VIETNAM GENERAL CONFEDERATION OF LABOR

TON DUC THANG UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

ĐỖ PHẠM QUANG HƯNG - 520K0127
TRỊNH BẢO TOÀN - 520K0332

HIERARCHY-AWARE TEXT CLASSIFICATION USING DEEP LEARNING APPROACHES

UNDERGRADUATE THESIS OF COMPUTER SCIENCE

HO CHI MINH CITY, 2024

VIETNAM GENERAL CONFEDERATION OF LABOR

TON DUC THANG UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

ĐỖ PHẠM QUANG HƯNG - 520K0127
TRỊNH BẢO TOÀN - 520K0332

HIERARCHY-AWARE TEXT CLASSIFICATION USING DEEP LEARNING APPROACHES

UNDERGRADUATE THESIS OF COMPUTER SCIENCE

Advised by

MSc. Duong Huu Phuc

HO CHI MINH CITY, 2024

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to everyone who has supported and guided me throughout this project. First and foremost, I am sincerely thankful to my advisor, MSc. Duong Huu Phuc, for his invaluable guidance, encouragement, and insightful feedback. Your expertise and patience have been instrumental in shaping this work.

I am also grateful to my colleagues and peers, whose discussions and suggestions have enriched my understanding and helped refine my ideas. Special thanks to for Ms. Trang Le her assistance in report structuring and Mr. Tuan Nguyen for his constructive critiques and moral support. Thank you you both.

I want to acknowledge the support of Mr. Thinh P. Le for providing the resources and environment necessary to carry out this research. I am also appreciative the contribution of my partner – Mr. Toan Trinh, who were also crucial to the success of this project.

Finally, my heartfelt thanks go to my family and friends for their unwavering support and encouragement. Your belief in me has been a constant source of motivation and inspiration.

Thank you all for being a part of this journey.

Ho Chi Minh city, 15th August 2024.

Author

(Signature and full name)

Hung

Do Pham Quang Hung

Toan

Trinh Bao Toan

This thesis was carried out at Ton Duc Thang University.

Advisor: MSc. Duong Huu Phuc

....

(Title, full name and signature)

This thesis is defended at the Undergraduate Thesis Examination Committee was hold at Ton Duc Thang University on … /…/……

Confirmation of the Chairman of the Undergraduate Thesis Examination Committee and the Dean of the faculty after receiving the modified thesis (if any).

CHAIRMAN	**DEAN OF FACULTY**
………………………….	………………………………

DECLARATION OF AUTHORSHIP

We hereby declare that this is our own project and is guided by Msc. Duong Huu Phuc; and that the content research and results contained herein are central and have not been published in any form before. The data in the tables for analysis, comments and evaluation are collected by the main author from different sources, which are clearly stated in the reference section.

In addition, the project also uses some comments, assessments as well as data of other authors, other organizations with citations and annotated sources.

We will take full responsibility for any fraud detected in my thesis. Ton Duc Thang University is not related to the infringing rights, the copyrights that we give during the implementation process (if any).

Ho Chi Minh city, 15th August 2024

Author

(Signature and full name)

Hung

Do Pham Quang Hung

Toan

Trinh Bao Toan

HIERARCHY-AWARE TEXT CLASSIFICATION USING DEEP LEARNING APPROACHES

ABSTRACT

This thesis explores the application of deep learning techniques to hierarchical text classification. The objectives here are to do a systematic survey on text classification with hierarchically structured documents and conduct experiments with different language models. Hierarchical text classification involves categorizing documents into a structured set of labels with varying levels of specificity, presenting challenges such as maintaining accuracy across different hierarchical levels. For instance, accurately classifying research papers into broad categories like "Computer Science" and more specific subfields such as "Artificial Intelligence" or “Computer Vision” requires sophisticated models that respect hierarchical relationships of labels. The thesis will utilize three datasets, namely WOS-46985, which includes a wide range hierarchical structure of 46985 academic papers, the NYTimes Annotated Corpus, providing 1.8 million annotated news articles, and RCV1-v2 dataset, a collection of 804,414 newswire documents produced by Reuters in 1996-1997. By evaluating different approaches, we aim to enhance understanding of hierarchy-aware classification and possibly improve performance on complex text classification tasks.

PHÂN LOẠI VĂN BẢN PHÂN CẤP SỬ DỤNG KỸ THUẬT HỌC SÂU

TÓM TẮT

Luận văn này nghiên cứu việc ứng dụng các kỹ thuật học sâu vào phân loại văn bản theo cấu trúc phân cấp. Mục tiêu của đề tài này là tiến hành khảo sát có hệ thống về phân loại văn bản với các tài liệu có cấu trúc phân cấp và thực hiện các thí nghiệm với các mô hình ngôn ngữ khác nhau. Phân loại văn bản theo cấu trúc phân cấp liên quan đến việc phân loại tài liệu thành một tập hợp các nhãn có cấu trúc với các mức độ cụ thể khác nhau, đồng thời trình bày những thách thức như duy trì độ chính xác trên các cấp độ phân cấp khác nhau. Ví dụ, việc phân loại chính xác các bài báo nghiên cứu vào các loại chung như "Khoa học Máy tính" và các lĩnh vực con cụ thể như "Trí tuệ Nhân tạo" hoặc "Thị giác Máy tính" yêu cầu các mô hình tinh vi nhận biết được các mối quan hệ phân cấp của các nhãn. Luận văn sẽ sử dụng ba tập dữ liệu, cụ thể là WOS-46985, bao gồm 46.985 bài báo học thuật được phân cấp thành nhiều lĩnh vực; Bộ dữ liệu NYTimes Annotated Corpus, cung cấp hơn 1,8 triệu bài báo tin tức đã được chú thích; và tập dữ liệu RCV1-v2, một tập hợp 804.414 tin tức được tập hợp bởi hãng thông tấn Reuters (Anh Quốc) vào những năm 1996-1997. Bằng cách đánh giá các phương pháp khác nhau, chúng tôi hướng tới việc nâng cao hiểu biết về phân loại văn bản theo cấu trúc phân cấp và có thể cải thiện hiệu suất trên các nhiệm vụ phân loại văn bản phức tạp.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF EQUATIONS

`

LIST OF TABLES

ABBREVIATIONS

BERT	Bidirectional Encoder Representation Transformer
DistilBERT	Distilled BERT
DAG	Directed Acyclic Graph
HMC	Hierarchical Multi-Label Classification
HTC	Hierarchical Text Classification
NLP	Natural Language Processing
RoBERTa	Robust BERT Approach
SOTA	State-of-the-art
TC	Text classification

CHAPTER 1. INTRODUCTION

1.1 Introduction

Text classification (TC) is a central task in the field of natural language processing (NLP). In essence, TC involves using supervised learning algorithms to assign predefined labels to text documents. The most common form of TC is multiclass classification, where each document is assigned a single label from a set of possible classes. This could involve binary classification with two labels or multiclass classification with more options. In contrast, multi-label classification allows each document to be assigned multiple labels simultaneously. TC has numerous practical applications, including topic labeling, sentiment analysis, and named entity recognition.

From a theoretical perspective, binary classification represents the most basic scenario, assuming the categories are stochastically independent [1]. Under this assumption, the problem can be divided into multiple independent binary classification tasks. However, this assumption does not hold in all practical scenarios, such as in hierarchical text classification (HTC), which is a specialized area within TC and part of the broader hierarchical multi-label classification (HMC). [2] describes HMC as a task where instances may belong to multiple classes simultaneously and are organized within a hierarchical structure. In such cases, the dependency between labels cannot be ignored, as they are interrelated according to the hierarchy.

The key feature of HTC is that it involves labels organized in a multi-level hierarchy, where each label is represented as a node with potentially many children. This hierarchical structure is crucial for effective classification, as it reflects the inherent organization of categories in many real-world datasets. For example, document collections often have labels arranged in hierarchical taxonomies, such as a “sports” category encompassing “tennis” and “football.” Even datasets without an initially defined hierarchy can often be organized into macro-categories and subcategories, facilitating intuitive modeling of complex label relationships, such as “is-a” or “part-of.”

The growing availability of hierarchical TC datasets, combined with industrial interest in TC applications, has led to the development of numerous HTC methods. These methods extend beyond traditional TC applications and are valuable in areas such as International Classification of Diseases (ICD) medical coding, legal document concept labeling, patent classification, and IT ticket categorization. The advantage of hierarchical classifiers lies in their ability to utilize label dependencies to enhance classification performance. This is particularly beneficial in multi-label TC, where managing a large number of related labels can be challenging. Hierarchical methods often demonstrate improved generalization capabilities when encountering new classes, as they can leverage knowledge from parent nodes of newly introduced categories.

1.2 Problem statement

Hierarchical text classification (HTC) is a text classification problem where the label set is predefined and represented as a directed acyclic graph (DAG), referred to as the label hierarchy. Each label that appears in the corresponding dataset corresponds to a unique node in the hierarchy. Each non-root node is pointed to by only one node from the upper level, which is its parent node. In the ground-truth label set Y of any sample, a non-root label y_i always appears simultaneously with its parent nodes; in other words, for any $y_i \in Y$, the parent node of y_i is also in Y. For a document D that needs to be classified, where $\mathrm{D} = \{\mathrm{w}_1, \mathrm{w}_2, \dots, \mathrm{w}_\mathrm{N}\}$ is typically considered a sequence with N tokens, an HTC model will predict a subset $Y\hat{}$ of the complete label set Y.

HTC involves categorizing documents into a structured set of labels organized in a hierarchy, where labels at higher levels are more general and those at lower levels are more specific. For example, in tasks like categorizing research papers, a traditional flat classification model might struggle with differentiating between broader categories like "*Computer Science*" and more specific subcategories like "*Natural Language Processing*" and "*Deep Learning.*"

The complexity of HTC arises from the need to accurately capture relationships between categories at different levels while maintaining consistent and relevant predictions across the hierarchy. Traditional flat multi-label classification approaches struggle with this hierarchical system, leading to issues such as misclassification across levels, propagation of errors down the hierarchy, and failure to leverage the relationships between parent and child categories. Despite advancements in deep learning and transformer models like BERT, effectively incorporating hierarchical dependencies remains challenging still. Addressing these challenges requires models that not only perform well in general text classification but also respect the hierarchical structure of labels, maintain consistency and accuracy at each label level.

1.3 Thesis scope

Our thesis focuses on text classification model for hierarchical text classification using contrastive learning. The thesis will be conducted, tested and evaluated on three data sets, Web of Science (WOS-46895), NYTimes Annotated Corpus and RCV1-v2.

The Web of Science (WOS) dataset is a comprehensive collection of scholarly literature encompassing various academic disciplines. Maintained by Clarivate Analytics, it serves as a valuable resource for researchers, offering extensive bibliographic information and citation data. The dataset includes metadata from scientific articles, conference proceedings, and other academic publications, spanning several decades. Its strength lies in providing a detailed view of the interconnections between scholarly works through its citation network. Researchers often utilize the WOS dataset for bibliometric analyses, trend identification, and impact assessment in scientific literature [3].

The New York Times Annotated Corpus [4] is a rich dataset comprising articles published in The New York Times over a two-decade period from the late 1980s to the late 2000s. This corpus stands out for its comprehensive annotation, making it a valuable resource for various natural language processing and information retrieval tasks. The dataset includes not just the full text of articles but also extensive metadata such as section classifications, taxonomic classifiers, and named entities. Its use in the Text REtrieval Conference (TREC) Core Track highlights its significance in developing and evaluating information retrieval systems.

The thesis also experiment HILL with RoBERTa and DistilBERT architecture, which are two variants of the BERT model designed to improve upon the original architecture.

RoBERTa [5] builds on BERT by using more training data and longer training times, optimizing performance with dynamic masking and removing the next-sentence prediction objective. It achieves better accuracy on various NLP tasks by leveraging a more robust training approach.

DistilBERT [6] on the other hand, is a smaller, faster version of BERT. It is created through knowledge distillation, a process that transfers knowledge from a larger model to a smaller one. DistilBERT maintains around 97% of BERT's language understanding capabilities while being significantly more efficient.

The MPNet model [7] (Masked and Permuted Pre-training) is a transformer-based language model that improves upon previous pre-training methods like BERT and XLNet by combining the strengths of both. MPNet uses a novel pre-training strategy that incorporates both masked language modeling (MLM) and permuted language modeling (PLM). MPNet captures dependencies among masked tokens by permuting the input sequence, enabling the model to better understand context. This leads to more accurate language representations and improved performance on a wide range of natural language processing tasks, such as text classification, sentiment analysis, and question answering.

For this thesis, we choose above models can be addressed shortly by their strengths, RoBERTa is robust and highly optimized, offering improved performance over BERT due to more extensive pretraining and dynamic masking. DistilBERT is a smaller, faster variant of BERT, maintaining around 97% of its performance while being lighter and faster, making it ideal for resource-constrained environments. MPNet combines the strengths of BERT and XLNet, offering better generalization and capturing word dependencies more effectively, also being amongst the best embedding models at the moment. In conclusion, these models may offer a more general view on how different embedding models affect performance of HTC task, especially in encoding label hierachy.

1.4 Thesis contributions

This thesis focuses on three outcomes. First, we will research traditional and deep learning model contributions to the hierarchical text classification (HTC) task. Secondly, we will implement a novel proposed contrastive learning model for HTC and conducting the experiments under the different hyperparamters settings. Finally, we will experiment with two more BERT variants and MPNet model within the novel framework to assess their impact on performance and determine if they offer improvements over the original model.

1.5 Report structure

This report is structured into five chapters, each addressing a critical aspect of the research on the hierarchical text classification. The progression of the chapters is designed to guide readers through the background, methodology, experimental validation, and conclusion of this thesis.

Table 1 Report structure

Chapter	**Description**
Chapter 1: Introduction	This chapter introduces the problem statement, our objectives and scope. It also provides a brief overview of the methodology.
Chapter 2: Related work	Within this chapter, we survey methods of hierarchical text classification, includes both traditional and deep learning approaches. This chapter will introduce the concepts, models, and theories that initialize the foundation and motivation for contrastive learning technique development.
Chapter 3: Methodology	We will provide a specific explanation on how the novel contrastive learning model has improved from its predecessors, and a deeper look in information lossless contrastive learning

Chapter	Description
Chapter 4: Experiment and results	In this chapter, our experiments with different text embedding models. The collected data is also reported in this section.
Chapter 5: Conclusion and Future work	This chapter summarizes the novelty of new contrastive learning approach and discusses their broader implications in hierarchical text classification.

In summary, this chapter has introduced the research task, outlined the objectives and scope, and provided an overview of the contributions of this thesis. In the next chapter, we will explore the landscape of hierarchical text classification by reviewing related works. This will include a discussion of both traditional and more recent deep learning approaches, highlighting the key advancements that have paved the way for contrastive learning.

CHAPTER 2. RELATED WORK

2.1 Background

Hierarchical Text Classification is a challenging task that involves assigning text documents to multiple categories within a predefined hierarchical structure. Traditionally, HTC approaches relied heavily on feature engineering, a labor-intensive process requiring domain expertise to extract relevant information from raw text data. These methods often struggled with capturing complex semantic and syntactic relationships inherent in text, limiting their performance.

Existing self-supervised methods in Natural Language Processing (NLP), particularly in hierarchical text classification (HTC), primarily focus on self-supervised contrastive learning, heavily relying on human-designed augmentation rules to create contrastive samples, which can distort or obscure the original information.

Most previous studies on HTC have focused on using dual encoders to process text and the label hierarchy separately, but they combine the outputs of the encoders in a simplistic manner. Some other studies attempt to integrate the label hierarchy into language models like BERT through contrastive learning techniques, but this learning process fundamentally still relies on data augmentation, which may lead to the loss of important semantic information for prediction.

In recent years, the emergence of deep learning has revolutionized various fields, including natural language processing. By automatically learning hierarchical representations from raw text, deep learning models have shown remarkable success in overcoming the limitations of traditional feature engineering methods. This section will delve into the evolution of HTC techniques, exploring the transition from feature engineering-based approaches to the current state-of-the-art deep learning models.

2.2 Traditional approaches in HTC

The "Hierarchical Text Classification with Support Vector Machines" by Lijuan Cai and Thomas Hofmann. This paper explores the use of Support Vector Machines (SVMs) for hierarchical text classification. The authors propose a hierarchical SVM approach where the classification task is broken down into a series of binary classification problems arranged in a tree structure. [8]

Decision trees for hierarchical multi-label classification [2] by Vens et al. explores the application of decision tree algorithms for hierarchical multi-label classification (HMC) tasks, particularly in functional genomics. The authors compare three decision tree approaches: learning a single HMC tree (CLUS-HMC), learning a separate tree for each class (CLUS-SC), and learning separate trees hierarchically (CLUS-HSC). They evaluate these methods on yeast gene function prediction datasets using a tree-structured classification scheme (FunCat) and a DAG-structured scheme (Gene Ontology). Their findings suggest that CLUS-HMC outperforms the other methods in terms of predictive accuracy, model size, and induction time, making it a suitable approach for HMC tasks requiring interpretable models. The study also highlights the importance of precision-recall based evaluation measures for assessing and understanding the performance of HMC methods.

The paper “Improving Text Classification by Shrinkage in a Hierarchy of Classes” by McCallum et al. [9] introduces a new method for text categorization using a hierarchical naive Bayes model with a shared vocabulary. The traditional naive Bayes model for text categorization uses a "bag of words" approach, where each document is represented as a set of words, and the order of the words is not considered. However, this approach ignores the fact that words often have different meanings depending on the context in which they are used. For example, the word "bank" might refer to a financial institution or the edge of a river, depending on the other words in the document.

The hierarchical naive Bayes model addresses this problem by using a hierarchy of word senses. Each word sense is associated with a probability distribution over the categories, and the model learns these probabilities from the training data. The model then uses these probabilities to classify new documents by calculating the probability of each category given the word senses in the document. The paper shows that the hierarchical naive Bayes model outperforms the traditional naive Bayes model on a number of text categorization tasks. This is because the hierarchical model is better able to capture the context of words and to learn more accurate probability distributions.

2.3 Deep learning approaches for HTC

2.3.1 Deep neural networks

HDLTex, [10] or Hierarchical Deep Learning for Text Classification, introduces an innovative method for classifying documents arranged in a hierarchical structure. This is one of the very first work to employs deep neural networks in HTC task. Moreover, HDLTex contribute a re-known dataset for HTC – the **WOS-46985**, which used by many following works, included ones mentioned in this thesis. Unlike traditional multi-class classification techniques, which often struggle with large numbers of categories, HDLTex addresses this issue by employing specialized deep learning architectures tailored for each level of the document hierarchy.

First, it embraces Hierarchical Classification by acknowledging the hierarchical nature of document categories and sub-categories, thereby improving the organization and comprehension of documents within a particular field. Second, it leverages a range of Deep Learning Architectures including Deep Neural Networks (DNNs), Recurrent Neural Networks (RNNs) such as GRU and LSTM, and Convolutional Neural Networks (CNNs).

In terms of Feature Extraction, HDLTex applies various methods suited to the specific deep learning model in use. For CNNs and RNNs, text vector-space models like GloVe with 100 dimensions are employed, while DNNs utilize count-based and term frequency–inverse document frequency (TF-IDF) techniques, including the analysis of N-gram.

Finally, HDLTex utilizes Stochastic Gradient Optimization (SGD) methods such as RMSProp and Adam. Comparisons with traditional methods like SVM and Naive Bayes reveal HDLTex’s advantages, particularly in the context of hierarchically organized documents. Empirical tests on datasets from the Web of Science have shown HDLTex’s superior accuracy in both top-level and sub-level document classifications.

The work of HiAGM [11] presents some important contributions. This is a novel end-to-end model for HTC that leverages fine-grained hierarchy information to aggregate label-wise text features. It employs traditional structure encoders, like bidirectional Tree-LSTM and hierarchy-GCN, to model label dependencies in both top-down and bottom-up approaches, incorporating prior hierarchical knowledge. This innovative method, which has not been explored before in hierarchical text classification, offers a fresh perspective on the problem. HiAGM introduces a novel end-to-end hierarchy-aware global model with two variants: HiAGM-LA and HiAGM-TP. HiAGM-LA is a multi-label attention model that learns hierarchy-aware label embeddings via the hierarchy encoder and performs inductive fusion of label-aware text features. In contrast, HiAGM-TP is a text feature propagation model that directly inputs text features into hierarchy encoders, producing hybrid information in a deductive manner. Empirical results show that HiAGM consistently improves performance across various datasets with different structure encoders. For instance, on the RCV1-V2 dataset, the best-performing model exceeds the state-of-the-art by 3.25% in Macro-F1 and 0.66% in Micro-F1. According to

2.3.2 First contrastive learning for HTC

Hierarchy-guided Contrastive Learning [12] (HGCLR) is a novel approach for Hierarchical Text Classification. This is the very first work using contrastive learning in HTC. Our main model - the HILL framework, derived and compared directly to this HGCLR work.

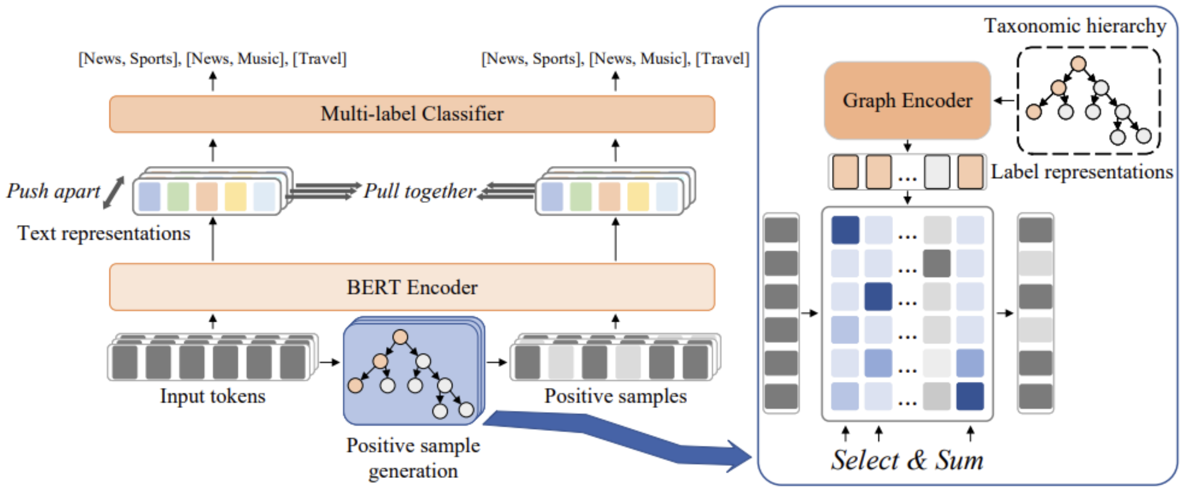


Figure 1 An overview of HGCLR

HGCLR directly embeds the hierarchy into a text encoder (like BERT), enabling the model to learn hierarchy-aware text representations independently and uses a customized Graphormer, a state-of-the-art graph encoder, to understand the label structure very effectively. This encoder leverages label embeddings, label name embeddings (which are particularly impactful and contribute to understanding label features), spatial encoding, and edge encoding to capture the relationships between labels in the hierarchy.

Also, HGCLR constructs positive samples by identifying and retaining important keywords in the text while masking other tokens. This selection process leverages the attention weights between token embeddings and label features. Tokens with attention weights above a certain threshold (indicating high relevance to the label) are considered important and retained in the positive sample. This approach ensures that the positive samples remain closely related to the original text while emphasizing label-relevant information.

Through above components, HGCLR integrates hierarchy information into the text representation learning process, result to improved performance in HTC tasks. It outperforms previous methods on datasets like WOS and NYT, demonstrating its effectiveness. Yet, *HGCLR still rely on data augmentation process, which possibly leads to distortion in data properties, later* [13] *point out as weakness and make improvement as "lossless" learning process*.

Previous Hierarchical Text Classification (HTC) methods, particularly those categorized as global approaches, has limitations in how they incorporate and leverage the hierarchical structure of labels.

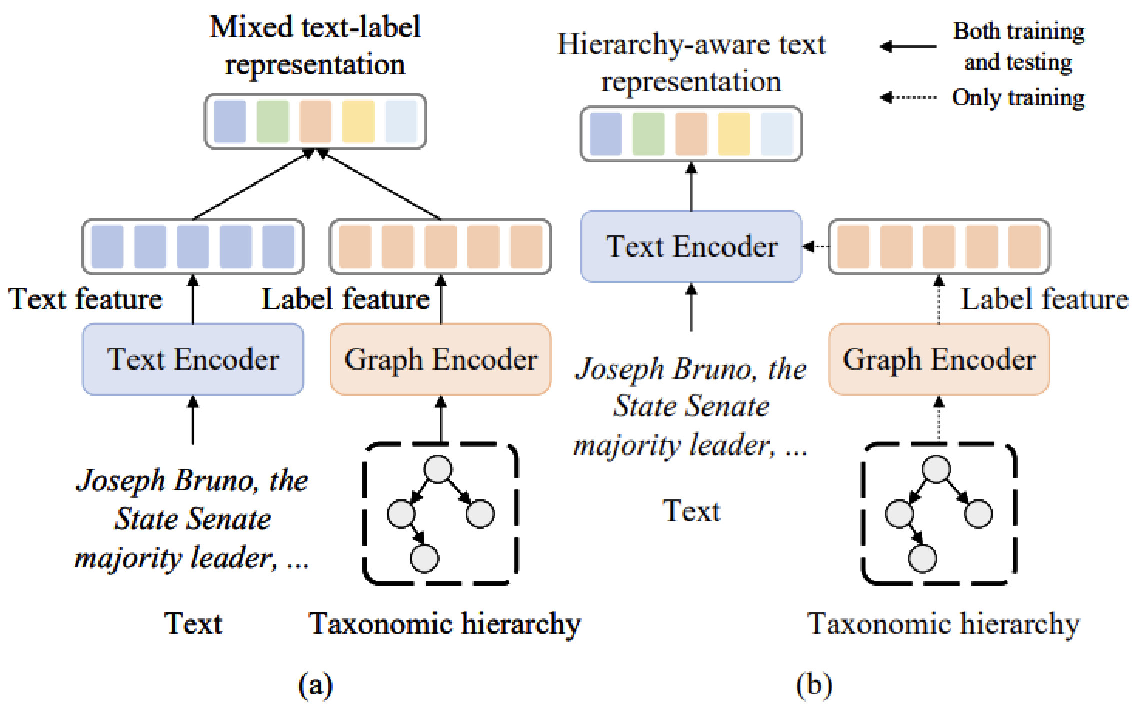

Mixed text-label representation
Text feature
Label feature
Text Encoder
Graph Encoder
Joseph Bruno, the State Senate majority leader, ...
Text
Taxonomic hierarchy
(a)
Hierarchy-aware text representation
Both training and testing
Only training
Text Encoder
Label feature
Joseph Bruno, the State Senate majority leader, ...
Graph Encoder
Text
Taxonomic hierarchy
(b)

Figure 2 Two ways of introducing hierarchy information. (a) Previous work model text and labels separately and find a mixed representation. (b) HGCLR incorporating hierarchy information into text encoder for a hierarchy-aware text representation [12]

Early global approaches often disregarded the hierarchical structure entirely, treating the task as a flat multi-label classification problem. This simplification omits the valuable relationships between labels inherent in the hierarchy.

Later global approaches attempted to integrate label structure using techniques like recursive regularization, reinforcement learning, capsule networks, and meta-learning. While these methods tried to capture hierarchical information, they often fell short of directly encoding the holistic label structure, limiting their ability to exploit hierarchical relationships for better performance.

More recent work, although showing improvement, still relies on encoding text and label hierarchies separately and then finding a mixed representation for classification. This separate encoding, while attempting to capture interactions between text and hierarchy, create redundancy. Since the label hierarchy is constant for all text inputs, the graph encoder generates the same representation regardless of the specific text being processed, leading to potentially less effective interaction and learning.

2.4 Text Representation

2.4.1 Word embeddings

Traditional text representation methods, such as TF-IDF weighted Bag-of-Words (BoW), fall short in capturing the semantic and syntactic nuances of text. A significant advancement in creating more meaningful text representations came with the introduction of word embeddings, popularized by models like Word2Vec [14] and GloVe [15]. These embeddings are vector representations of words learned through unsupervised language modeling tasks. Language modeling, a technique with a long history, involves constructing statistical models based on word prediction tasks. These models, known as language models (LMs), excel in generating useful representations of words and sentences.

The word embeddings are derived from shallow neural networks that are pre-trained on extensive document corpora. This training process enables the models to generate vectors that encapsulate the semantic properties of words. The effectiveness of these embeddings is often demonstrated through vector arithmetic, such as the famous example where the vector operation $\overrightarrow{king} - \overrightarrow{man} + \overrightarrow{woman}$ approximates the vector for $\overrightarrow{queen}$. These word embeddings serve as enhanced input features for various downstream algorithms, including classifiers, leading to significant improvements in performance. [16]

2.4.2 Language model

Language models represent the next evolution in text representation, where everything gets contextualized. Unlike static word embeddings, these models dynamically generate word representations based on the surrounding context within a text. This adaptability allows them to capture the nuances of meaning that can vary depending on context, further advancing the ability to understand and process natural language.

Early word embeddings were often described as "static" because they produced fixed representations that struggled to disambiguate the meanings of polysemous words—words with multiple meanings. For instance, an embedding for such a sentence would be an average of its own words' embeddings, leading to a loss of nuanced information. Although significant research has aimed at incorporating context into word embeddings, as exemplified by models like ELMo [17], the advent of the Transformer architecture marked a revolutionary development in creating contextualized language models.

In essence, contextualized models enhance word embeddings by considering the surrounding words in a sentence, thus capturing the context in which a word appears. A key innovation of Transformer-based models is their departure from recurrence-based architectures, such as Recurrent Neural Networks (RNNs), which were previously dominant for text processing due to their ability to handle sequential data effectively. Instead, Transformers rely on an attention mechanism, which allows them to process words in parallel rather than sequentially. This shift has enabled Transformers to scale effectively with increased model depth and larger datasets, a capability that RNNs lack. Consequently, Transformer-based models have become increasingly prominent in recent research, thanks to their efficiency and scalability.

The introduction of the Transformer architecture paved the way for a new generation of language models, among which the BERT (Bidirectional Encoder Representations from Transformers) family stands out as a significant milestone. BERT, introduced by Devlin et al. in 2018, represents a major advancement in contextualized language models. Unlike previous models that processed text in a unidirectional manner, BERT employs a bidirectional approach, meaning it considers both the left and right context of each word simultaneously. This bidirectional context allows BERT to achieve a deeper understanding of word meanings based on their surrounding words, addressing the limitations of static embeddings.

The BERT model is built on the Transformer architecture, utilizing its attention mechanism to generate contextually rich representations of words. This capability enables BERT to excel in a wide range of natural language processing tasks, from question answering to sentiment analysis. Following the success of BERT, a series of related models have emerged, further refining and expanding its capabilities. Notable among these are RoBERTa [18] (Robustly optimized BERT approach), which improves upon BERT by adjusting training parameters and data, and DistilBERT [6], which provides a more compact and efficient version of BERT while retaining much of its performance.

These advancements in the BERT family illustrate the ongoing evolution of contextualized language models, leveraging the Transformer architecture to achieve more nuanced and effective text representations. The continued development and application of these models highlight their transformative impact on natural language understanding and processing (Devlin et al., 2018; Liu et al., 2019; Sanh et al., 2019).

In this chapter, we have conducted a comprehensive review of the landscape of hierarchical text classification (HTC), exploring both traditional and modern deep learning approaches. We began by examining conventional methods that laid the foundation for HTC, including early machine learning techniques. These approaches, while foundational, often struggled with scalability and the complexity of hierarchical structures in text data. Notwithstanding the fact that deep learning models are already capable of solving that data scalability problem, it still undergoing data augmentation, which distort or losing important semantic information of original information, and raise the need of an information “lossless” approach.

In the next chapter, we will transition to discussing a novel methodology, which introduces an information lossless approach using contrastive learning. This innovative method aims to address some of the limitations identified in the existing literature as stated above and offer new avenues for improving hierarchical text classification.

CHAPTER 3. METHODOLOGY

3.1 Information Lossless Contrastive Learning Approach

Hierarchical Information Lossless Contrastive Learning [13] (HILL) is a lossless contrastive learning strategy for HTC, consisting of a text encoder representing the input document and a structural encoder that directly generates positive samples.

The Structural Encoder employs a structural entropy minimization algorithm to extract structural information from the label hierarchy, constructing an optimal encoding tree that minimizes structural complexity. It incorporates a hierarchical representation learning mechanism to embed this structural information into text embeddings. This process involves propagating information from leaf nodes layer by layer until it reaches the root node, thereby capturing multi-granularity information from nodes at different levels. As a result, the Structural Encoder generates a positive view of the document while preserving both semantic and structural information in a lossless manner. Additionally, the Contrastive Learning Module utilizes document embeddings from the text encoder and structural embeddings from the structural encoder to form positive pairs. To achieve effective contrastive learning, it applies the NT-Xent loss function, further enhancing the learning process.

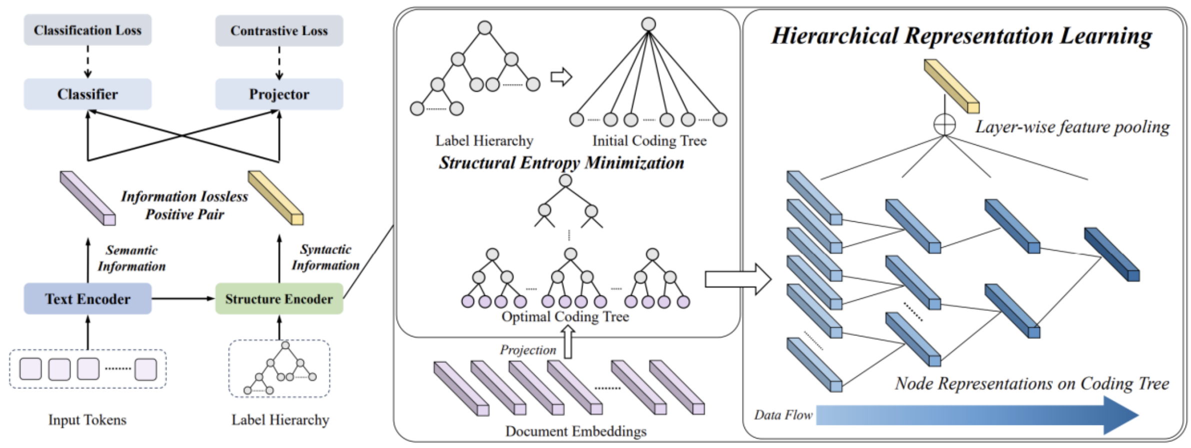


Figure 3 Example HILL model with K=3 [13]

Another point worth noticing is that they train HILL simultaneously with a classification loss function (implemented using binary cross-entropy loss) to learn both supervised information and contrastive information.

HILL addresses the limitations of previous research by avoiding data augmentation and effectively combining the text encoder and structural encoder. Instead of altering the input data, HILL extracts structural information from the label hierarchy and directly incorporates it into the learning process, preserving as much semantic information in the text as possible. Secondly, HILL allows the text encoder and structural encoder to learn from each other through the contrastive learning mechanism, leveraging both semantic information from the text and structural information from the label hierarchy.

HILL's approach to information lossless contrastive learning diverges from traditional data augmentation techniques used in hierarchical text classification (HTC) by directly infusing extracted syntactic information into text representations instead of altering the input document through augmentation. Traditional data augmentation often employs techniques like masking or replacing words in the input document to create variations for contrastive learning, aiming to improve model robustness by exposing it to diverse input representations. However, this approach has a significant drawback: it may inadvertently remove or distort important semantic information from the original text, hindering downstream prediction accuracy.

In contrast, HILL's information lossless contrastive learning focuses on preserving the maximum possible semantic information from the input document. It utilizes a structure encoder to extract essential syntactic information from the label hierarchy. Rather than augmenting the input text, the structure encoder generates feature vectors by merging textual and structural information, injecting this extracted syntactic information directly into the text embeddings. This enriches the representation without altering the original text, thereby retaining all crucial semantic information while incorporating valuable structural insights from the label hierarchy.

The theoretical support for HILL's approach defines information lossless learning in the context of HTC and demonstrates that the information retained by HILL represents the upper bound compared to any augmentation-based method. HILL's structure encoder minimizes structural entropy and employs hierarchical representation learning, capturing the minimal but sufficient information necessary for downstream tasks. This method directly enhances the text representation without relying on potentially destructive data augmentation techniques.

3.2 Text Encoder

The Text Encoder is commonly used in hierarchical text classification tasks and can be built based on various text representation models. However, for consistency, BERT has been used in HILL as the Text Encoder.

More specifically, the operation of the Text Encoder is described as follows:

First, the input text D is tokenized into a sequence of N tokens and then two special tokens are added: The tokens $[CLS]$ and $[SEP]$ are recognized as the start and end of the text, respectively.

$$\widetilde{D} = \{[CLS], w_1, w_2, \dots, w_N, [SEP]\}$$

Equation 1 Document fed to BERT with 2 padding tokens, where the first special token's embedding will represent the entire document

Next, the BERT encoder takes the token-augmented text $\tilde{D}$ as input and generates hidden representations for each token, specifically $H_{\{BERT\}} = F_{\{BERT\}}\ \tilde{D}$, where $H_{\{BERT\}} \in \mathbb{R}^{\{(N+2)\times\, d_B\}}$ is the token embedding matrix and $F_{\{BERT\}}\ (\cdot)$ represents the overall BERT model.

Then, the embedding corresponding to $[CLS]$ is taken as the representation of the entire text. This means that $h_D = H_{\{[CLS]\}}^{\{BERT\}} = H_0^{\{BERT\}}$, where $h_D \in \mathbb{R}^{\{d_B\}}$ is the text embedding and d_B is the hidden size of BERT.

In short, the Text Encoder module takes the input text, processes it, and transforms it into a semantic representation vector for that text.

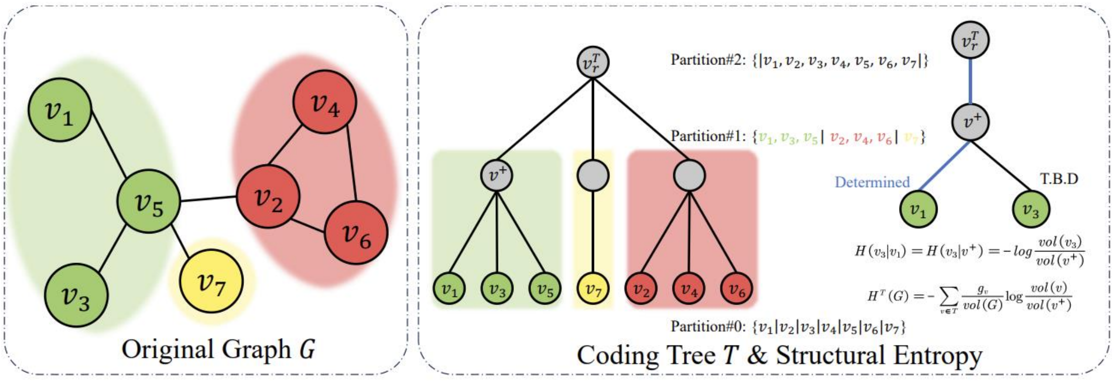
3.3 Structure Encoder

3.3.1 Structural Entropy

Structural entropy, as defined by Li and Pan, is a way to measure the complexity of a graph's structure. It builds upon the idea of Shannon entropy, which quantifies the average amount of information in a message. Instead of messages, structural entropy considers random walks on a graph. Imagine you're walking randomly along the edges of a graph. Structural entropy measures how much information, on average, is needed to pinpoint your location as you move between nodes.

To calculate structural entropy, the concept of a "coding tree" is crucial. A coding tree is a hierarchical representation of the graph, where each node in the tree corresponds to a subset of nodes in the original graph. The root of the coding tree represents the entire graph, while its leaves represent individual nodes in the original graph. The coding tree provides a way to encode the structure of the graph at different levels of granularity.

The structural entropy of a graph, then, is calculated based on the coding tree and reflects the average length of "codewords" needed to describe random walks on the graph under the coding scheme provided by the coding tree. Each "codeword" signifies a path taken during the random walk and is determined by the nodes traversed and the structure of the coding tree. A lower structural entropy implies a less complex graph structure, as shorter "codewords" are sufficient to describe the random walks.


v_1
v_4
v_5
v_2
v_6
v_3
v_7
Original Graph G
v_r^T
Partition#2: $\{|v_1, v_2, v_3, v_4, v_5, v_6, v_7|\}$
Partition#1: $\{v_1, v_3, v_5 | v_2, v_4, v_6 | v_7\}$
v^+
Determined
T.B.D
$H(v_3|v_1) = H(v_3|v^+) = -log\frac{vol(v_3)}{vol(v^+)}$
$H^T(G) = -\sum_{v \in T} \frac{g_v}{vol(G)} \log \frac{vol(v)}{vol(v^+)}$
Partition#0: $\{v_1|v_2|v_3|v_4|v_5|v_6|v_7\}$
Coding Tree T & Structural Entropy

From the paper, provides a formula to calculate the structural entropy (denoted as HT(G)) of a graph G given its coding tree T:

$$H^T\ (G) = \ -\sum(v \in T)[\frac{g_v}{vol(G)}] * \log\frac{vol(v)}{vol(v^+\ )}$$

- g_v: Represents the number of edges that cross the boundary of the node v's partition in the original graph G.
- $vol(G)$: Represents the total number of edges in graph G (also known as the volume of the graph).
- $vol(v)$: Denotes the sum of the degrees of all nodes within the partition represented by node v in the coding tree.
- $vol(v^+)$: Represents the sum of the degrees of all nodes within the partition represented by the parent node (v^+) of node v in the coding tree.

The term $[gv\ /\ vol(G)]$ indicates the probability of a random walk in G involving the subset of nodes represented by node v. The logarithmic term log $[vol(v)/vol(v+)]$ quantifies the information needed to pinpoint a node within v's partition given that the random walk has reached its parent node (v^+).

In essence, structural entropy provides a quantitative measure of the structural information embedded within a graph by considering the information required to describe random walks under a specific coding scheme provided by the coding tree.

3.3.2 Structural Entropy Minimization

Structural Entropy Minimization in the paper describes the process of constructing an optimal coding tree for a given graph (specifically, the label hierarchy graph) to minimize its structural entropy. The goal is to represent the graph structure efficiently, encoding the relationships between labels with minimal information loss.

Structural entropy, as defined in the above, quantifies the complexity of a graph structure (in this case, the label hierarchy) by considering the average information needed to navigate the graph. A lower structural entropy indicates a more organized and informative structure. HILL utilizes two algorithms to construct a coding tree that represents the label hierarchy with minimal structural entropy. The goal is to find a hierarchical partitioning of labels that captures the essential relationships between them efficiently.

- Algorithm 1 focuses on building the overall coding tree to a desired height 'K' by strategically invoking Algorithm 2 and integrating the resulting sub-trees.

- Algorithm 2 constructs sub-coding trees of height 2. It uses operations like 'T.compress(.,.)' to merge nodes and 'T.delete(.)' to remove nodes, always prioritizing choices that maximize structural entropy reduction.

Once the optimal coding tree (with minimal structural entropy) is constructed, HILL's structure encoder performs hierarchical representation learning on this tree. The leaf nodes of the coding tree are initialized with representations derived from the input document. Then, the structure encoder propagates information from the leaf nodes up the tree, iteratively computing representations for higher-level nodes until it reaches the root node. This process effectively encodes the structural relationships between labels into the final representation. The work of HILL argues that minimizing structural entropy helps preserve the essential information within the label hierarchy, leading to information lossless learning. By encoding the label relationships in a structure-aware manner, HILL's structure encoder can create a representation that captures the critical information necessary for the downstream classification task.

The paper achieves this minimization by using two algorithms. Algorithm 1 outlines the overall greedy approach to construct the coding tree, aiming to reach a specified height 'K.' Algorithm 2, invoked repeatedly by Algorithm 1, focuses on creating sub-coding trees of height 2. These sub-trees are then integrated strategically into the main coding tree.

The process, in other words, can be expressed as below (Algo 1 and Algo 2 will be explained in details in the rest of this essay):

- Algorithm 1 starts by creating a simple coding tree with height 1, where all nodes in the original label hierarchy are treated as leaf nodes directly connected to a new root node.
- Sub-Coding Tree Construction (Algorithm 2): This algorithm constructs a sub-coding tree with a height of 2 for a designated non-leaf node 'v' (which acts as the root of the sub-tree). The algorithm goes through two main stages:
    - Compression: It iteratively merges pairs of child nodes under the root 'v' using the 'T.compress(.,.)' operation. This operation prioritizes merging node pairs that lead to the most significant reduction in structural entropy. This continues until a full-height binary tree is formed (which might have a height greater than 2).
    - Condensation: To ensure the sub-coding tree adheres to the height constraint (height 2), the algorithm condenses the tree. It selectively removes non-leaf nodes using the 'T.delete(.)' operation, choosing nodes whose removal causes the least increase in structural entropy.
    - Alignment: Since removing nodes might leave leaf nodes at different depths, the algorithm uses the 'T.align(.)' operation to insert intermediate nodes and ensure all leaf nodes reach the same depth (height 'K').
- Integration into Main Tree (Algorithm 1) The sub-coding tree (T), constructed by Algorithm 2, is then merged back into the main coding tree (T) at the position of the original node 'v'. Algorithm 1 strategically selects 'v' from a set of candidate nodes to minimize the overall structural entropy after merging. This process of invoking Algorithm 2 and integrating the sub-tree is repeated until the main coding tree reaches the target height 'K'.

The paper emphasizes that this minimization process - constructing the optimal coding tree - is crucial for extracting and leveraging the essential structural information inherent in the label hierarchy, within the Hierarchical Text Classification (HTC) task. By minimizing structural entropy, HILL aims to construct a representation of the label hierarchy that is both efficient (using minimal information) and effective (preserving the essential relationships between labels).

3.4 Contrastive Learning Module

3.4.1 Positive Pairs and Contrastive Loss

The model generates positive pairs by creating two distinct views of the same input document: one from a semantic perspective (text encoder) and the other from a syntactic perspective (structure encoder).

The text encoder, typically a BERT family model, processes the input document and generates a document embedding (h_D) representing the semantic information of the text. The structure encoder focuses on the syntactic information encoded within the label hierarchy. It constructs an optimal coding tree from the label hierarchy by minimizing structural entropy. This coding tree guides a hierarchical representation learning process, ultimately producing a structural embedding (h_T) that encapsulates the syntactic information of the label hierarchy in relation to the input document.

$$h = W_{\{D\}}^{\{2\}} ReLU\left(W_{\{D\}}^{\{1\}} \cdot h_{\{D\}} + b_{\{D\}}\right)$$

Equation 2 Document embedding

$$\hat{h} = W_{\{T\}}^{\{2\}} ReLU(W_{\{T\}}^{\{1\}} \cdot h_{\{T\}} + b_{\{T\}})$$

Equation 3 Structural embedding

To ensure both embeddings reside in a comparable space, HILL projects the document embedding (h_D) and the structural embedding (h_T) into a shared embedding space using separate projectors. This results in projected vectors h and $\hat{h}$, respectively.

The projected vectors h and $\hat{h}$ derived from the same input document constitute the positive pair in HILL's contrastive learning framework.

3.4.2 Information Lossless Contrastive Learning for HTC

Information lossless learning, in the context of hierarchical text classification (HTC), centers on preserving the essential information required for downstream prediction, particularly the mutual information between the input (document and label hierarchy) and the target (predicted labels). It doesn't aim to retain all input data but rather focuses on the minimal yet sufficient information for effective classification.

The mutual information represents between inputs and targets in HTC as $I((GL \circ D); Y)$, where:

- GL denotes the random variable for the label hierarchy.
- D represents the random variable for the input document.
- Y signifies the random variable for the target labels.
- ∘ indicates any combination of GL and D as input.

The HILL paper defines an optimal function (F^*), which represents an ideal fusion of text and structure encoders. This function maximizes the mutual information between the input variables ($G_L \circ D$) and the encoded representation.

Being deterministic, F^* ensures that the encoded embedding $F * (G_L \circ D)$ remains consistent for a given $(G_L \circ D)$. Consequently, the mutual information between this embedding and the target variable Y is equivalent to the mutual information between the original input variables and Y:

$$I(F * (G_L \circ D); Y) = I((G_L \circ D); Y).$$

Equation 4

The sources argue that directly encoding the document (D) and the coding tree (TL) of the label hierarchy, as in HILL, preserves more information compared to methods relying on data augmentation. This argument stems from the data processing inequality, which states that augmenting data inevitably leads to some information loss. Therefore, the mutual information between the encoded representation (TL ∘ D) and the target variable (Y) is theoretically greater than or equal to the mutual information between the augmented representation $\theta(G_L, D))$ and Y:

$$I((T_L \circ D); Y) \geq I(\theta(G_L, D); Y)$$

Equation 5

In short, information lossless learning, strives to encode both the document and the label hierarchy in a manner that maximizes mutual information with the target labels, theoretically outperforming methods that rely on potentially information-degrading data augmentation techniques.

3.4.3 Supervised Contrastive Learning

HILL utilizes a combination of Binary Cross-Entropy Loss for multi-label classification and NT-Xent loss for contrastive learning.

3.4.3.1 Binary Cross-Entropy Loss

This loss function is standard for multi-label classification tasks where each label is treated as an independent binary classification problem.

$$L_C = -\frac{1}{|\mathbb{Y}|}\sum_j y_i \log(p_j) \quad + (1 - y_{\{j\}}) log(1 - p_{\{j\}})$$

Equation 6 Binary Cross Entropy Loss

where:

- y_j represents the ground truth (0 or 1) for the j-th label.
- p_j represents the predicted probability (between 0 and 1) for the j-th label.

3.4.3.2 NT-Xent Loss

This contrastive loss function encourages the text encoder and structure encoder to produce similar representations for the same document (positive pair). The paper detail Lclr in Equation 8:

$$L_{\{clr\}} = -log \frac{\left\{e^{\varepsilon\left(h_i,\widehat{h_\imath}\right)/T}\right\}}{\sum_{\{j=1,j\neq i\}}^{\{|B|\}} e^{\left\{\left\{\varepsilon\left(h_j,\widehat{\{h\}_\jmath}\right)\right\}/\,\{T\}\right\}}}$$

Equation 7 Contrastive loss function

where:

- h_j and $\widehat{h_j}$ represent the projected representations of the positive pair (document embedding and structural embedding) for the i-th document in a batch.
- $|B|$ is the batch size.
- τ is a temperature parameter.
- $\varepsilon(h, \hat{h})$ calculates the cosine similarity between two vectors.

3.4.3.3 Final Loss Function (L)

HILL combines LC and Lclr using a weighted sum, as shown in Equation 16:

$$L = L_C + \lambda_{clr} \cdot L_{clr}$$

Equation 8 Loss function for HILL

where:

- λ_{clr} controls the contribution of the contrastive loss.

HILL employs Binary Cross-Entropy Loss to address the multi-label classification task and NT-Xent loss to facilitate contrastive learning between its text and structure encoders. The final loss function blends these two components to optimize the model's performance.

CHAPTER 4. EXPERIMENTS AND RESULTS

4.1 Experiment Setup

4.1.1 Datasets

The origin HILL paper conduct experiments with 3 common datasets in hierachical text classification, WOS [10], RCV1-v2 [19], and NYTimes [4]. However, we did not get the permission to RCV1-v2 dataset, so in this thesis, we only review and experiment with the other two.

The WOS-46985 [10] dataset is a large-scale collection of academic papers sourced from the Web of Science database. It consists of 46,985 documents categorized into 7 fields, including *Computer Science, Electrical Engineering, Psychology, Mechanical Engineering, Civil Engineering, Medical Science, and Biochemistry*. This dataset is commonly used for hierarchical text classification tasks. WOS has total labels: 141 unique labels, average labels per Abstract is 2.0 labels, and maximum depth of label hierarchy is 2 levels.

The NYTimes Annotated Corpus [1] is a dataset comprising over 1.8 million articles from The New York Times, published between 1987 and 2007. However, in this thesis, so as HILL, only conduct experiment in a subset of **46985 samples** of the dataset. The dataset has 166 unique labels, with an average of 7.6 labels per article. The label hierarchy in the NYTimes dataset has a maximum depth of 8.

The RCV1-v2 (Lewis et. al, 2004) dataset is a widely used benchmark in text classification research, particularly for tasks involving hierarchical text classification. It consists of over 800,000 news articles collected by Reuters from 1996 to 1997. These articles are annotated with a rich set of categories organized into a hierarchical structure, covering topics such as politics, economics, business, and more. The dataset's multi-

[1] The NYTimes Annotated Corpus from https://catalog.ldc.upenn.edu/LDC2008T19

level categorization and large size make it ideal for evaluating models in scenarios requiring both flat and hierarchical classification.

Our models are evaluated using two different metrics, Micro-F1 and Macro-F1. Micro-F1 provides an overall performance score by considering all test instances together, making it more sensitive to the performance on frequently occurring labels. On the other hand, Macro-F1 evaluates each category individually and then averages the results, ensuring that each category has an equal impact on the final score, regardless of its frequency.

Table 2 Summary statistics of the three datasets

Dataset	$\|\mathbb{Y}\|$	$\boldsymbol{Avg(Y)}$	**Depth**	**#Train**	**#Dev**	**#Test**
WOS	141	2.0	2	30070	7518	9397
RCV1-v2	103	3.24	4	20833	2316	781265
NYTimes	166	7.6	8	23345	5834	7292

One notable observation from experiments of HILL's authors is that while contrastive learning, improves performance on WOS, the gains are less pronounced compared to other datasets like RCV1-v2 and NYTimes. Sources attribute this to two main factors: Domain Gap and Limited Hierarchical Information. Firstly, a significant difference exists between the types of text in the WOS dataset (academic abstracts) and the data used to pre-train the BERT language model. This gap can limit the effectiveness of contrastive learning.

And regarding the latter factor, the WOS dataset has a relatively shallow label hierarchy (maximum depth of 2) compared to RCV1-v2 (depth of 4) and NYTimes (depth of 8). This shallow hierarchy suggests that WOS might contain less essential structural information for hierarchy-aware models to leverage, leading to smaller gains from contrastive learning.

4.1.2 Evaluation metrics

Micro-F1 and Macro-F1

Both metrics are derived from precision and recall calculations, combining these measures to provide a balanced view of a model's performance.

Micro-F1 calculates the F1 score by aggregating the contributions of all classes to compute the average metric. This approach treats each instance equally, effectively giving more weight to larger classes. It's particularly useful when dealing with imbalanced datasets where some classes may have significantly more instances than others. Micro-F1 provides a good indication of overall accuracy across all instances, regardless of their class distribution [20].

On the other hand, Macro-F1 computes the F1 score independently for each class and then takes the unweight mean of these scores. This method treats all classes equally, regardless of their size or frequency in the dataset. Macro-F1 is valuable when you want to understand the model's performance across all classes, giving equal importance to minority classes that might be overlooked in a micro-averaged approach.

After implementing the base model for HILL and collect enough datasets, we rented a server with GPU from vast.ai for 2 weeks to run the experiment. The configuration of the server is

- CPU: 4 core
- RAM: 32 GB DDR4
- Storage: 100GB
- GPU: RTX 3060 12 GB VRAM

Averagely, each run of 25 epochs cost 18 to 24 hours to converge.

4.2 Experiment models

The model's parameters are updated using the Adam optimizer, with key parameters and settings defined as follows. The learning rate for the Text Encoder (BERT) is set to $3 \times 10^{\{-5\}}$, while the learning rate for the Structure Encoder varies based on the dataset: $1 \times 10^{\{-3\}}$ for WOS, and $1 \times 10^{\{-4\}}$ and NYTimes. The batch size is set to 24 for WOS and NYTimes. All hidden sizes (d_B, d_V, d_T) are uniformly set to 768 across all datasets. The height of the coding tree (K) is determined empirically for each dataset, with values of 3 for WOS, and 3 for NYTimes. Notably, the optimal K does not directly correspond to the label hierarchy's depth but appears to relate to the number of labels in the dataset. The contrastive loss weight ($\lambda_{\{clr\}}$) is tuned for each dataset, set at 0.001 for WOS, 0.1 for RCV1, and 0.3 for NYTimes. The function $(\eta\ \cdot)$ is implemented as a summation for WOS and NYTimes, combining information from different levels of the coding tree in the structure encoder.

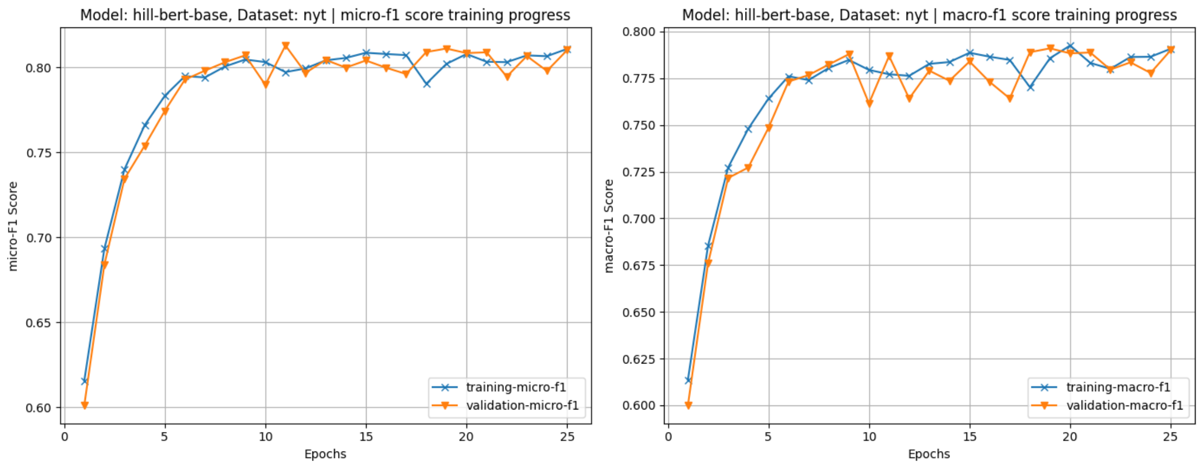
Table 3 Hyperparamters setting

Dataset	**WOS**	**NYTimes**	**RCV1-v2**
Optimizer	Adam	Adam	Adam
Learning rate	3e-5	3e-5	3e-5
Hidden size	768	768	768
K layers of MLP	2	2	2
Batch size	24	24	16
Structural encoder's learning rate	1e-3	1e-4	1e-4
Contrastive loss weight λ_{clr}	0.001	0.1	0.3
# parameters	7.37M	7.30M	7.34M

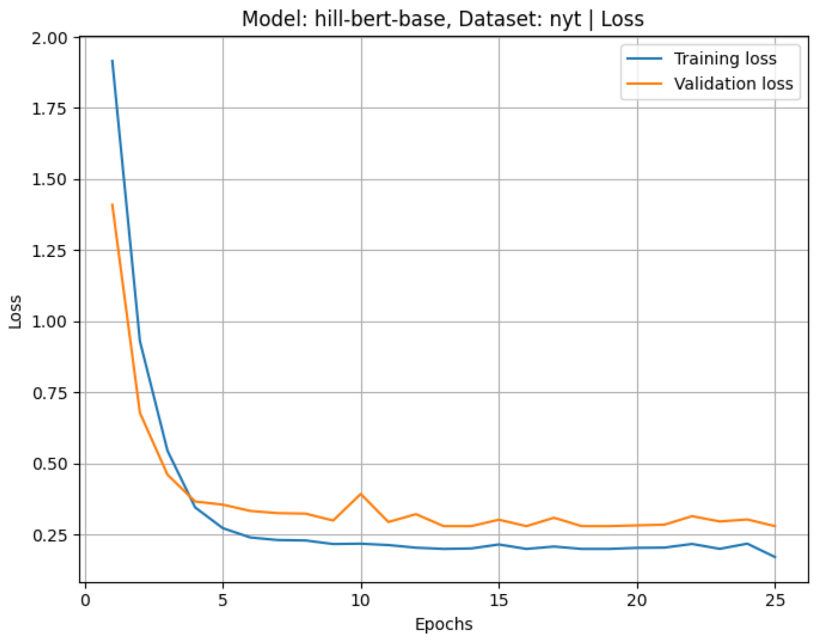
In terms of the training process, the text encoder (BERT) is initialized with pre-trained weights. The model is trained on the respective datasets, where the input documents are fed through both the text and structure encoders. The model learns to minimize the combined loss function, effectively capturing both semantic information from the text and structural information from the label hierarchy during training. The paper emphasizes that HILL's architecture, particularly the structure encoder, is designed for efficiency, utilizing fewer parameters and training faster than comparable models like HGCLR, thereby providing a time- and memory-efficient approach.

The next section is our model training progress.

BERT-base-uncased, Dataset: NYTimes



Micro and Macro-F1 on train and validation set



Loss progress on train and validation set

BERT-base-uncased, Dataset: WOS

Reportedly, HILL paper got (87.28, 81.77) on WOS dataset, and (80.47, 69.96) on NYTimes set, for micro-F1 and macro-F1 respectively

Table 4.2. Performance across variants

	WOS		**NYT**		**RCV-v2**	
Model\Metrics	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**
Contrastive learning model						
HILL	87.28	**81.77**	**80.47**	**69.96**	**87.31**	**70.12**
HiAGM	86.20	80.53	-	-	84.73	64.11
HGCLR	87.11	81.20	78.86	80.47	86.49	68.31
Our model						
BERT	**87.35**	81.47	80.39	69.86	86.53	69.70
RoBERTa	87.32	81.57	79.05	67.41	-	-
DistilledBERT	85.60	80.44	74.47	66.61	81.12	64.43
MPNet	-	-	60.55	17.31	-	-

4.3 Experimental Results

a. Coding tree K height

The height of the coding tree (K) in the HILL model significantly influences its performance on hierarchical text classification tasks.

As the height (K) increases, the performance of HILL generally degrades. This degradation is likely due to the increasing depth of the coding tree, which can lead to more explosive gradients during training, making optimization more difficult. It’s worth noting that the ideal height (K) doesn't seem to directly correlate with the height of the label hierarchy in the dataset. For instance:

- WOS dataset: label hierarchy height = 2, optimal K = 3
- RCV1-v2 dataset: label hierarchy height = 4, optimal K = 2
- NYTimes dataset: label hierarchy height = 8, optimal K = 3

Instead, the optimal K appears to have a positive correlation with the number of labels in the dataset (|Y|):

- WOS: |Y| = 141
- RCV1-v2: |Y| = 103
- NYTimes: |Y| = 166

The paper provides visualizations of HILL's performance on the WOS, RCV1-v2, and NYTimes datasets with varying coding tree heights. The visualizations clearly show performance drops as K goes beyond the optimal value for each dataset.

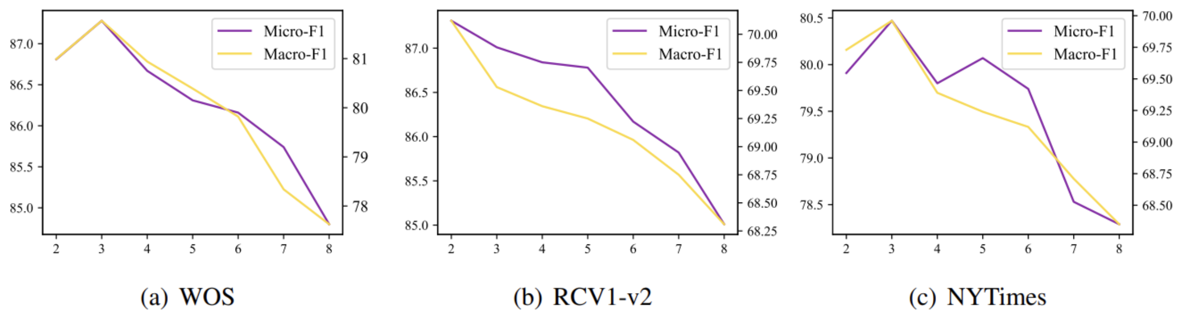


Figure 4.2. Drop in performance HILL with different height K of the coding tree

b. Time-and-Memory-Saving Contrastive Learning

This section highlights the time-and-memory-saving advantages of the HILL model for hierarchical text classification, particularly when compared to the HGCLR model. This efficiency stems from HILL's comparatively smaller structure encoder.

First of all, HILL's structure encoder relies on a hierarchical representation learning module composed of K multi-layer perceptions. In contrast, HGCLR utilizes the more complex Graphormer, which is built upon multi-head attention mechanisms. Multi-head attention, while powerful, typically demands more computational resources than multi-layer perceptions.

Secondly, the authors directly compare the number of trainable parameters in HILL and HGCLR (Figure 4). This analysis reveals that HILL uses significantly fewer parameters, averaging 7.34 million compared to HGCLR's 19.04 million.

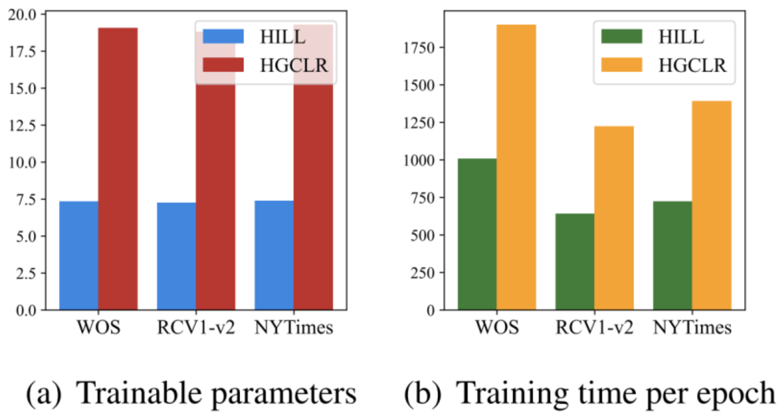


Figure 4.3. HILL parameters and training time compared to HGCLR (in courtesy of HILL paper)

Lastly, HILL also evaluate the training speed of both models on three datasets: WOS, RCV1-v2, and NYTimes. HILL consistently exhibits faster training times. For instance, HILL's average training time is 789.2 seconds, approximately half of HGCLR's 1504.7 seconds.

CHAPTER 5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this report, we experimented the new model HILL approach of information lossless contrastive learning successfully and confirmed the original findings, demonstrating reliability of the HILL framework for hierarchical text classification. Our efforts validated the effectiveness of the HILL model as presented in the original study.

Additionally, we extended our work by testing 3 language models, RoBERTa and DistilBERT and MPNet. The results were consistent with our predictions, showing that these models perform similarly to HILL in the hierarchical classification tasks, through each with its own strengths. RoBERTa provided better accuracy due to its more robust training approach, whilst DistilBERT offered a balance between performance and efficiency, making it suitable for practical applications.

Generally, this project has provided insights into the performance and capability of different models in hierarchical text classification. It has broadened our understanding of model efficiency and accuracy, and noted the importance of selecting the right model based on specific requirements. The valuable lessons we learned from this project will inform future research and development in the field of hierarchical text classification.

5.2 Future work

Future work for this thesis can be focused on 3 areas to enhance the understanding and performance of the HILL model. First, a cross-domain evaluation of HILL could be conducted to test its robustness and adaptability across datasets from various domains, for example, medical records, legal documents, news articles, or social media posts. Currently, most of HTC works only evaluated on scientific article or professional writings, which were not very insightful to the other domain. This will provide insights into how well HILL generalizes across different types of data and reveal room for potential improvement.

Additionally, a fine-grained analysis of hierarchical levels is essential to understand how HILL performs at different levels of the hierarchy. This involves breaking down the hierarchical structures in test datasets and evaluating the model’s accuracy and efficiency at each level, from root to leaf nodes. Through this analysis we can see strengths and weaknesses of HILL in handling hierarchical complexities, guiding further refinements.

Finally, we may conduct comparative analysis with more models to benchmark HILL against the latest state-of-the-art text embedding models, such as E5 [21] or Multilingual E5 Text Embeddings [22], other advanced transformers. This comparison will involve implementing these new models within the HILL framework or alongside it, testing their performance on the same datasets, and analyzing the results to determine how HILL measures up against these SOTA approaches.

REFERENCES

[1] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys (CSUR), Volume 34, Issue 1,* pp. 1-47, 2002.

[2] Celine Vens, Jan Struyf, Leander Schietgat, Sašo Džeroski & Hendrik Blockeel , "Decision trees for hierarchical multi-label classification," *Mach Learn,* p. 185–214, 2008.

[3] Kai Li, Jason Rollins & Erjia Yan, "Web of Science use in published research and review papers 1997–2017: a selective, dynamic, cross-domain, content-based analysis," *Scientometrics,* pp. 1-20, 2018.

[4] Sandhaus, "The New York Times Annotated Corpus," 2008. [Online]. Available: https://catalog.ldc.upenn.edu/LDC2008T19.

[5] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov, RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019.

[6] Sanh et al., "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019,* 2019.

[7] Song et al., "MPNet: Masked and Permuted Pre-training for Language Understanding," in *NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.

[8] L. Cai, "Hierarchical document categorization with support vector machines," *ACM International Conference on Information and knowledge management,* pp. 78-87, 2004.

[9] Andrew McCallum, Ronald Rosenfeld, Tom Mitchell, Andrew. Y. Ng, "Improving Text Classification by Shrinkage in a Hierarchy of Classes," 1998.

[10] K. Kowsari, "HDLTex: Hierarchical Deep Learning for Text Classification," in *Advances in Neural Information Processing Systems 32*, 2017.

[11] J. Zhou, "Hierarchy-Aware Global Model for Hierarchical Text Classification," *Aclanthology,* 2020.

[12] Zihan Wang, Peiyi Wang, Lianzhe Huang, Xin Sun, Houfeng Wang, "Incorporating Hierarchy into Text Encoder: a Contrastive Learning Approach for Hierarchical Text Classification," Dublin, Ireland, 2022.

[13] He Zhu, Junran Wu, Ruomei Liu, Yue Hou, Ze Yuan, Shangzhe Li, Yicheng Pan, Ke Xu, HILL: Hierarchy-aware Information Lossless Contrastive Learning for Hierarchical Text Classification, 2024.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, "Efficient Estimation of Word Representations in Vector Space," *ICLR,* 2013.

[15] J. Pennington, R. Socher and C. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014.

[16] A. Zangari, "Hierarchical Text Classification and Its Foundations: A Review of Current Research," *Advanced Natural Language Processing Technology and Applications,* 2024.

[17] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer, "Deep Contextualized Word Representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana, 2018.

[18] Y. Liu, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," *ICLR 2020,* 2020.

[19] Lewis et. al., "RCV1: A New Benchmark Collection," *Journal of Machine Learning Research ,* p. 361–397, 2004.

[20] S. B. Juri Opitz, "Macro F1 and Micro F1," Arxiv preprint, 2019.

[21] L. Wang, "Text Embeddings by Weakly-Supervised Contrastive Pre-training," *Arxiv preprint,* December 2022.

[22] N. Y. X. H. L. Y. R. M. F. W. Liang Wang, "Multilingual E5 Text Embeddings: A Technical Report," *Arxiv Preprint,* 2024.

[23] Fatos Torba, Christophe Gravier, Charlotte Laclau, Abderrhammen, "A Study on Hierarchical Text Classification as a Seq2seq," *46th European Conference on Information,* March 2024.