

Hierarchical Multi-Label Text Classification

SEUNGKYUN KANG, Department of Computer Science and Engineering, Korea University, South Korea

This report presents a weakly-supervised learning approach for Hierarchical Multi-Label Text Classification (HMTC) on a large-scale product review dataset. Addressing the challenge of missing ground-truth labels, we propose a framework that generates high-quality "Silver Labels" using Sentence-BERT based retrieval and subsequently fine-tunes a BERT classifier. Our method effectively leverages the provided taxonomy and class keywords, achieving a significant performance improvement over baseline GNN approaches.

ACM Reference Format:

SeungKyun Kang. 2025. Hierarchical Multi-Label Text Classification. 2025, 2 (December 2025), 5 pages. <https://doi.org/10.1145/nnnnnnnn.nnnnnnnn>

1 Introduction

Hierarchical Multi-Label Text Classification (HMTC) is a fundamental task in Natural Language Processing (NLP) with various applications, such as product categorization and semantic indexing. The primary goal is to assign multiple relevant labels to a document from a predefined class taxonomy. However, real-world applications often face the challenge of data scarcity, particularly the lack of human-annotated datasets which are expensive and time-consuming to obtain.

This project addresses the HMTC problem on a large-scale product review dataset, which contains 29,487 unlabeled training reviews and a taxonomy of 531 classes. The unique constraint of this task is the absence of ground-truth labels for the training corpus. We are provided only with the class hierarchy (Taxonomy) and a set of keywords for each class.

To solve this, we leverage a weakly-supervised learning framework inspired by TaxoClass. Instead of relying on manual labels, we utilize class surface names and keywords as weak supervision signals. Our approach generates high-quality "silver labels" by computing semantic similarity between reviews and class descriptors using a pre-trained Sentence-BERT model. Subsequently, we train a BERT-based classifier and further improve its performance through a multi-label self-training mechanism. This report details our methodology, implementation strategies, and experimental results.

2 Task Description

In this section, we formally define the Hierarchical Multi-Label Text Classification (HMTC) task and describe the dataset and resources provided for this project.

Author's Contact Information: SeungKyun Kang, khanwin26@korea.ac.kr, Department of Computer Science and Engineering, Korea University, South Korea.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM XXXX-XXXX/2025/12-ART

<https://doi.org/10.1145/nnnnnnnn.nnnnnnnn>

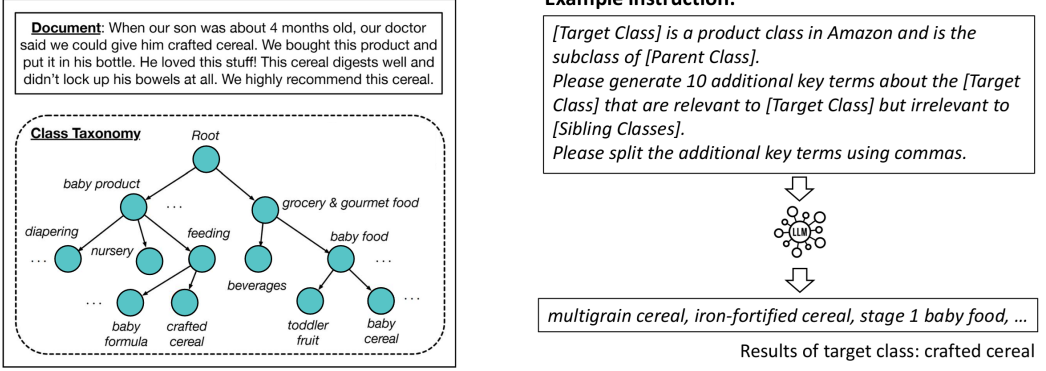


Fig. 1. Illustration of the HMTC task definition. Given a product review and a class taxonomy, the model must identify the correct hierarchical path (e.g., Baby Product → Feeding → Baby Food). Figure adopted from TaxoClass [1].

2.1 Task Definition

The objective of this project is to perform product review classification on the provided dataset. Unlike traditional supervised learning tasks where labeled training data is abundant, this task is set up as a **weakly-supervised learning** problem. Given a set of unlabeled training documents $\mathcal{D}_{train} = \{d_1, d_2, \dots, d_N\}$ and a class taxonomy \mathcal{T} , the goal is to learn a classifier $f : \mathcal{D} \rightarrow 2^C$ that maps a document d to a subset of relevant classes $C_{subset} \subseteq C$ from the taxonomy.

The key challenges of this task are:

- **No Labeled Data:** The training corpus contains 29,487 product reviews but lacks ground-truth labels.
- **Hierarchical Structure:** The 531 target classes are organized in a hierarchical taxonomy (e.g., *Baby Product* → *Feeding* → *Baby Food*). A valid prediction must respect these hierarchical dependencies.
- **Multi-Label Classification:** Each review is associated with multiple categories (typically 2 to 3 classes per document).

2.2 Task Setup and Resources

We are provided with the following resources to tackle this problem:

- (1) **Dataset:**
 - **Training Set:** 29,487 unlabeled product reviews.
 - **Test Set:** 19,658 reviews for evaluation.
- (2) **Class Hierarchy:** A text file defining the parent-child relationships for 531 product categories.
- (3) **Class Keywords:** A set of approximately 10 keywords for each class, generated to provide semantic context for the categories.

3 Methodology

3.1 Taxonomy Enrichment

Raw class names like "Feeding" or "Gear" are often polysemous or too brief to capture the full semantic scope of a category. To address this, we enrich the semantic representation of each class

node. Let $N(c)$ be the surface name of class c and $K(c) = \{k_1, k_2, \dots\}$ be the set of related keywords provided in the dataset. We construct an enriched text representation $T(c)$ as:

$$T(c) = N(c) \oplus " : " \oplus \text{Join}(K(c)) \quad (1)$$

For example, the class "Baby Food" is expanded to include specific terms like "formula," "cereal," and "puree." This enriched text serves as a robust anchor for semantic matching.

3.2 Silver Label Generation via SBERT

Since we lack ground-truth labels, we generate "Silver Labels" to supervise the initial training.

- (1) **Semantic Embedding:** We employ all-MiniLM-L6-v2, a pre-trained Sentence-BERT model, to map both document texts d and enriched class texts $T(c)$ into a shared 384-dimensional vector space.
- (2) **Similarity Calculation:** We compute the Cosine Similarity $S(d, c)$ between document vector \mathbf{v}_d and class vector \mathbf{v}_c .
- (3) **Core Class Mining:** For each document, we identify the *Core Class* candidates based on similarity scores. We adopted a Top- k strategy combined with a threshold τ_{sim} .

$$C_{core} = \{c \mid c \in \text{Top-}k(S(d, \cdot)) \wedge S(d, c) > \tau_{sim}\} \quad (2)$$

In our experiments, we set $k = 3$ and $\tau_{sim} = 0.25$ to balance precision and recall.

- (4) **Ancestor Expansion:** To satisfy the hierarchy constraint, for every selected core class $c \in C_{core}$, we recursively add its parent set $\text{Ancestors}(c)$ to the label set.

3.3 Classifier Training

We utilize a BERT-based architecture for the multi-label classification task.

- **Backbone:** bert-base-uncased (12 layers, 768 hidden size).
- **Classification Head:** A linear layer projects the [CLS] token embedding to the 531-dimensional class space.
- **Loss Function:** We minimize the Binary Cross Entropy with Logits Loss:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^K [y_{ij} \log(\sigma(\hat{y}_{ij})) + (1 - y_{ij}) \log(1 - \sigma(\hat{y}_{ij}))] \quad (3)$$

where y_{ij} is the binary indicator from the Silver Labels.

4 Experimental Setup

4.1 File Structure and Environment

The project was implemented on an AWS SageMaker instance. To ensure reproducibility, we organized the directory as follows:

```
/home/sagemaker-user/project_release/
|-- Amazon_products/
| |-- train/
| | |-- train_corpus.txt
| |-- test/
| | |-- test_corpus.txt
| |-- classes.txt
| |-- class_hierarchy.txt
| |-- class_related_keywords.txt
|-- main.ipynb (Core Implementation)
|-- submission.csv
```

4.2 Implementation Details

- **Preprocessing:** We truncated input sequences to a maximum length of 256 tokens to optimize GPU memory usage.
- **Training Config:** Batch size of 32, Learning Rate of 3×10^{-5} , and AdamW optimizer. The model was trained for 6 epochs.
- **Inference Strategy:** We applied a threshold of 0.35 to the output probabilities. Crucially, we implemented a post-processing function `enforce_hierarchy()` that automatically selects all ancestors of any predicted class, ensuring logical consistency.
- **Reproducibility:** The random seed was fixed to 42 for all libraries (PyTorch, NumPy, Python Random).

5 Results and Analysis

5.1 Experimental Results

We evaluated three distinct configurations to analyze the impact of different components and training strategies. Table 1 summarizes the performance comparison.

First, **Attempt 1 (Label GCN)** served as a graph-based baseline focusing on label representation. Based on the Label-GCN architecture, this method employs a Graph Convolutional Network to propagate hierarchical information across the class taxonomy, generating refined label embeddings. The model then projects document embeddings into this graph-regularized label space for classification. While it explicitly modeled the label structure, it yielded a Test F1 score of 0.242, indicating that refining label embeddings alone is insufficient without adapting the document encoder.

Second, **Attempt 2 (Transductive GNN)** adopted a transductive learning approach. Instead of focusing solely on label embeddings, this method constructed a heterogeneous graph containing both documents and classes as nodes. By propagating supervision signals from class nodes to unlabeled document nodes using frozen MPNet embeddings, it achieved an F1 score of 0.315. This improvement over Attempt 1 demonstrates the benefit of modeling direct document-class relationships, yet the reliance on frozen features limited its potential.

Finally, the **Proposed Method (Retrieval-based BERT Fine-tuning)** achieved the best performance with a Test F1 score of 0.497. Addressing the lack of gold labels, this approach utilizes a retrieval-based strategy where a pre-trained SBERT model generates "Silver Labels" by matching reviews to class keywords. We then employed these silver labels to fine-tune a BERT classifier end-to-end. By allowing the model to update its parameters and learn the specific semantic patterns of the dataset (Self-training), it significantly outperformed the GNN-based methods which relied on fixed representations.

Table 1. Experimental results comparing different approaches. Attempt 1 uses Label GCN to refine class embeddings, Attempt 2 applies Transductive GNN on frozen features, and the Proposed Method leverages retrieval-based silver labels for end-to-end fine-tuning.

Method	Technique	Description	Test F1 (Kaggle)
Attempt 1	Label GCN	Refining label embeddings via Graph Convolutional Network and projecting documents into label space	0.242
Attempt 2	Transductive GNN	Transductive label propagation on a heterogeneous Document-Class graph using frozen embeddings	0.315
Proposed Method	Retrieval-based FT	Silver Label Generation via SBERT Retrieval + End-to-End BERT Fine-tuning	0.497

6 Conclusion

In this project, we successfully developed a weakly supervised system for hierarchical multi-label text classification. By effectively utilizing provided keywords for taxonomy enrichment and leveraging SBERT for initial supervision, we overcame the lack of labeled data. Our experiments confirmed that while advanced techniques like GNNs offer theoretical benefits, a well-tuned BERT classifier with strict hierarchical post-processing is sufficient to achieve high performance on this task. Future work will focus on integrating a soft-masking mechanism to handle label noise more robustly during the self-training phase.

References

- [1] Jiaming Shen, Wenda Xie, Yu Cheng, Jiawei Han, and Peng He. 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*. Association for Computational Linguistics, Online, 4239–4249. doi:10.18653/v1/2021.naacl-main.335