen, Meng Jiang, Tim Hanratty, Lance Kaplan, and Jiawei Han. 2018. Doc2cube: Allocating documents to text cube without labeled data. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 1260–1265. IEEE.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022. ZeroGen: Efficient zero-shot learning via

dataset generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11653–11669, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yunyi Zhang, Ruozhen Yang, Xueqiang Xu, Rui Li, Jinfeng Xiao, Jiaming Shen, and Jiawei Han. 2025. Teleclass: Taxonomy enrichment and llm-enhanced hierarchical text classification with minimal supervision. *Preprint*, arXiv:2403.00165.

A Supplementary Results

The supplementary experimental results are reported here, including the detailed value of F1 score in ablation studies and the sensitivity analysis.

A.1 Ablation

4.3 demonstrated the ablation experiment of the proposed HierPrompt, in which the F1 score of Category Name Extension and Example Text Generation are displayed by bar charts. This part complements the numerical values of the specific experimental results of these two experiments.

Detailed Results of Category Name Extension The experimental results of the coarse-grained and fine-grained Category Name Extension module are demonstrated in table 7 and 8. The macro-F1 score (maF1) and micro-F1 score (miF1) of 'base-label' which directly uses the category name, and 'ours-hier' which leverages the hierarchy-aware prompt to extend the category name, are included. Δ is the numerical difference between 'ours-hier' and 'base-label'. For the fine-grained label extension module, the results of 'ours-w/hier', which uses prompt without hierarchical information, are also displayed.

method	NYT		DBpedia		Amazon	
	miF1	maF1	miF1	maF1	miF1	maF1
base-label	88.72	70.60	46.90	31.60	59.02	56.35
ours-hier	92.88	77.16	50.56	38.48	73.20	74.10
Δ	4.16	6.56	3.66	6.88	14.18	17.75

Table 7: Ablation: Coarse-grained (L1) Category Name Extension

Detailed Results of Example Text Generation Five variant models are designed to perform ablation experiments on the Example Text Generation module, and the comparison results of coarse-grained (L1) and fine-grained (L2) are listed in table 9 and 10, including the macro-F1 score (maF1) and micro-F1 score (miF1).

method	NYT		DBpedia		Amazon	
	miF1	maF1	miF1	maF1	miF1	maF1
base-label	84.29	66.13	36.76	33.64	35.64	26.97
ours-w/hier	73.09	62.61	39.34	36.65	36.16	30.63
Δ	-11.20	-3.52	2.58	3.01	0.52	3.66
ours-hier	84.72	68.71	37.16	36.69	39.86	32.74
Δ	0.43	2.58	0.40	3.05	4.22	5.77

Table 8: Ablation: Fine-grained (L2) Category Name Extension

method	NYT		DBpedia		Amazon	
	miF1	maF1	miF1	maF1	miF1	maF1
base-text	95.57	89.05	74.94	56.81	71.06	70.19
ours-w/hier	61.92	78.49	56.60	42.90	75.50	74.77
ours-L1	94.27	85.04	59.56	42.97	76.70	**76.40**
ours-L2	**94.97**	**86.72**	71.34	51.09	75.48	74.91
ours-L1+L2	93.92	82.44	**72.58**	**51.41**	**76.94**	76.31

Table 9: Ablation: Example Text Generation with hierarchy-aware Prompts (L1)

A.2 Sensitivity Analysis

In HierPrompt, both the category name prototype CNP and the example text prototype ETP are changed according to the output of LLM. In order to explore the influence of different LLM, sensitivity analysis is further carried out. This experiment utilizes three versions of Ali Cloud Qwen model with different scale of parameters {qwen-turbo,qwen-plus,qwen-max} on DBpedia dataset. The three models are utilized for category name extension and example text generation with P_{coarse}, P_{fine} and P_{text}) respectively.

The results are presented in Figure 5. For the category name extension module, F1-score varies little with different LLM. This may be because the task of understanding and extending categories is too simple to rely on complicated LLM. However, in the example text generation module, there is a large performance difference: the performance of qwen-turbo and qwen-max decreases with the increase of the generality of the category(L3→L1), while that of qwen-plus is the opposite: with the increase of the generality of the category (L3→L1), the performance gradually increases.

A.3 Computational Budget

The query of LLM is required in the Category Name Contextualization and Example Text Generation. The cost of LLM querying grows linearly

method	NYT		DBpedia		Amazon	
	miF1	maF1	miF1	maF1	miF1	maF1
base-text	65.19	58.22	25.56	25.62	25.44	22.53
ours-w/hier	68.32	61.92	29.52	25.78	37.40	27.14
ours-L1	**78.73**	**62.66**	26.22	19.67	35.18	27.95
ours-L2	70.66	62.60	33.34	**31.88**	39.88	31.22
ours-L1+L2	75.35	62.62	**33.54**	31.21	**40.82**	**32.81**

Table 10: Ablation: Example Text Generation with hierarchy-aware Prompts (L2)

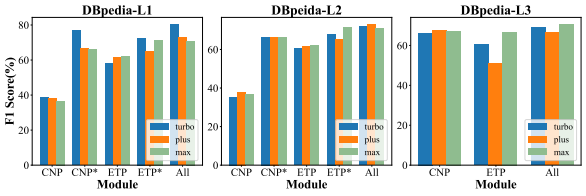

Figure 5: Sensitivity Analysis: Effect of different LLM for the proposed HierPrompt

with the number of categories in the hierarchical taxonomy ($\mathcal{O}(n)$).

In our experiment, the largest dataset Amazon requires around 120 seconds and 6000 seconds in non-batch querying for Category Name Contextualization and Example Text Generation respectively. During inference, it requires less than 5 seconds. All experiments are conducted with an GeForce RTX 2080 GPU.

B Prompt Design

In this section, we demonstrated the detailed prompts used in this paper, including in our proposed HierPrompt and variant models of ablation study.

B.1 Prompts in HierPrompt

In the proposed HierPrompt, three hierarchy-aware prompts are leveraged. In Category Name Extension module, there are two prompting functions, $P_{\text{coarse}}\left(c_i^1\right)$ for categories in level 1 and $P_{\text{fine}}\left(c_i^l\right)$ for other fine-grained categories. In Example Text Generation module, the Chain-Of-Generation (COG) is also integrated into the prompts, with the prompting function $P_{\text{text}}^{\text{COG}}\left(c_i^L\right)$. The three prompting function as well as corresponding examples are demonstrated in table 11.

B.2 Prompts in Ablation Study

There are multiple variant prompts proposed in 4.3:

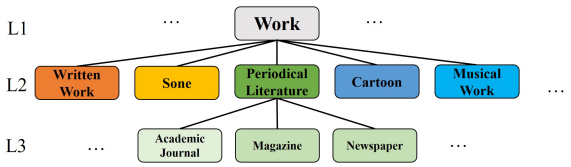

Figure 6: The hierarchical taxonomy in example of prompt design

- $P_{\text{text}}^{\text{direct}}$: $P_{\text{fine}}^{\text{direct}}$ is the prompt of 'ours-w/hier' in the part 'Category Name Extension with hierarchy-aware Prompting', which prompts LLM without leveraging hierarchical taxonomy to extend categories in levels where $l > 1$.
- $P_{\text{text}}^{\text{direct}}$: $P_{\text{text}}^{\text{direct}}$ is the prompt of 'ours-w/hier' in the part 'Example Text Generation with hierarchy-aware Prompting', which prompts LLM without leveraging hierarchical taxonomy to generate example texts.
- $P_{\text{text}}^{L_1}(c_i^L)$: $P_{\text{text}}^{L_1}(c_i^L)$ is the prompt of 'ours-L1' in the part 'Example Text Generation with hierarchy-aware Prompting'. $P_{\text{text}}^{L_1}(c_i^L)$ integrates coarse-grained information in level 1.
- $P_{\text{text}}^{L_2}(c_i^L)$: $P_{\text{text}}^{L_2}(c_i^L)$ is the prompt of 'ours-L2' in the part 'Example Text Generation with hierarchy-aware Prompting'. It integrates fine-grained information in level 2.
- P_{text}: P_{text} is the prompt of variant model 'ours-L1+L2' in the part 'Example Text Generation with hierarchy-aware Prompting', which includes hierarchical information from all levels. It also appears in the part 'Example Text Generation with COG-based-prompting'.
- $P'_{\text{text}}(c_i^L)$: $P'_{\text{text}}(c_i^L)$ is the variant prompt in the part 'Example Text Generation with COG-based-prompting', which generates n pieces of example texts with hierarchy-aware prompts without integrating the Chain-Of-Generation (COG).

All of the aforementioned prompts are summarized in table 12.

Function	Prompt	
$P_{\text{coarse}}\left(c_i^1\right)$	template	A [dataset] can be classified into $\left[\downarrow c_i^1\right]$, please summarize them into one class, give the class name and its corresponding description sentence.
	example	An **DBpedia article** can be classified into **'Written Work', 'Periodical Literature', 'Song', 'Cartoon', 'Musical Work', 'Software', 'Database', 'Comic'**, please summarize them into one class, give the class name and its corresponding description sentence.
$P_{\text{fine}}\left(c_i^l\right)$	template	[dataset] of $\left[\uparrow c_j^l\right]$ can be classified into classes: $\left[\downarrow \left\{\uparrow c_j^l\right\}\right]$. Write one sentence to summarize the features of [dataset] that is classified into the class c_i^l.
	example	An **DBpedia article** can be classified into classes **'Written Work', 'Periodical Literature', 'Song', 'Cartoon', 'Musical Work', 'Software', 'Database', 'Comic'**, Write one sentence to summarize the features of **DBpedia article** that is classified into the class Written Work.
$P_{\text{text}}^{\text{COG}}\left(c_i^L\right)$	template	There are [dataset] about $\left[\Uparrow c_i^L\right]$, which can be divided into $\left[\downarrow \{\uparrow c_i^L\}\right]$. Please think of n specific example of each fine-grained class. Then generate [dataset] for each fine-grained class, each example should be raw texts without special format and less than 300 words. Answer my questions with a python dictionary containing the lists of typical sentences for every classes. Use double quotes for strings.
	example	There are **DBpedia article** about **"work, periodical literature"**, which can be divided into **'Academic Journal', 'Magazine', 'Newspaper'**. Please think of **2** specific example of each fine-grained class. Then generate **DBpedia article** for each fine-grained class, each example should be raw texts without special format and less than 300 words. Answer my questions with a python dictionary containing the lists of typical sentences for every classes. Use double quotes for strings.

Table 11: Prompt Design of HierPrompt

Function	Prompt	
$P_{\text{fine}}^{\text{direct}}\left(c_i^l\right)$	template	"Please describe the class $\left[c_i^l\right]$ in a dataset of [dataset] with one sentence."
	example	Please describe the class **Periodical Literature** in a dataset of **DBpedia article** with one sentence.
$P_{\text{text}}^{\text{direct}}\left(c_i^L\right)$	template	Please generate [dataset], which can serve as a typical example of the class $\left[c_i^L\right]$. The response should be raw texts without special format.
	example	Please generate **DBpedia article**, which can serve as a typical example of the class **'Academic Journal'**. The response should be raw texts without special format.
$P_{\text{text}}^{L_1}(c_i^L)$	template	Please generate [dataset], which can serve as a typical example of the class $\{\uparrow c_i^L\} - c_i^L$. The response should be raw texts without special format.
	example	Please generate **DBpedia article**, which can serve as a typical example of the class **"periodical literature-Academic Journal"**. The response should be raw texts without special format.
$P_{\text{text}}^{L_2}(c_i^L)$	template	There are [dataset] about $\downarrow \{\uparrow c_i^L\}$. Please generate a [dataset], which can serve as a typical example of the class c_i^L. The response should be raw texts without special format.
	example	There are **DBpedia article** about **'Academic Journal', 'Magazine', 'Newspaper'**. Please generate a **DBpedia article**, which can serve as a typical example of the class **'Academic Journal'**. The response should be raw texts without special format.
$P_{\text{text}}\left(c_i^L\right)$	template	There are [dataset] about $\left[\Uparrow c_i^L\right]$, which can be divided into $\left[\downarrow \{\uparrow c_i^L\}\right]$. Please generate a [dataset], which can serve as a typical example of the class $\left[c_i^L\right]$.The response should be raw texts without special format.
	example	There are **DBpedia article** about **"work, periodical literature"**, which can be divided into **'Academic Journal', 'Magazine', 'Newspaper'**. Please generate a **DBpedia article**, which can serve as a typical example of the class **'Academic Journal'**.The response should be raw texts without special format.
$P'_{\text{text}}(c_i^L)$	template	There are [dataset] about $\uparrow c_i^L$, which can be divided into $\downarrow \{\uparrow c_i^L\}$. Please generate n typical sentences in [dataset] for each fine-grained class, each example should be raw texts without special format and less than 300 words. Answer my questions with a python dictionary containing the examples for every classes. Use double quotes for strings.
	example	There are **DBpedia article** about **"periodical literature"**, which can be divided into **'Academic Journal', 'Magazine', 'Newspaper'**. Please generate n typical sentences in **DBpedia article** for each fine-grained class, each example should be raw texts without special format and less than 300 words. Answer my questions with a python dictionary containing the examples for every classes. Use double quotes for strings.

Table 12: Prompt Design in Ablation Study