KG-HTC: Integrating Knowledge Graphs into LLMs for Zero-shot Hierarchical Text Classification

Qianbo Zang [a,*], **Igor Tchappi**[a], **Christophe Zgrzendek**[b], **Afshin Khadangi**[a] **and Johannes Sedlmeir**[c]

[a]Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg
[b]Enovos Luxembourg S.A.
[c]University of Münster

Abstract. Hierarchical Text Classification (HTC) involves assigning documents to labels organized within a taxonomy. Most previous research on HTC has focused on supervised methods. However, in real-world scenarios, employing supervised HTC can be challenging due to a lack of annotated data. Moreover, HTC often faces issues with large label spaces and long-tail distributions. In this work, we present **K**nowledge **G**raphs for zero-shot **H**ierarchical **T**ext **C**lassification (KG-HTC), which aims to address these challenges of HTC in applications by integrating knowledge graphs with Large Language Models (LLM) to provide structured semantic context during classification. Our method retrieves relevant subgraphs from knowledge graphs related to the input text using a Retrieval-Augmented Generation (RAG) approach, thereby augmenting the model's understanding of label semantics at various hierarchy levels. We evaluate KG-HTC on three open-source HTC datasets: WoS, DBpedia, and Amazon. Our experimental results show that KG-HTC significantly outperforms three baselines in the strict zero-shot setting, particularly achieving substantial improvements at deeper levels of the hierarchy. This evaluation demonstrates the effectiveness of incorporating structured knowledge into LLMs to address HTC's challenges in large label spaces and long-tailed label distributions. Our code is available at: https://github.com/QianboZang/KG-HTC.

1 Introduction

Text classification is a fundamental task in natural language processing that focuses on assigning one or more predefined categories to a given piece of text. A specific and increasingly important extension of text classification is Hierarchical Text Classification (HTC), where the labels are not simply flat but are organized within a multi-level taxonomy. HTC aims to categorize textual data into multi-level label systems with parent-child relationships, where labels at different levels are organized as a hierarchical taxonomy. Figure 1 features an example of HTC from the Amazon Product Review dataset. [1] In this figure, the root of the hierarchy is the name of the dataset, and a review needs to be classified to labels at multiple levels. The example input text can be sequentially classified as Health Personal Care, Household Supplier, and Dishwashing. HTC has been successfully applied in different domains, including e-commerce product categorization [4, 26], hierarchical topic modeling of e-participation platforms [29], fine-grained literature management [6, 16], to name a few.

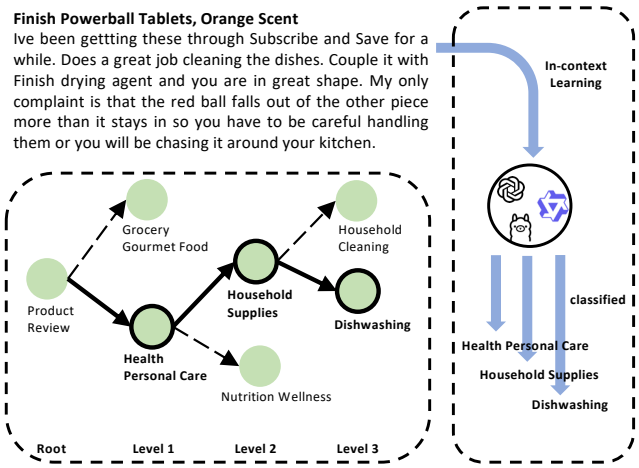

Figure 1. An example of HTC from the Amazon Product Review dataset.

However, in real-world applications, HTC can face one or multiple out of the following three significant challenges. First, there may be a shortage of annotated data, particularly as the cost of manually labeling custom data at multiple hierarchical levels is prohibitively high [5]. This problem becomes even more severe in dynamic environments such as retail systems, where taxonomies evolve with new product lines. Second, practical taxonomies often exhibit large-scale label spaces [34]. For instance, the hierarchy of the Amazon Product Review dataset contains over 500 leaf categories. Third, real-world datasets in HTC can exhibit highly imbalanced long-tail distributions, i.e., a small number of frequent categories dominates the dataset while most classes remain underrepresented. For instance, 15 % of categories account for 80 % of total instances in the third level of the Amazon Product Review dataset. In contrast, the bottom 50 % of categories collectively represent merely 6 % of the data volume.

For these three reasons, and contrary to flat text classification, standard supervised approaches are not well-suited to HTC in many industry settings [11, 13, 16, 24]. As a consequence, researchers have increasingly turned to zero-shot learning methods to approach HTC [2, 11, 25]. We can group recent zero-shot learning based studies into three main approaches. The first approach leverages the in-context learning of pre-trained Large Language Models (LLMs)

* Corresponding Author. Email: qianbo.zang@uni.lu.
[1] www.kaggle.com/datasets/kashnitsky/hierarchical-text-classification

through carefully designed prompts to accomplish accurate classification. Halder et al. [11] transforms any text classification task into a universal binary classification problem, where deep learning models determine a binary (`True` or `False`) prediction for a given text and a potential class label. However, this method requires multiple classifications for each textual data, as each possible label needs to be evaluated individually. The second approach utilizes embedding models converted from pre-trained LLMs to calculate the distance between input text and labels for classification. Secondly, Bongiovanni et al. [2] proposed Z-STC to propagate similarity scores up the hierarchy and leverage this propagated information to improve classification quality for upper-level labels. The third method combines embedding models with LLM-based classification. Thirdly, Paletto et al. [25] introduced HiLA, where LLMs generate new label layers inserted into the bottom of the current taxonomy. Then, Paletto follows Z-STC for classification. However, Bongiovanni et al. [2]'s and Paletto et al. [25]'s methods exhibit low classification performance for deeper levels.

In this paper, we propose KG-HTC, a new method to enhance HTC by integrating Knowledge Graphs (KGs) into LLMs for HTC. To this end, we will focus on the **strict zero-shot learning** setting. Since Directed Acyclic Graphs (DAGs) can represent the hierarchical structure of labels in HTC, the combination of LLMs and knowledge graphs seems particularly promising for this purpose. We represent the taxonomy as a DAGs-based knowledge graph and compute the cosine similarity between the text and the embeddings of labels at each level. By applying preset thresholds, candidate labels that are highly semantically relevant to the input text are chosen at every hierarchical level. Then, leveraging these candidate labels, the system dynamically retrieves the most pertinent subgraph from the complete label knowledge graph corresponding to the given text. For the retrieved subgraph, an upwards propagation algorithm is employed to systematically enumerate all possible hierarchical paths from the leaf nodes to the root, with each path representing a complete reversed hierarchical label sequence. These structured sequences are subsequently concatenated into a prompt, which is fed into an LLM to perform the zero-shot classification task.

We evaluate our approach using three public datasets and achieve new state-of-the-art results for all of them. Without relying on any annotated data, the KG-HTC method significantly enhances the model's capability to overcome the issues of long-tail distributions and sparse labels. We find that compared to the approaches of Bongiovanni et al. [2] and Paletto et al. [25], our KG-HTC exhibits significantly smaller performance degradation as the label hierarchy deepens. As such, the contributions of this paper are threefold:

1. We present a novel approach – KG-HTC – to integrate KGs into LLMs for HTC under a strict zero-shot setting.
2. We implement a novel pipeline that semantically retrieves relevant subgraphs from the label taxonomy based on cosine similarity with the input text and transforms it to a structured prompt.
3. We conduct an in-depth assessment of KG-HTC, comparing it with the previous state-of-the-art on three main public HTC datasets.

The remainder of this paper is structured as follows: First, we survey related work in Section 2. Subsequently, we introduce KG-HTC in Section 3. Section 4 describes our experimental settings, followed by an analysis of our experimental results in Section 5. Finally, we conclude our research in Section 6.

2 Related Work

Hierarchical Text Classification. The task of HTC was initially proposed by Sun and Lim [30], who suggested using Support Vector Machine classifiers as a solution. Subsequently, Kowsari et al. [16] explored various deep learning methods, training different deep neural networks for each level of the taxonomy. More recently, Liu et al. [22] introduced the integration of knowledge graphs to enhance HTC. They employed Graph Neural Networks (GraphSAGE) to encode knowledge graphs and combined graph embeddings with BERT text embeddings for fine-tuning. This approach currently represents the state-of-the-art in supervised settings. However, supervised methods face challenges in industrial applications. The primary issue is the lack of annotated data, as the cost of labeling data across multiple hierarchical levels can be prohibitively high. As LLMs continue to improve [31, 20, 28, 32], zero-shot inference has become increasingly popular in many fields [3, 12, 19, 35].

Zero-shot HTC. Three recent studies align with our strict zero-shot learning setting of HTC. First, Halder et al. [11] proposed the Task-Aware Representation method for zero-shot text classification. This approach allows any text classification task to be converted into a universal binary classification problem. Given a text and a set of labels, the LLMs determines whether (`True`) or not (`False`) a label matches. However, due to HTC's extensive label space, this method requires multiple iterations to complete a single classification.

Second, Bongiovanni et al. [2] proposed Zero-shot Semantic Text Classification (Z-STC), which utilized the Upward Score Propagation (USP) method for HTC. This approach leverages pre-trained language models, such as BERT [7] and RoBERTa [21], to independently encode documents and taxonomy labels into a semantic vector space. The prior relevance score of each label is then computed via cosine similarity. By incorporating the hierarchical structure of the taxonomy, USP recursively propagates the relevance scores of lower-level labels upward to their parent nodes. The key idea is that if a child label is relevant to the content to be classified, its parent label should also be considered relevant. This hierarchical score propagation effectively integrates local semantic cues into the global taxonomy, improving overall classification performance.

Finally, Paletto et al. [25] proposed the Hierarchical Label Augmentation (HiLA) method for HTC. They used pre-trained LLMs to generate additional child labels for the leaf nodes of the existing label hierarchy. Since satisfying token constraints by directly inputting the whole hierarchy into an LLM is challenging, they adopted an iterative approach to generate extended sub-labels for each branch to avoid redundancies and label overlap. Moreover, Paletto et al. [25] applied Bongiovanni's Z-STC to the new deeper levels to accomplish the HTC task. As such, HiLA represents the state-of-the-art in the strict zero-shot setting. However, as we demonstrate in Section 5, Bongiovanni's [2] and Paletto's [25]'s methods exhibit low classification accuracy for deeper-level labels. As the taxonomy deepens, the label space for the corresponding single-layer classification task becomes larger, and the distributions of labels become more biased.

Retrieval Augment Generation. Retrieval-Augmented Generation (RAG) can dynamically retrieve relevant information from an external corpus or database during in-context learning inference [15, 17]. Graph RAG extends RAG to retrieve interconnected entities and relationships through KGs, enabling richer contextual understanding [8]. Both RAG and graph RAG have demonstrated significant improvements in open-domain question answering [9, 27]. However, there is limited research on leveraging RAG and graph RAG for text classification.

Table 1. List of symbols.

Symbol	Explanation
l	A hierarchy level of a label taxonomy, $l \in \{1, \ldots, L\}$
C^l	Set of all labels at level l
c_i^l	A label at level l
$\uparrow c_i^l$	The parent label of c_i^l
$\downarrow c_i^l$	The set of child labels of c_i^l
$\Psi(x)$	A vector embedding of a text x
$S_C(\Psi_1,\ \Psi_2)$	The cosine similarity of two embeddings Ψ_1 and Ψ_2
Q^l	All queried labels by S_C with a threshold τ_l at level l

3 Method

3.1 Problem Identification

Zero-shot text classification via LLMs is formally framed as a generative classification task. Given a text sequence $x = (x_1, x_2, \ldots, x_n)$ as input, where n denotes text length, an LLM generates an output text $y \sim \mathcal{LLM}(x)$ using a Top-p sampling strategy. In classification contexts, the set of labels can be defined as C. The generated text y can then be mapped to a predicted label $\hat{c} \in C$.

Our mathematical framework follows the formulation established by Paletto et al. [25], with key symbols summarized in Table 1. In hierarchical text classification, all labels C in the label space are organized into a hierarchical taxonomy $C = (C^1, C^2, \ldots, C^L)$, where L is the maximum depth of the taxonomy, i.e., the number of levels. To formally capture the dependencies of the labels, we adopt the upward and downward arrow notation employed by [25], where $\uparrow c_i^l$ represents the parent node of c_i^l in the hierarchy and $\downarrow c_i^l$ represents the set of child nodes of c_i^l in the hierarchy. For all $l \in \{2, \ldots, L\}$, each label $c_i^l \in C^l$ must satisfy the structural constraint $\uparrow c_i^l \in C^{l-1}$. Likewise, for all $l \in \{1, \ldots, L-1\}, \downarrow c_i^l \subset C^{l+1}$. The task of hierarchical text classification requires an LLM to iteratively generate outputs $(y^1, y^2, \ldots, y^L)$ corresponding to predicted labels $(\hat{c}^1, \hat{c}^2, \ldots, \hat{c}^L)$ within a hierarchical taxonomy, i.e., $\hat{c}^l \in C^l\ \forall l \in \{1, \ldots, L\}$.

3.2 System architecture

Figure 2 illustrates the full process of our KG-HTC. First, we store all labels in a graph database and a vector database, respectively. Given an input text, we subsequently retrieve labels of top candidates Q^l at each level l from the vector database and retrieve a valid subgraph from the graph database by checking parent-child relationships between candidates from adjacent levels. Then, we convert a set of paths in a retrieved subgraph into a structured prompt and concatenate the structured context with a classification prompt. Finally, we leverage In-context Learning for zero-shot text classification.

3.3 Storage of hierarchical labels

In the first step, we store all labels into a vector database and graph database, respectively. The RAG component relies on a vector database to quickly locate nodes at each level. The graph database enables structural reasoning to generate paths that adhere to hierarchical constraints. In hierarchical text classification, the taxonomies of labels can be conceptualized as DAG knowledge graphs, where multi-level labels are interconnected through hierarchical affiliation relationships. By explicitly representing the relational pathways between labels in each tier, a LLM can develop a structural understanding of individual labels and their conceptual boundaries within the hierarchy. This graph-based knowledge representation equips LLMs with dual advantages: it not only establishes explicit semantic navigation pathways for text processing tasks, but also creates topological constraints that may substantially enhance classification accuracy in zero-shot scenarios through improved semantic disambiguation [22].

Algorithm 1: Subgraph Retrieval

Input: Input text x, labels $\{C^1, C^2, \ldots, C^L\}$
Output: Retrieved sub-graph G

Initialize an empty graph structure $G \leftarrow \emptyset$;
Compute the embedding $\Psi(x)$ for the input text x;
for $l \leftarrow 1$ *to* L **do**
  Compute $\Psi(c_i^l)$ for all labels at level l;
  Calculate $d_C(\Psi(x), \Psi(c_i^l))$ via Equation 1;
  Retrieve candidates $Q^l \subset C^l$ via Equation 2;
end
for $l \leftarrow 2$ *to* L **do**
  for *each* $c_i^l \in Q^l$ **do**
    if $\uparrow c_i^l \in Q^{l-1}$ **then**
      Add nodes $\uparrow c_i^l$ and c_i^l to G;
      Add edge $(\uparrow c_i^l,\ c_i^l)$ to G;
    end
  end
end
Delete repeated $(\uparrow c_i^l,\ c_i^l)$ in G ;
return G;

3.4 Subgraph Retrieval

Empirical studies have demonstrated that the RAG framework exhibits significant advantages in open question-answering tasks for LLMs [10, 14, 17]. A defining challenge in HTC stems from the scenarios of large label spaces, where the label taxonomy spans many categories. By dynamically retrieving relevant documents from vector databases through a similarity check with in-context learning, RAG effectively enhances the factual accuracy of generated responses.

The research of Li et al. [18] shows that LLMs exhibit limitations in both long-context processing and classification tasks of large label spaces. Directly encoding the full knowledge graph into LLMs may therefore suffer from performance degradation due to information overload or attention dilution. To mitigate this issue, we propose an RAG enhanced framework that dynamically retrieves semantically relevant subgraph (sub-tree) components from the whole knowledge graph based on the input text. These retrieved subgraphs are subsequently structured as contextualized prompts, enabling the classifier to prioritize critical hierarchical dependencies while suppressing (heuristically) irrelevant noise. We will thoroughly introduce this approach in Section 3.5.

Specifically, for each hierarchical level l, we compute the cosine similarity distance between the embedding of the input text and all labels at level l:

$$d_C(x,\ c_i^l) = 1 - S_C\left(\Psi(x),\ \Psi(c_i^l)\right), \tag{1}$$

where $\Psi : X \to \mathbb{R}^d$ denotes the embedding function fine-tuned from the pre-trained LLMs, and $S_C(\cdot,\ \cdot)$ represents the cosine similarity operator. We then retrieve labels of top candidates at each level l:

$$Q^l = \{c_i^l \,|\, d_C(x,\ c_i^l) \leq \tau_l\}, \tag{2}$$

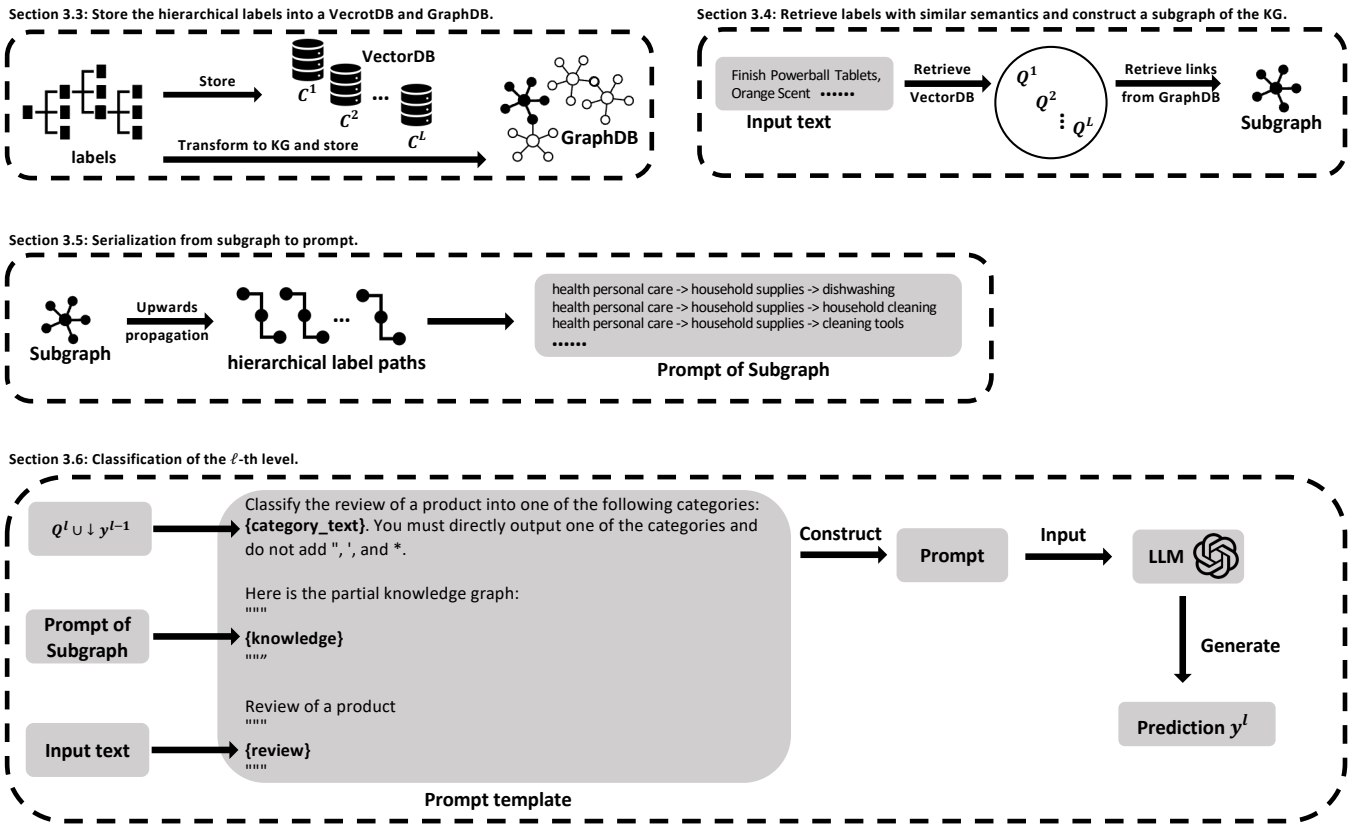

Figure 2. The system architecture of KG-HTC.

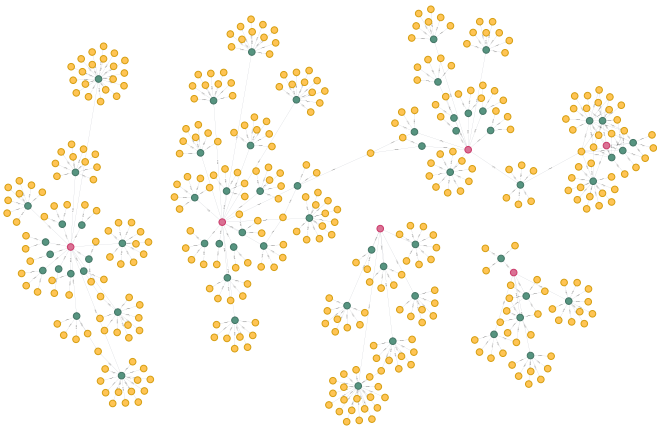

Figure 3. Visualization of the knowledge graph (tree) constructed from the multi-level taxonomy in the Amazon Product Review dataset. The red nodes represent labels in the first hierarchical level. The green nodes denote sub-categories (second level) interconnected through parent-child relationships. The yellow nodes correspond to the fine-grained leaf categories in the third (deepest) level.

where τ_l is a similarity threshold and represents the maximum cosine distance allowed for a label c_i^l to be included in the set of candidate labels Q^l for level l.

To ensure hierarchical consistency, we validate cross-level dependencies by checking parent-child relationships between candidates from adjacent levels. A candidate label $c_i^l \in Q^l$ must satisfy the condition that its parent label $\uparrow c_i^l$ belongs to Q^{l-1}. Algorithm 1 shows the complete process for retrieving a subgraph based on input text x as the query.

Algorithm 2: From Subgraph to Hierarchical Label Paths

Input: Input text x, subgraph G retrieved via Algorithm 1
Output: Set of hierarchical paths $\{P_1, P_2, ...\}$

Initialize path set $P \leftarrow \emptyset$;
for *each leaf node* $c_i^L \in C^L$ **do**
  Initialize a stack structure $S \leftarrow \{[c_i^L]\}$;
  for $l \leftarrow L$ *to* 2 **do**
    Push the parent node $\uparrow c_i^l \in C^{l-1}$ of c_i^L to S;
  end
  Add S to P;
end
return P;

3.5 *Serialization from Subgraph to Prompt*

For an input text x, we first retrieve a subgraph G through Algorithm 1. To effectively inject the KG into the LLM, we adopt a strategy that converts one graph structure into a set of paths through parent-child edge connections [33]. Specifically, each subgraph is serialized as a set of hierarchical paths from root to leaf nodes. This chain-based representation ensures structural consistency while maintaining compatibility with sequential input formats of LLMs. For HTC tasks, the length of paths equals the taxonomy depth L.

LLMs cannot directly understand graph information. We need to transfer the subgraph to a prompt. A feasible method is to input the paths inside graphs as a prompt for graph understanding and reasoning [23]. Valid hierarchical label paths in the subgraph are preserved as contextual prompts with the upwards propagation. The upwards propagation is implemented as a loop traversal algorithm

Algorithm 3: KG-HTC

Input: Input text x, prompt template P
Output: Predicted hierarchical labels $\mathbf{y} = (y^1, y^2, ..., y^L)$

Initialize the set of predicted labels $y \leftarrow \emptyset$;
Retrieve subgraph G via Algorithm 1;
Transform G to hierarchical label paths P;
for *level* $l \leftarrow 1$ *to* L **do**
  if $l == 1$ **then**
    $prompt \leftarrow concat(x,\ P,\ C^1)$;
  else
    Retrieve candidates Q^l via Equation 2;
    $prompt \leftarrow concat(x,\ P,\ Q^l \cup \downarrow y^{l-1})$;
  end
  $y^l \leftarrow \mathcal{LLM}(Prompt)$;
  Add y^l to y ;
end
return $\mathbf{y} = (y^1, ..., y^L)$

that systematically explores all possible paths from the terminal leaf nodes to the root node in a hierarchical structure. Starting from any leaf node $c_i^L \in C^L$ in the deepest hierarchical level, Algorithm 2 can progressively traverse parent nodes until reaching root nodes, and then backtrack to explore alternative branches. This exhaustive traversal guarantees the complete enumeration of all valid path combinations. After obtaining all path combinations, the sequence of elements in each path P_i will be reversed to ensure the directional consistency from the first layer to the L^{th} layer, with the final output being the complete set of paths $P = \{P_1,\ P_2,\ ...\ \}$. Each path $P_i = (p_i^1 \rightarrow p_i^2 \rightarrow ... \rightarrow p_i^L)$ represents a connected node sequence, where $p_i^l \in G$ and $p_i^l \in C^l$.

We can convert a set of paths in a retrieved subgraph into a structured prompt using $\rightarrow$ to connect two nodes in adjacent levels. This structured context enables the LLM to recognize hierarchical constraints during classification. An illustrative prompt example of such multi-level label paths for the Amazon Product Review dataset is provided in the third block of Figure 2.

3.6 Classification of Each Level

Our approach leverages In-context Learning and prompt engineering by concatenating the structured context extracted from the retrieved subgraph with a classification prompt template. Subsequently, the concatenated prompt is fed into the LLM for inference. The final block of Figure 2 features an example of the classification template from the Amazon Product Review dataset.

For HTC, we follow a layer-wise classification strategy. The model begins by predicting the label at the first level, then proceeds to classify labels at deeper levels sequentially, until the deepest level is reached. One major challenge in HTC is the large number of candidate labels. Providing all possible labels in a single prompt often leads to performance degradation [18]. However, since the number of first-level labels is usually small (no more than 10 among our evaluation datasets), we can directly include all first-level labels into the prompt. This strategy simplifies inference and heuristically enhances classification performance.

For classification beyond the first level, the model uses the prediction from the previous level as a constraint for the current level. For instance, the predicted label at level l is y^l. The candidate label set at the next level $l+1$ mainly consists of the subcategories $\downarrow y^l$ of the previous prediction y^l, along with additional labels Q^{l+1} retrieved using Equation 2. This equation helps mitigate the impact of potential errors in the previous level's prediction, as the confidence in all labels in $\downarrow y_l$ that are consistent with previous labeling is continuously benchmarked against any close labels in the corresponding layer C_{l+1}. This approach promises to improve the overall robustness and accuracy of the classification process. Algorithm 3 illustrates the full process of our KG-HTC.

4 Experiment

4.1 Dataset

Amazon Product Reviews (Amazon). [2] Each data item has a title and description and is classified according to a three-level hierarchical taxonomy with 6, 64, and 510 label categories, respectively.

Web of Science (WoS) [16]. [3] This dataset is about scientific literature. It includes data from multiple fields, such as natural sciences, social sciences, and humanities and arts, and is widely used in academic research, bibliometric analyses, and scientific evaluations. Each item is classified according to a two-level hierarchical taxonomy with 7 and 134 label categories, respectively.

Dbpedia [1]. [4] This dataset is an open knowledge base project built on Wikipedia. Scientists extract and transform the vast wealth of information from Wikipedia and present it in a structured and standardized format. The data within the Dbpedia forms a massive and complex knowledge graph that supports users in cross-domain knowledge exploration. Each data item is classified according to a three-level hierarchical taxonomy, encompassing 9, 70, and 219 label categories at each level, respectively.

4.2 Metric

We employ the F1-macro score as the evaluation metric. The F1-macro score ensures that rare classes contribute equally to the final metric. This prevents overoptimistic performance estimates in imbalanced datasets, where some classes are underrepresented. Formally, it is defined as:

$$\text{F}_1\text{-macro} = \frac{1}{C}\sum_{i=1}^{C}\text{F}_{1i}, \tag{3}$$

where C denotes the total number of classes, and the F1-score for the i-th class is the harmonic mean of precision and recall:

$$\text{F}_{1i} = 2 \times \frac{\text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}. \tag{4}$$

To evaluate the compounded challenges of large label spaces and long-tailed distributions, we implement a metric called average decay rate via Equation 6:

$$\text{decay}_i = \frac{(\text{F}_1\text{-macro of level}_{i-1}) - (\text{F}_1\text{-macro of level}_i)}{(\text{F}_1\text{-macro of level}_{i-1})} \tag{5}$$

$$\text{decay}_{\text{avg}} = \frac{1}{L-1}\sum_{i=2}^{L}\text{decay}_i \tag{6}$$

As label spaces tend to become larger at deeper labels, it is not surprising that the F1-score tends to degrade.

[3] huggingface.co/datasets/HDLTex/web_of_science
[4] www.kaggle.com/datasets/danofer/DBpedia-classes

Table 2. Main results. The evaluation metric is F1-macro. Our KG-HTC provides consistent and significant improvements over both the weak baseline and the strong baselines.

Model		WoS		Dbpedia			Amazon		
		Level 1	Level 2	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
Weak baselines	GPT-3.5-turbo	0.642	0.358	0.616	0.645	0.672	0.754	0.178	0.129
Strong baselines	Z-STC [2]	0.741	0.462	0.759	0.656	0.628	0.712	0.348	0.173
	HiLA [25]	0.647	0.371	0.768	0.660	0.629	0.762	0.393	0.249
Ours	**KG-HTC**	**0.756**	**0.630**	**0.883**	**0.796**	**0.894**	**0.908**	**0.654**	**0.445**

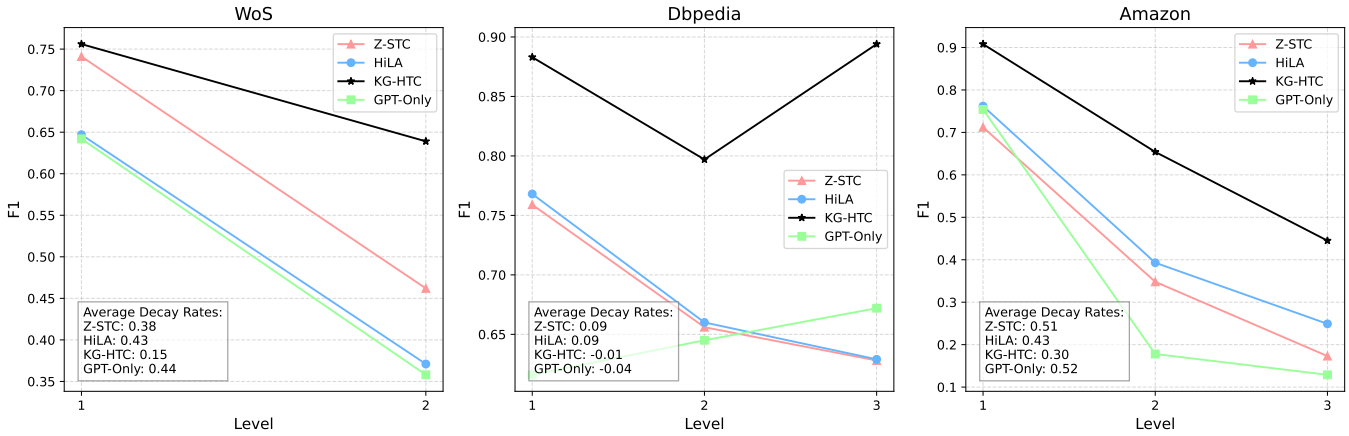

Figure 4. As the taxonomy deepens, KG-HTC exhibits a slower performance degradation on the WoS and Amazon datasets.

4.3 Experiment setup

Baselines. We evaluate our method by comparing the results to two baselines. The first baseline is a weak baseline, where LLMs will directly classify each data point per each layer of labels. Secondly, we use two LLM-based HTC approaches introduced by previous studies, Z-STC and HiLA [2, 25], as strong baselines (see also Section 2). To the best of our knowledge, these two studies mark the previous state-of-the-art in the strict zero-shot setting.

Experiment details. We employ the `GPT-3.5-turbo` [5] LLM to align with the previous evaluations as our strong baselines [25]. We chose `text-embedding-ada-002` [6] as the embedding model for the RAG system and `neo4j` [7] as the graph database in our experiments. Furthermore, we deployed `ChromaDB` [8] as the vector database.

In our experiments among all datasets, we set the temperature of the `GPT-3.5-turbo` model to 0.4 and the Top-p parameter to 0.4. We use a moderate temperature and a low Top-p value to obtain a relatively stable response of `GPT-3.5-turbo`. For the RAG system, we selected similarity threshold values τ_l such that there are 10 label candidates retrieved at the second level and 40 at the third level for both Dbpedia and Amazon datasets. For the WoS dataset, we selected similarity threshold values τ_l such that there are 20 label candidates retrieved at the second level. Note that due to the inherent randomness (non-determinism) in the generation of LLMs, the generated output may occasionally fall outside the predefined label space. In these cases, we convert the invalid output to a valid label by randomly sampling from the label space, using a fixed random seed of 42 for reproducibility.

[5] platform.openai.com/docs/models/gpt-3.5-turbo
[6] platform.openai.com/docs/guides/embeddings
[7] neo4j.com
[8] trychroma.com

5 Results

5.1 Main Results

The experimental results in Table 2 demonstrate that our KG-HTC provides consistent and significant improvements over both the weak baseline and the strong baselines. Compared to using `GPT-3.5-turbo` alone for zero-shot classification (our weak baseline), KG-HTC demonstrates remarkable performance improvements. Our experimental results show that the average performance improvement is 27.1 % for the first-level classification, while the second and third levels achieve enhancements of 123.1 % and 139.0 %, respectively. These results validate that integrating knowledge graphs into LLMs can significantly enhance the performance of HTC. As the classification level increases, KG-HTC demonstrates progressively larger improvements, particularly in handling high-level abstract information. This indicates that KG-HTC can effectively address the challenges associated with large label space and long-tailed distributions in hierarchical classification.

As we illustrate in Figure 4, our KG-HTC indicates the lowest performance degradation on the WoS and Amazon datasets as the taxonomy deepens, and the performance gap among the whole three baselines significantly widens. These findings further demonstrate that our approach effectively addresses the challenges posed by large label spaces and long-tailed distributions in hierarchical classification.

Table 3. Error Analysis. The Hit@K of the RAG system decreases at the second and third levels across all misclassified samples.

Dataset	WoS	Dbpedia		Amazon	
	Level 2	Level 2	Level 3	Level 2	Level 3
Hit@K	0.731	0.760	0.649	0.498	0.392

Table 4. Ablation study on the effectiveness of the subgraph retrieval. Full-KG replaces the RAG-based subgraph retrieval by the retrieval of the entire knowledge graph.

Dataset	WoS		Dbpedia			Amazon		
	Level 1	Level 2	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
Full-KG	**0.762**	0.616	**0.913**	0.670	0.884	**0.926**	0.633	0.431
KG-HTC	0.749	**0.651**	0.886	**0.811**	**0.902**	0.901	**0.651**	**0.462**

5.2 Error Analysis

The Hit@K metric specifies the fraction of queries for which the correct label appears in the top K results returned by the retriever. We described the details of our RAG system in Section 4. In line with our choice of τ_l, for Dbpedia and Amazon, we use Hit@10 for the classification of the second level and Hit@40 at the third level. We further use Hit@20 for the classification of the second level of WoS. From the results in Table 3, we observe that the Hit@K of the RAG system decreases at the second and third levels across all misclassified samples. This decline suggests that one key reason for performance degradation is the RAG system's inability to retrieve the correct subgraph during inference. When the retrieved knowledge is not closely aligned with the input text or the classification task, the model lacks sufficient contextual support for accurate reasoning. As a result, it becomes more likely that incorrect labels are chosen. Inaccurate or irrelevant subgraphs may also introduce noise or confusion, further reducing the model's ability to assign the correct labels. This observation also implies that the performance of KG-HTC can continue to improve as information retrieval techniques evolve, especially in terms of retrieval precision and relevance.

5.3 Ablation study

We sampled a subset with 5000 data samples for each evaluation dataset with a random seed of 42 in the following experiments.

Effectiveness of Subgraph Retrieval. Our empirical results indicate that involving knowledge graphs via RAG can improve the performance of LLMs in HTC. In this experiment, we will evaluate the component responsible for retrieving a subgraph. For a comparative assessment, we remove the RAG system that previously was responsible for fetching a subgraph from the entire knowledge graph while keeping the other parts of the system unchanged. Instead, we hand the full knowledge graph to the LLM. This adjusted setup is referred to as Full-KG.

The corresponding performance results are presented in Table 4. Our KG-HTC has substantially better results in F1-macro than Full-KG, except in the classification of the first level. The reason is that the total number of labels in the first-level classification is smaller. The complete knowledge graph hence enables the LLMs to grasp better the semantic relationships among labels, which can consequently aid Full-KG in achieving improved classification performance. However, as the label space expands, the subgraph retrieval allows LLMs to extract essential information, helping to alleviate performance declines associated with long text inputs.

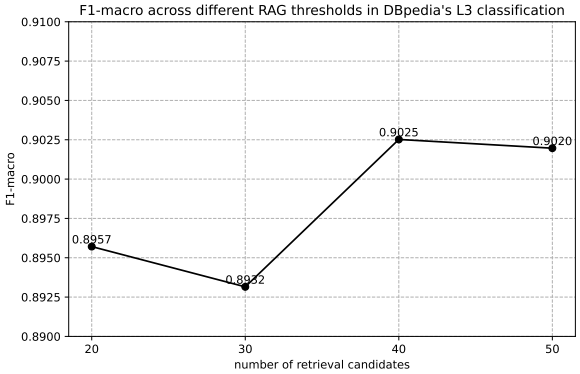

Figure 5. Ablation study on the the impact of the number of retrieval candidates on model performance, with the results shown for Dbpedia's L3 classification.

Table 5. Ablation study on the effectiveness of an open-source LLM `Qwen2.5-8b`.

Dataset	WoS		Dbpedia			Amazon		
	Level 1	Level 2	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
Qwen	0.550	0.367	0.472	0.466	0.691	**0.783**	0.326	0.230
Qwen KG-HTC	**0.715**	**0.509**	**0.699**	**0.575**	**0.799**	0.762	**0.418**	**0.343**

Sensitivity of Subgraph Retrieval. In this experiment, we investigate the impact of the number of retrieval candidates on model performance, with the results shown for Dbpedia's L3 classification from Figure 5. The F1-macro score peaks at 0.9025 when 40 candidates are retrieved. However, the overall performance remains highly stable within the tested range of 20 to 50 candidates. This indicates that in practical applications, meticulous tuning of this parameter is not required to achieve strong results.

Performance in open-source LLMs. In this experiment, we changed the LLM from `GPT-3.5-turbo` to `Qwen2.5-8b`. Table 5 demonstrates that our KG-HTC increases the performance of `Qwen2.5-8b` except for the first-level classification in the Amazon dataset. This enhancement suggests that the improvements offered by our KG-HTC may be applicable to many open-source LLMs.

6 Conclusion

Supervised methods for HTC face challenges in industrial applications. This paper presents KG-HTC, a zero-shot HTC method that leverages knowledge graphs and LLMs for HTC. Our proposed KG-HTC dynamically retrieves subgraphs that are semantically relevant to the input text and constructs structured prompts, thereby significantly enhancing classification performance under strict zero-shot scenarios. Experimental results demonstrate that KG-HTC achieves state-of-the-art performance on three open datasets (Amazon, WoS, and Dbpedia) in the zero-shot setting, with significant improvements observed in the classification of deeper hierarchical labels.

The limitation of this study is the assumption that a complete and correct label taxonomy is available. Consequently, any inaccuracies or errors within the knowledge graph are likely to result in a degradation of our method's performance. Our future research will focus on leveraging large language models to construct knowledge graphs.

Acknowledgements

This research was funded in part by the Luxembourg National Research Fund (FNR) and PayPal, PEARL grant reference 13342933/Gilbert Fridgen. The research was carried out as part of a partnership with Enovos Luxembourg S.A. In fulfillment of the obligations arising from the grant agreement, the author has applied a Creative Commons Attribution 4.0 International license.

References

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *international semantic web conference*, pages 722–735. Springer, 2007.

[2] L. Bongiovanni, L. Bruno, F. Dominici, and G. Rizzo. Zero-shot taxonomy mapping for document classification. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, pages 911–918, 2023.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[4] S. Chatterjee, A. Maheshwari, G. Ramakrishnan, and S. N. Jagarlapudi. Joint learning of hyperbolic label embeddings for hierarchical multi-label classification. In P. Merlo, J. Tiedemann, and R. Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2829–2841, Online, Apr. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.247.

[5] D. Chen, Z. Yu, and S. R. Bowman. Clean or annotate: How to spend a limited data collection budget. In C. Cherry, A. Fan, G. Foster, G. R. Haffari, S. Khadivi, N. V. Peng, X. Ren, E. Shareghi, and S. Swayamdipta, editors, *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 152–168, Hybrid, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.deeplo-1.17.

[6] H. Chen, Q. Ma, Z. Lin, and J. Yan. Hierarchy-aware label semantics matching network for hierarchical text classification. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4370–4379, Online, Aug. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.337.

[7] 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)*, pages 4171–4186, 2019.

[8] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, D. Metropolitansky, R. O. Ness, and J. Larson. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

[9] W. Fan, Y. Ding, L. Ning, S. Wang, H. Li, D. Yin, T.-S. Chua, and Q. Li. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501, 2024.

[10] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, and H. Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2, 2023.

[11] K. Halder, A. Akbik, J. Krapac, and R. Vollgraf. Task-aware representation of sentences for generic text classification. In D. Scott, N. Bel, and C. Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3202–3213, Barcelona, Spain (Online), Dec. 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.285.

[12] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

[13] W. Huang, E. Chen, Q. Liu, Y. Chen, Z. Huang, Y. Liu, Z. Zhao, D. Zhang, and S. Wang. Hierarchical multi-label text classification: An attention-based recurrent network approach. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1051–1060, 2019.

[14] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. Dense passage retrieval for open-domain question answering. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550.

[15] M. Klesel and H. F. Wittmann. Retrieval-augmented generation (rag). *Business & Information Systems Engineering*, 2025.

[16] K. Kowsari, D. E. Brown, M. Heidarysafa, K. Jafari 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)*, pages 364–371, 2017. doi: 10.1109/ICMLA.2017.0-134.

[17] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[18] T. Li, G. Zhang, Q. D. Do, X. Yue, and W. Chen. Long-context llms struggle with long in-context learning. *arXiv preprint arXiv:2404.02060*, 2024.

[19] Z. Li, Q. Zang, D. Ma, J. Guo, T. Zheng, M. Liu, X. Niu, Y. Wang, J. Yang, J. Liu, et al. Autokaggle: A multi-agent framework for autonomous data science competitions. *arXiv preprint arXiv:2410.20424*, 2024.

[20] A. Liu, B. Feng, B. Xue, B. Wang, B. Wu, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[21] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[22] Y. Liu, K. Zhang, Z. Huang, K. Wang, Y. Zhang, Q. Liu, and E. Chen. Enhancing hierarchical text classification through knowledge graph integration. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5797–5810, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.358.

[23] Z. Luo, X. Song, H. Huang, J. Lian, C. Zhang, J. Jiang, and X. Xie. Graphinstruct: Empowering large language models with graph understanding and reasoning capability. *arXiv preprint arXiv:2403.04483*, 2024.

[24] Y. Meng, J. Shen, C. Zhang, and J. Han. Weakly-supervised hierarchical text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6826–6833, 2019.

[25] L. Paletto, V. Basile, and R. Esposito. Label augmentation for zero-shot hierarchical text classification. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7697–7706, Bangkok, Thailand, Aug. 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.416.

[26] D. Patel, P. Dangati, J.-Y. Lee, M. Boratko, and A. McCallum. Modeling label space interactions in multi-label classification using box embeddings. *ICLR 2022 Poster*, 2022.

[27] B. Peng, Y. Zhu, Y. Liu, X. Bo, H. Shi, C. Hong, Y. Zhang, and S. Tang. Graph retrieval-augmented generation: A survey. *arXiv preprint arXiv:2408.08921*, 2024.

[28] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. *OpenAI*, 2018.

[29] A. Simonofski, J. Fink, and C. Burnay. Supporting policy-making with social media and e-participation platforms data: A policy analytics framework. *Government Information Quarterly*, 38(3):101590, 2021.

[30] A. Sun and E.-P. Lim. Hierarchical text classification and evaluation. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 521–528, 2001. doi: 10.1109/ICDM.2001.989560.

[31] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[32] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[33] Q. Zhang, J. Dong, H. Chen, D. Zha, Z. Yu, and X. Huang. KnowGPT: Knowledge graph based prompting for large language models. *Advances in Neural Information Processing Systems*, 37:6052–6080, 2024.

[34] 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. *arXiv preprint arXiv:2403.00165*, 2024.

[35] K. Zhu, Q. Zang, S. Jia, S. Wu, F. Fang, Y. Li, S. Gavin, T. Zheng, J. Guo, B. Li, et al. LIME: Less is more for MLLM evaluation. *arXiv preprint arXiv:2409.06851*, 2024.