RelaClass: A Residual Tree with LLM-Augmented Prototypes for Hierarchical Text Classification

Wenting He, Qiang Yan, Bing Li, Huaming Liao, Jiafeng Guo, Xueqi Cheng
State Key Laboratory of AI Safety; Institute of Computing Technology, Chinese Academy of Sciences;
University of Chinese Academy of Sciences
{hewenting01, yanqiang, libing16, lhm, guojiafeng, cxq}@ict.ac.cn

***Abstract*—Hierarchical text classification (HTC) aims to assign documents to a predefined hierarchical taxonomy but often requires extensive manually labeled data for training. Although large language models (LLMs) show promise in zero-shot settings, they struggle to capture fine-grained hierarchical dependencies due to limitations in prompt expressiveness and structural inductive bias. To address these limitations, we propose RelaClass (A Residual Tree Framework with LLM-Augmented Prototypes), a novel weakly supervised framework that integrates the semantic richness of LLMs with an efficient, taxonomy-aware residual tree model, which relies solely on class names provided by the taxonomy as supervision. RelaClass addresses two core challenges: (1) structural alignment: by reformulating the taxonomy into a hierarchy of codebooks that recursively decompose document representations into taxonomic residuals, enabling coarse-to-fine classification; and (2) semantic anchoring: by leveraging LLMs to generate discriminative prototypes that provide high-quality supervisory signals without manual labeling. Experiments on two benchmark datasets show that RelaClass outperforms state-of-the-art weakly-supervised methods and zero-shot LLM prompting approaches. Code and data are available at https://github.com/hexiaoting/RelaClass.**

***Index Terms*—residual tree, hierarchical text classification, LLM**

I. Introduction

Hierarchical text classification (HTC) is a fundamental task with broad applications, including query classification [1], scientific data categorization [2], and product tagging on e-commerce platforms. It involves assigning a document to a complete path from root to leaf within a predefined tree-structured taxonomy. Unlike flat classification, HTC leverages the hierarchical organization of labels to capture fine-grained categorical dependencies, making it especially valuable for organizing large-scale unstructured corpora.

Traditional approaches to HTC typically employ hierarchy-aware encoders to enhance text representation by incorporating label dependencies. For instance, HiAGM [3] employs Graph Neural Networks (GNNs) to explicitly model label correlations, while HiTIN [4] compresses the label hierarchy into a coding tree via structural entropy. Although effective, these methods rely heavily on large-scale manually labeled data, which is expensive and often impractical to acquire.

To mitigate the annotation bottleneck, recent work has turned to zero-shot capabilities of Large Language Models (LLMs). Methods such as HierPrompt [5] synthesize category-specific training samples using carefully designed prompts, and TELEClass [6] employs LLMs to enrich taxonomy semantics and generate class annotations. However, effectively applying LLMs to HTC remains difficult: prompting with the entire hierarchy often dilutes structural information and degrades performance, whereas decomposing the taxonomy layer-by-layer loses the global context to disambiguate similar categories, leading to suboptimal results.

Inspired by the Residual Quantized Variational Autoencoder (RQ-VAE) [7], which learns hierarchical coarse-to-fine representations in a self-supervised manner, we observe that its recursive residual mechanism naturally aligns with the taxonomic progression of HTC. However, adapting unconstrained semantic quantization for taxonomy-governed classification presents two key challenges: (1) *Structural Alignment.* How can the unsupervised process of semantic quantization be strictly aligned with a predefined, rigid, and often unbalanced label hierarchy? (2) *Semantic Anchoring.* In the absence of large-scale annotations, what constitutes an effective alignment criterion to supervise the learning for each category?

To address these challenges, we propose RelaClass, a novel Residual Tree-based framework designed for weakly supervised HTC. To strictly align with the predefined taxonomy, we structure the taxonomy as a hierarchy of local codebooks embedded within a Residual Tree. Each non-leaf node contains a trainable codebook, where each code represents an immediate child category. During classification, the model recursively matches and subtracts the selected code from the document embedding, transforming it into a sequence of residuals. This structural alignment is further reinforced by a classification alignment loss that pulls each code closer to its corresponding category prototype, thereby enhancing the distinction between semantically close sibling categories. To construct semantic anchors, we employ an LLM to bridge the supervision gap. By generating discriminative descriptions for similarity matching, followed by cross-verification through LLM-based annotation, the LLM selects high-quality representative samples from which semantic anchor prototypes [8] are constructed for each category. These anchors provide essential supervisory signals to guide the model's latent space, enabling robust hierarchical classification without manual labeling.

Our contributions are summarized as follows: (1) We propose RelaClass, a residual tree framework that performs hierarchical classification via a layer-wise residual matching mechanism, enabling coarse-to-fine semantic refinement under

weak supervision. (2) We integrate LLMs as a semantic engine to generate discriminative prototypes and introduce a category alignment loss to enhance semantic representativeness and classification accuracy. (3) Experiments on two datasets demonstrate the superiority of RelaClass. It outperforms the state-of-the-art weakly supervised approach by 1.7% in Micro-F1 and 5.7% in Accuracy@3 on the Amazon dataset.

II. Related Work

A. Hierarchical Text Classification

Hierarchical text classification (HTC) assigns documents to nodes in a tree-structured taxonomy, a task complicated by large-scale hierarchies and imbalanced label distributions. Existing methods can be broadly categorized into local and global approaches, based on how the hierarchical label structure is modeled [3]. Local approaches train independent classifiers for individual nodes [9] or hierarchy levels [10]. These methods often require a large number of parameters and tend to focus on local contextual cues, potentially overlooking global hierarchical dependencies. This may lead to inconsistent predictions across related categories. In contrast, global approaches model the entire label taxonomy within a unified framework [11], utilizing techniques such as reinforcement learning [12], GNN-based encoders [3] or coding tree [4]. Zhang et al. [13] propose a probability-guided contrastive learning framework that improves label differentiation by constructing high-quality samples from both sequence and node perspectives. Despite their effectiveness, these methods largely rely on manually labeled data. Weakly supervised HTC has gained traction as a means to reduce annotation dependency [14]. WeSHClass [14] uses a small set of keywords to generate pseudo-documents and iteratively refines the classifier via self-training. TaxoClass [15] uses only class names and employs textual entailment to generate pseudo-labels, but often overlooks nuanced semantic structures and is vulnerable to label noise.

B. LLM for data synthesis and annotation

Recent studies leverage large language models (LLMs) to mitigate the annotation bottleneck in HTC. For instance, HierPrompt [5] synthesizes taxonomy-aware training samples, while TELEClass [6] uses LLMs for core classes annotation. Other approaches include path-traversal annotation strategies [16] and iterative refinement of LLM-initialized classifiers via adaptive boosting [17]. Despite these advancements, directly applying LLMs to HTC remains challenging. Prompting with full label hierarchy often obscures structural relationships, while processing the taxonomy layer by layer fragments the global semantic context needed to discriminate between fine-grained sibling categories.

C. RQ-VAE as a semantic indexer

The Residual-Quantized Variational AutoEncoder (RQ-VAE) [7] recursively quantizes residuals via multiple codebooks to compress high-dimensional data into discrete codes, and has been adapted for semantic indexing [18], [19]. Its coarse-to-fine quantization mechanism naturally aligns with hierarchical structure. However, its global codebook-sharing mechanism and unsupervised training make it unsuitable for HTC, as learned dimensions often misalign with predefined hierarchies.

III. Problem Definition

The task of minimally supervised hierarchical text classification (HTC) is defined as training a text classifier using only category names from a predefined taxonomy as supervision [15]. The objective is to assign each document to a specific path from the root to a leaf node within the taxonomy, representing a coherent semantic progression.

Formally, the HTC task involves a document collection $D = \{d_1, \ldots, d_{|\mathcal{D}|}\}$ and a tree-structured label taxonomy $L = (C, H)$. Here, each node $c_i \in C$ represents a category identified by its unique class name, and each directed edge $(c_i, c_j) \in H$ indicates a parent-child relationship. The goal is to learn a mapping function $f : d \rightarrow P$, where $P = [c_0, c_1, \ldots, c_k]$ denotes a valid label path such that (1) c_0 is the root node of L, (2) c_k is a leaf node, and (3) $(c_i, c_{i+1}) \in H$ for all $0 \leq i < k$.

IV. Methodology

The proposed RelaClass framework, as shown in Figure 1, comprises three core components: (1) a text encoder that projects input documents into semantic embeddings; (2) a residual tree model, a tree-structured module explicitly aligned with the hierarchical label taxonomy, which utilizes residual vector computation for coarse-to-fine representation learning. By restricting classification candidates to relevant child nodes at each level, the model reduces the search space and improves both efficiency and accuracy; and (3) an LLM-enhanced prototype generation module that integrates multiple strategies, including discriminative semantic enhancement, automated annotation, and synthetic sample generation, all aimed at deriving representative prototype vectors for each category. The following subsections detail the architecture of each component and our overall training strategy.

A. Text Encoder

We use a pre-trained text encoder (e.g., BERT [20]) to convert document d content features (e.g., titles, descriptions) into semantic embeddings $\boldsymbol{x} \in \mathbb{R}^d$, which are fed into the residual tree model for hierarchical classification.

B. Residual Tree Model

Conventional hierarchical classification methods often suffer from two major issues: exponentially growing search space with taxonomy depth, and difficulty in distinguishing between semantically fine-grained sibling categories. In response to these challenges, we introduce a residual tree model that explicitly embeds the label hierarchy into a multi-level residual encoding architecture. This model narrows the search space at each level to only the relevant child nodes and uses residual vectors to progressively refine semantic representations, enabling coarse-to-fine discrimination with high accuracy and efficiency.

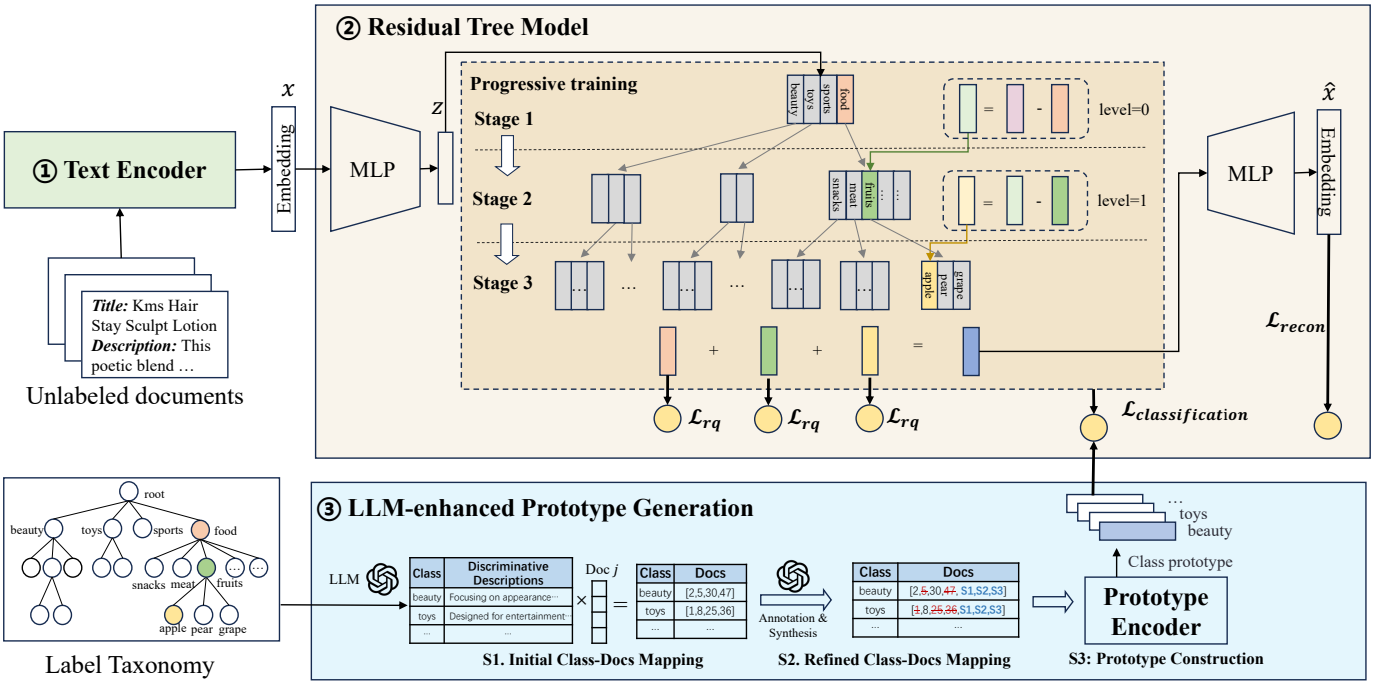


Fig. 1: Overall structure of the RelaClass framework. RelaClass consists of a text encoder, a residual tree model, and a prototype generation module that leverages LLMs to generate high-quality category prototypes for model training. Within the mapping tables, red struck-through numbers indicate documents filtered out via LLM-based annotation consistency, while blue identifiers denote LLM-generated synthetic samples.

Algorithm 1 Residual Tree Classification Process

Input: Residual Tree T, mapping function M, input embedding $\boldsymbol{x}$.
Output: Hierarchical label path $P = \{y_0, y_1, \ldots, y_L\}$.

$\mathbf{z} \leftarrow \mathcal{E}(\mathbf{x})$ ▷ Project to latent space
$v \leftarrow r$, $\mathbf{r}_0 \leftarrow \mathbf{z}$, $P \leftarrow \emptyset$
$i \leftarrow 0$
while $v \neq \emptyset$ **do**
  $\mathbf{C}_v = \{\mathbf{c}_{v,k}\}_{k=1}^{n_v} \leftarrow \text{GetCodebook}(v)$
  $k^* = \arg\min_{k \in \{1,\ldots,n_v\}} \|\mathbf{r}_i - \mathbf{c}_{v,k}\|_2$
  $y_i \leftarrow M(v, k^*)$ ▷ Map index to category
  P.append(y_i)
  $\mathbf{r}_{i+1} \leftarrow \mathbf{r}_i - \mathbf{c}_{v,k^*}$ ▷ Compute next-level residual
  $v \leftarrow \text{Child}(v, k^*)$ ▷ Move to selected sub-node
  $i \leftarrow i + 1$
end while
return P

Construction. The Residual Tree is a multi-way unbalanced tree $T = (V, E, r)$ mirroring the label taxonomy, where $r = V_0$ is the root node. Each non-leaf node $v \in \mathcal{V}$ contains a trainable codebook $\mathbf{C}_v = \{\mathbf{c}_{v,1}, \mathbf{c}_{v,2}, \ldots, \mathbf{c}_{v,n_v}\}$, where n_v is the number of immediate sub-categories. Each code $\mathbf{c}_{v,i} \in \mathbb{R}^d$ represents a sub-category. If that sub-category is non-leaf, the code also serves as a pointer to the next-level node $u \in \text{Child}(v)$. Consequently, this construction defines a mapping $M : (v, k) \rightarrow y$ where $k \in \{1, \ldots, n_v\}$, associating each code index with a specific category in the taxonomy, ensuring that the selected code sequence indices decode into a valid hierarchical classification path.

The Workflow of Classification. The classification process, formalized in Algorithm 1, is a top-down traversal that recursively matches residuals against codebooks. The document embedding $\boldsymbol{x}$ is first projected into a latent representation $\boldsymbol{z} = \mathcal{E}(\boldsymbol{x})$ via an MLP encoder $\mathcal{E}$, with the initial residual $\boldsymbol{r}_0 = \boldsymbol{z}$. At each node v_i, the model selects the code minimizing $k^* = \arg\min_k \|\mathbf{r}_i - \mathbf{c}_{v_i,k}\|_2$ and update the residual as $\mathbf{r}_{i+1} = \mathbf{r}_i - \mathbf{c}_{v_i,k^*}$. This operation subtracts the coarse-grained semantic component, allowing deeper levels to focus on finer discriminative information. The process terminates upon reaching a leaf node. The selected code indices form a hierarchical label path $P = (y_0, y_1, \ldots, y_L)$ from root to leaf.

C. LLM-enhanced Prototype Generation

The LLM-enhanced prototype generation module derives high-quality semantic anchors for the residual tree through a pipeline. First, we prompt an LLM to expand each label into a discriminative description that highlights unique features among sibling categories. These descriptions and the document corpus are projected into a joint embedding space, and initial class-document mappings are established by retaining pairs exceeding a cosine similarity threshold. Second, to improve the mapping quality, the LLM annotates the mapped document and filters out the inconsistently assigned documents. Third, to

TABLE I: Experiment results on Amazon and WOS datasets. The "-L" suffix indicates the layer-wise prompting strategy.

Supervision-Type	Method	Amazon					WOS			
		Micro-F1	A@1	A@2	A@3	Macro-F1	Micro-F1	A@1	A@2	Macro-F1
Fully-Supervised		0.9287	0.9714	0.9348	0.8600	0.8180	0.9209	0.9476	0.8832	0.9005
Zero-shot	Qwen3-Max	-	0.9310	0.8350	0.6790	-	-	0.7780	0.6520	-
	Qwen3-Max-L	-	0.9290	0.7920	0.6730	-	-	0.6470	0.5530	-
	DeepSeek-V3	-	0.9300	0.8370	0.6650	-	-	0.7690	0.6500	-
	DeepSeek-V3-L	-	0.9260	0.7820	0.6550	-	-	0.6250	0.5290	-
	DeepSeek-V3.2	-	0.9150	0.8100	0.6570	-	-	0.7590	0.6480	-
	DeepSeek-V3.2-L	-	0.9190	0.7760	0.6460	-	-	0.6210	0.4950	-
Weakly-Supervised	Hier-doc2vec	0.2004	0.4357	0.1150	0.0504	0.0339	0.2333	0.3465	0.1202	0.1068
	WeSHClass	0.5229	0.8175	0.4435	0.3028	0.2058	0.5859	0.7047	0.4671	0.4946
	PlainTree	0.6371	0.9287	0.6472	0.4433	0.4023	0.6652	0.7533	0.5771	0.5957
	PlainTree-PV	0.8180	0.9293	0.8205	0.7041	0.6380	0.6934	0.7617	0.6251	0.6352
	TELEClass	0.8205	0.9323	0.8130	0.6563	**0.6435**	0.7726	0.8468	0.6632	0.7083
	RelaClass(ours)	**0.8375**	**0.9421**	**0.8571**	**0.7133**	0.6427	**0.7776**	**0.8493**	**0.7059**	**0.7236**

avoid insufficient training data for certain categories, we use the LLM to generate synthetic samples for each category, ensuring minimal training support. As an additional complement, we adopt a keyword-based rule that assigns a document to a category only if it contains category-specific keywords and no keywords from sibling categories. Finally, the representative documents for each category are encoded, and the prototype vector is calculated as the mean embedding. These prototypes serve as the anchors for the classification alignment.

D. Overall Training

Training Objective. The total loss $\mathcal{L}$ integrates reconstruction, residual quantization, and semantic alignment to supervise the learning of the latent space:

$$\mathcal{L} \coloneqq \mathcal{L}_{\text{recon}} + \beta\mathcal{L}_{\text{rq}} + \gamma\mathcal{L}_{\text{cls}}. \tag{1}$$

Specifically, $\mathcal{L}_{\text{recon}} = ||\boldsymbol{x} - \hat{\boldsymbol{x}}||^2$ ensures the quantized path retains original semantics, with $\hat{\mathbf{x}} = \mathcal{D}(\sum \boldsymbol{c}_{v,k^*})$ reconstructed by the MLP Decoder. The residual quantization loss $\mathcal{L}_{\text{rq}}$ follows the standard RQ-VAE formulation:

$$\mathcal{L}_{\text{rq}} \coloneqq \sum_{i=0}^{L} ||sg[\boldsymbol{r}_i] - \boldsymbol{c}_{v,k^i}||^2 + \alpha||\boldsymbol{r}_i - sg[\boldsymbol{c}_{v,k^i}]||^2, \tag{2}$$

where "sg" denotes the stop-gradient operator. To enforce taxonomic constraints, the classification alignment loss $\mathcal{L}_{\text{cls}}$ anchors each code $\boldsymbol{c}_{v,k}$ to its corresponding LLM-enhanced prototype $\boldsymbol{p}_{v,k}$:

$$\mathcal{L}_{\text{cls}} \coloneqq \sum_{v\in V}\sum_{k=1}^{n_v} ||\boldsymbol{c}_{v,k} - \boldsymbol{p}_{v,k}||^2. \tag{3}$$

Progressive Training Scheme. Training proceeds top-down, level by level. In Stage 1, all parameters (including the MLP encoder $\mathcal{E}$, codebooks, and the MLP decoder $\mathcal{D}$) are trained jointly. However, the classification alignment loss $\mathcal{L}_{\text{cls}}$ is restricted solely to the root-level codebook.

In Stage $t > 1$, we freeze the encoder $\mathcal{E}$ and all ancestor codebooks, optimizing only the level-t codebook, its descendants, and the decoder, with $\mathcal{L}_{\text{cls}}$ applied exclusively at level t. Freezing the encoder and ancestor codebooks preserves coarse-grained semantics, while the decoder remains trainable to reconstruct from progressively updated residuals. This ensures fine-grained residuals are learned only after coarse representations stabilize.

V. EXPERIMENTS

A. Experimental Setup

1) Dataset: We evaluate on two public HTC datasets, each dataset is randomly split 7:3 for training and testing. Amazon [21] contains 31,694 product descriptions under a three-level taxonomy with 5 root, 52 intermediate, and 265 leaf categories. Web Of Science (WOS) [2] contains 11,967 academic paper abstracts in a two-level hierarchy comprising 7 top-level and 33 leaf categories.

2) Baselines: We compare with the following weakly supervised and zero-shot HTC methods: Hier-doc2vec [22] embeds documents and class names into a shared space using Doc2Vec and performs top-down greedy decoding via cosine similarity. WeSHClass [14] generates pseudo samples from class-related keywords and iteratively refines the classifier through self-training. TELEClass [6] enriches taxonomies and annotates data using LLMs and in our experiments, we use DeepSeek-V3 as the backbone LLM for a fair comparison. To isolate the effect of the residual mechanism, we design two tree-based variants: PlainTree is a top-down cosine similarity classifier that uses LLM-generated descriptions as node embeddings, as raw category names perform poorly in preliminary trials. PlainTree-PV replaces PlainTree's embeddings with our category prototypes. We also evaluate Qwen3-Max and DeepSeek-V3 employing two prompting strategies: holistic prompting, which presents the full taxonomy in one prompt, and layer-wise prompting, which sequentially selects categories from the top level down. A fully supervised baseline (based on [6]), trained on fully annotated data is included as an upper bound.

3) Evaluation Metrics: We use Micro-F1, Macro-F1 [23], and Accuracy@k as evaluation metrics. Accuracy@k measures whether the predicted label path exactly matches the ground

truth up to the first k levels, reflecting the model's ability to understand hierarchical structures.

4) Implementation Details: We employ `SFR-Embedding-2_R` [24] as the text encoder and prototype encoder. The Residual Tree consists of an MLP encoder, hierarchical codebooks, and an MLP decoder. The encoder utilizes six hidden layers with ReLU activations, projecting features into a 32-dimensional latent space. Correspondingly, the decoder follows a symmetric architecture. Codebooks are initialized with LLM-generated prototypes to prevent codebook collapse [25]. The model is optimized with AdamW (learning rate 0.001, batch size 1024) following the progressive training strategy. Due to computational constraints, all evaluations based solely on LLMs are conducted on the same random subset of $1,000$ samples from the test set. Due to page limitations, other hyperparameter settings (e.g., α, β, γ) are provided in our open-source code. For the WOS dataset, several top-level classes exhibit substantial semantic overlap, causing coarse-level residuals to lack sufficient discriminative signal. Therefore we train directly on the leaf subclasses and map predictions back to primary categories for all the tree-based methods including ours.

B. Experimental Results

As shown in Table I, RelaClass outperforms all baselines across both datasets and most metrics. Our method improves Accuracy@3 by 5.7% on Amazon and Accuracy@2 by 4.27% on WOS compared to TELEClass. While RelaClass excels in Micro-F1 and Accuracy@k, the slightly lower Macro-F1 on Amazon suggests room for improvement in long-tail category handling. The significant improvement of PlainTree-PV over PlainTree (Micro-F1 increasing by 18.09%) validates the discriminative power of our prototypes. RelaClass further outperforms PlainTree-PV, with accuracy gains of 1.28%, 3.66%, and 0.92% on Amazon across the three hierarchical levels, confirming that the residual mechanism plays a key role in refining semantic representations across hierarchical levels. Among zero-shot baselines, we compare holistic prompting (presenting the full taxonomy in one prompt) and layer-wise prompting (classifying sequentially from the top level down). Both underperform RelaClass, with layer-wise prompting being worse due to its lack of global semantic context.

Figure 2 illustrates the effect of support sample size on hierarchical accuracy using Amazon dataset. To mitigate data scarcity in leaf categories, we augment the leaf-level prototypes with five LLM-synthesized samples per class. Level-1 accuracy stabilizes around 94% with only five samples, while Level-3 shows the most gradual performance gain, plateauing near 84% with 50 samples. This highlights the challenge of representing fine-grained categories, where LLM-based strategies provide essential support.

C. Ablation Studies

Table II summarizes the ablation results on the Amazon dataset. Removing any component degrades performance, confirming their synergistic contribution. LLM-based annotation is the most critical: its removal causes the sharpest decline, reducing Accuracy@3 by 10.56% and Macro-F1 by 8.43%, indicating that high-quality pseudo-labels are essential for semantically ambiguous categories. Description-enhanced semantic similarity matching provides a complementary signal: removing it lowers Accuracy@3 by 5.12%. While keyword-based supplementation yields modest but consistent performance gain.

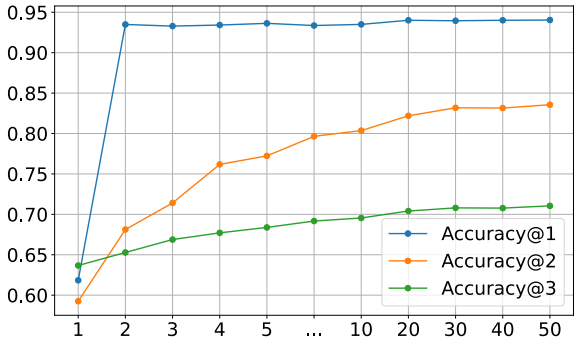


Fig. 2: Evaluation of accuracy with varying numbers of supporting samples for each prototype on Amazon dataset.

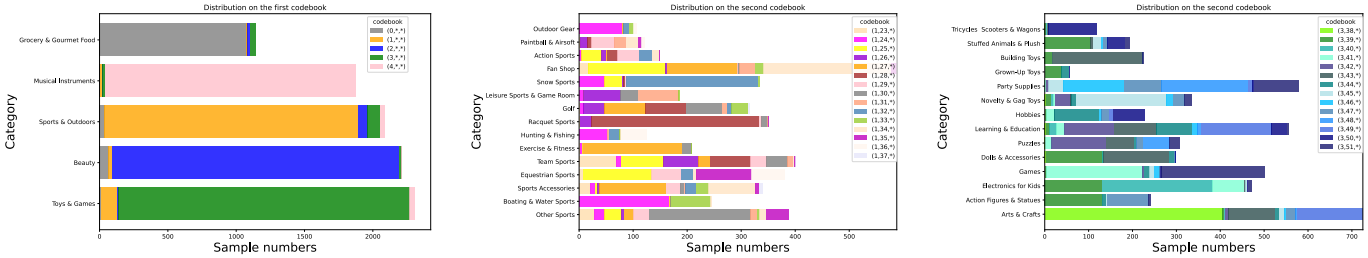
TABLE II: Ablation study of prototype generation strategies on the Amazon dataset. "Gen-Only" refers to using only synthesized samples.

Ablation module	Micro-F1	A@1	A@2	A@3	Macro-F1
Gen-Only	0.7458	0.9377	0.7663	0.5332	0.4746
w/o Description	0.8075	0.9421	0.8184	0.6621	0.5756
w/o Annotation	0.7911	0.9403	0.8254	0.6077	0.5584
w/o Keywords	0.8341	0.9421	0.8539	0.7062	0.6384
RelaClass	0.8375	0.9421	0.8571	0.7133	0.6427

Regarding the training objectives in Equation (1), removing either the reconstruction or quantization loss triggers codebook collapse. As shown in Figure 3, without the classification loss, performance deteriorates sharply at deeper levels, and category-to-code mapping becomes stochastic during training. This demonstrates that the classification loss is indispensable for anchoring the hierarchical taxonomy.

VI. Conclusion

In this work, we presented RelaClass, a novel residual tree-based framework for weakly supervised hierarchical text classification. By reformulating the taxonomy as a hierarchy of codebooks, RelaClass performs classification through a layer-wise residual matching mechanism, which progressively decomposes document embeddings to capture fine-grained semantic distinctions. Our framework integrates LLM-derived discriminative prototypes with a classification-aware training objective, providing an effective approach to align high-level semantic reasoning with rigid structural constraints. Evaluations demonstrate that RelaClass outperforms state-of-the-



(a) Ground-truth top-level category distribution for all the Amazon items colored by the first code index.

(b) The distribution for the second level category distributions for Sports & Outdoors items whose first code id should be 1.

(c) The distribution for the second level category distributions for Toys & Games items whose first code id should be 3.

Fig. 3: Qualitative study of top two level codebooks for classification while removing classification loss on the Amazon dataset. We show that the ground-truth categories are distributed across different codes.

art weakly supervised methods and zero-shot LLM baselines, especially when the taxonomy is well-defined and category semantics are unambiguous. Future work will examine how RelaClass performs in broader and more complex taxonomies.

VII. ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants No. 62441229, the Innovation Funding of ICT, CAS under Grant No. E561010.

REFERENCES

[1] B. He, S. Nag, L. Cui, S. Wang, Z. Li, R. Goutam, Z. Li, and H. Zhang, "Hierarchical query classification in e-commerce search," in *Companion Proceedings of the ACM Web Conference 2024*, 2024, pp. 338–345.

[2] K. Kowsari, D. E. Brown, M. Heidarysafa, K. J. Meimandi, M. S. Gerber, and L. E. Barnes, "Hdltex: Hierarchical deep learning for text classification," in *2017 16th IEEE international conference on machine learning and applications (ICMLA)*. IEEE, 2017, pp. 364–371.

[3] J. Zhou, C. Ma, D. Long, G. Xu, N. Ding, H. Zhang, P. Xie, and G. Liu, "Hierarchy-aware global model for hierarchical text classification," in *Proceedings of the 58th annual meeting of the association for computational linguistics*, 2020, pp. 1106–1117.

[4] H. Zhu, C. Zhang, J. Huang, J. Wu, and K. Xu, "Hitin: Hierarchy-aware tree isomorphism network for hierarchical text classification," *arXiv preprint arXiv:2305.15182*, 2023.

[5] Q. Zhang, Q. Su, W. Zhu, and Y. Pang, "Hierprompt: Zero-shot hierarchical text classification with llm-enhanced prototypes," in *Findings of the Association for Computational Linguistics: EMNLP 2025*, 2025, pp. 3846–3859.

[6] Y. Zhang, R. Yang, X. Xu, R. Li, J. Xiao, J. Shen, and J. Han, "Teleclass: Taxonomy enrichment and llm-enhanced hierarchical text classification with minimal supervision," in *Proceedings of the ACM on Web Conference 2025*, 2025, pp. 2032–2042.

[7] D. Lee, C. Kim, S. Kim, M. Cho, and W.-S. Han, "Autoregressive image generation using residual quantization," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11 523–11 532.

[8] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.

[9] S. Banerjee, C. Akkaya, F. Perez-Sorrosal, and K. Tsioutsiouliklis, "Hierarchical transfer learning for multi-label text classification," in *Proceedings of the 57th annual meeting of the association for computational linguistics*, 2019, pp. 6295–6300.

[10] J. Wehrmann, R. Cerri, and R. Barros, "Hierarchical multi-label classification networks," in *International conference on machine learning*. PMLR, 2018, pp. 5075–5084.

[11] R. You, Z. Zhang, Z. Wang, S. Dai, H. Mamitsuka, and S. Zhu, "Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification," *Advances in neural information processing systems*, vol. 32, 2019.

[12] Y. Mao, J. Tian, J. Han, and X. Ren, "Hierarchical text classification with reinforced label assignment," *arXiv preprint arXiv:1908.10419*, 2019.

[13] L. Zhang, J. Zhou, R. Fan, N. Xiong, L. Zhang, and J. Wan, "Sequences and nodes: Probability-guided contrastive learning for hierarchical text classification," *Knowledge-Based Systems*, p. 114224, 2025.

[14] Y. Meng, J. Shen, C. Zhang, and J. Han, "Weakly-supervised hierarchical text classification," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 6826–6833.

[15] J. Shen, W. Qiu, Y. Meng, J. Shang, X. Ren, and J. Han, "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*, 2021, pp. 4239–4249.

[16] F. Schmidt, K. Hammerfald, H. H. Jahren, A. H. Payberah, and V. Vlassov, "Single-pass hierarchical text classification with large language models," in *2024 IEEE International Conference on Big Data (BigData)*. IEEE, 2024, pp. 5412–5421.

[17] Y. Zhang, M. Wang, Q. Li, P. Tiwari, and J. Qin, "Pushing the limit of llm capacity for text classification," in *Companion Proceedings of the ACM on Web Conference 2025*, 2025, pp. 1524–1528.

[18] S. Rajput, N. Mehta, A. Singh, R. Hulikal Keshavan, T. Vu, L. Heldt, L. Hong, Y. Tay, V. Tran, J. Samost *et al.*, "Recommender systems with generative retrieval," *Advances in Neural Information Processing Systems*, vol. 36, pp. 10 299–10 315, 2023.

[19] B. Zheng, Y. Hou, H. Lu, Y. Chen, W. X. Zhao, M. Chen, and J.-R. Wen, "Adapting large language models by integrating collaborative semantics for recommendation," in *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 2024, pp. 1435–1448.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.

[21] J. McAuley and J. Leskovec, "Hidden factors and hidden topics: understanding rating dimensions with review text," in *Proceedings of the 7th ACM conference on Recommender systems*, 2013, pp. 165–172.

[22] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*. PMLR, 2014, pp. 1188–1196.

[23] S. Gopal and Y. Yang, "Recursive regularization for large-scale classification with hierarchical and graphical dependencies," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, pp. 257–265.

[24] R. Meng, Y. Liu, S. R. Joty, C. Xiong, Y. Zhou, and S. Yavuz, "Sfrembedding-2: Advanced text embedding with multistage training," 2024. [Online]. Available: https://huggingface.co/Salesforce/SFR-Embedding-2_R

[25] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2021.