ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230249

Hierarchical Text Classification Using Contrastive Learning Informed Path Guided Hierarchy

Neeraj Agrawal[a;*], Saurabh Kumar[a], Priyanka Bhatt[a] and Tanishka Agarwal[a]

[a]Walmart Global Tech, Bangalore, India
ORCiD ID: Neeraj Agrawal https://orcid.org/0000-0003-1496-6618,
Saurabh Kumar https://orcid.org/0009-0006-5396-2723, Priyanka Bhatt https://orcid.org/0009-0009-6431-0490,
Tanishka Agarwal https://orcid.org/0009-0007-8583-1653

Abstract. Hierarchical Text Classification (HTC) has recently gained traction given the ability to handle complex label hierarchy. This has found applications in domains like E- commerce, customer care and medicine industry among other real-world applications. Existing HTC models either encode label hierarchy separately and mix it with text encoding or guide the label hierarchy structure in the text encoder. Both approaches capture different characteristics of label hierarchy and are complementary to each other. In this paper, we propose a Hierarchical Text Classification using Contrastive Learning Informed Path guided hierarchy (HTC-CLIP), which learns hierarchy-aware text representation and text informed path guided hierarchy representation using contrastive learning. During the training of HTC-CLIP, we learn two different sets of class probabilities distributions and during inference, we use the pooled output of both probabilities for each class to get the best of both representations. Our results show that the two previous approaches can be effectively combined into one architecture to achieve improved performance. Tests on two public benchmark datasets showed an improvement of 0.99 - 2.37% in Macro F1 score using HTC-CLIP over the existing state-of-the-art models.

1 Introduction

In the literature, the problem of categorizing text into a set of labels that are organized in a structured hierarchy is defined as Hierarchical Text Classification (HTC) [18, 19, 13]. HTC is a particular multi-label text classification (MLC) problem, where the classification result corresponds to one or more nodes of a taxonomic hierarchy. The class dependency in HTC is usually assumed to follow a hierarchical structure represented by a tree or a directed acyclic graph as shown in Figure 1.

HTC approaches can be broadly classified as local approaches and global approaches. Algorithms that perform local learning attempt to discover the patterns that are present in regions of the class hierarchy, later combining the predictions to provide the final classification. The local approaches [22, 17, 1] exploit the parent and child hierarchy to overcome data imbalance in child node. Local approaches generally suffer from the error-propagation problem and are often computationally expensive. Global approaches for HTC, on the other hand, usually consist of a single classifier capable of associating objects with their corresponding classes in the hierarchy at once [7, 24]. Global approaches are usually less likely to capture local information from the hierarchy and eventually suffer from underfitting. [21] proposed a hybrid approach by combining local and global loss. Many researchers tried to encode text and label hierarchy separately and aggregate the two representations before being classified by a mixed feature [28, 5] (as shown in Figure 2b). Recently [20] proposed a state-of-the-art model by encoding label hierarchy in a text encoder using contrastive learning during training time (as shown in Figure 2c). This model doesn't have a hierarchical encoder explicitly during inference time. The explicit label hierarchical encoder might learn information complement to the text encoder.

Therefore, in this paper, we introduce path-guided hierarchy using chained architecture and embedded hierarchy information in text using contrastive learning as shown in Figure 2d. We propose Hierarchical Text Classification using Contrastive Learning Informed Path guided hierarchy (HTC-CLIP), which exploits embedded hierarchy in the text along with path-guided hierarchy during inference time. We learn two sets of classifiers: one with linear layer and another with path-guided hierarchy, on top of hierarchy encoded text encoder. Graphormer [26] is used to embed hierarchy in BERT based text encoder [20]. Linear and hierarchy classifiers learn different probability distributions. Therefore, during inference we use maximum pooling to get the final classifier probability.

The main contributions of our paper are summarised as follows:

- We propose an architecture that can encode hierarchy information in text and explicitly learns path-guided hierarchy with contrastive learning.
- We introduce two sets of classifiers during training and pooling operation during inference to imitate ensemble behaviour.
- Experiments demonstrate that the proposed model achieves an improvement of 0.99-2.37% in Macro F1 on two public datasets, WOS and NYT.

2 Related Work

Work on HTC can broadly be categorized into local and global approaches [20]. In local approaches, a classifier can be built per node, per parent, or per level. In global approaches only one classifier is built for the entire graph. [1] uses a local approach to build a classifier per label and transfers parameters of the parent model to the child model. [21, 22] use a hybrid approach to build a common classifier

* Corresponding Author. Email: neeraj.agrawal@walmart.com

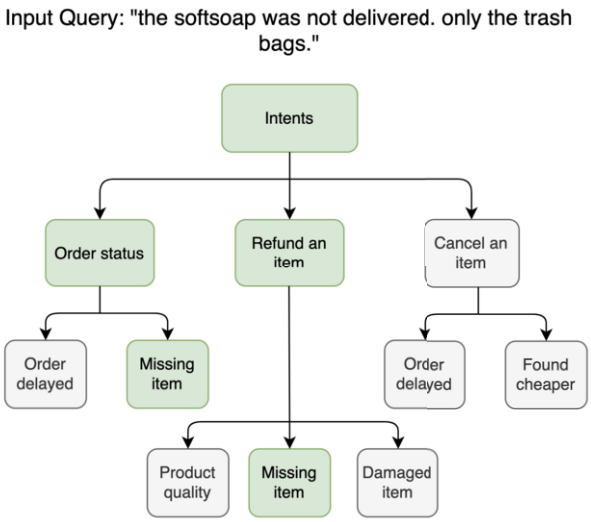

Figure 1. Input query is tagged with "missing items" from two different paths "order status" or "refund an item."

that optimizes global and local loss. [14] decomposes the hierarchy into subgraphs and conducts Text-GCN on n-gram tokens, whereas [17] applies CNN to utilize the data in the upper levels to contribute to categorization in the lower levels.

In early global approaches, HTC is solved by reducing the problem to a flat multi-layer classification problem [9]. Later approaches try to improve the MLC by embedding hierarchy information. [7] introduced the label structure by recursive regularisation. Some works use deep learning based architecture to learn hierarchical structure of labels, by employing sequence to sequence network [25], reinforcement learning [12], meta-learning [24] and capsule network [15]. These models mainly focus on improving decoders based on the constraint of hierarchical paths. Later literature focused on encoding the hierarchy information using structure encoders. [28] proposed GCN and LSTM-based hierarchical encoder and methods to fuse those embeddings with text encoder outputs. [27] extracts text features according to different hierarchy levels. [5] introduces information maximization to constrain label representation learning. [2] views the problem as semantic matching and tries BERT as a text encoder. All these works tried to model structure encoder and text encoder separately and later create mixed representations for classification. Recently [20] has shown that infusing hierarchy information in text encoder using Graphomer can further improve the state-of-the-art on three HTC datasets.

3 Problem Definition

Hierarchical Text Classification (HTC) is the task of classifying input text $x = \{x_1, x_2, ..., x_n\}$ to subset y of label set Y, where n is the number of tokens in x. Size of label set Y is $|C|$. The label hierarchy mainly contains a tree-like structure and a directed acyclic graph (DAG) structure. We formulate label hierarchy as DAG, G = (Y, E), where node set Y is labels and edge set E denotes the relation between parent and child node. Since a non-root label of HTC has one and only one parent, the label hierarchy can be converted to a tree-like hierarchy. Each sample x corresponds to a subset y that includes multiple classes. Those corresponding classes belong to either one or more sub-paths in the hierarchy as shown in Figure 1.

4 Methodology

In this section, we will describe the proposed HTC-CLIP in detail. Figure 3 shows the overall architecture of the model.

4.1 Text Encoder

BERT [6] is one of the state-of-the-art encoders for text. We use it as the text encoder similar to [2, 20]. BERT uses WordPiece algorithm to tokenize the input text. Given an input text x, it gets tokenized into sequence of tokens $x_1, x_2, ...x_{n-2}$. BERT adds [CLS] and [SEP], two special tokens indicating the beginning and the end of the sequence. Therefore, the final sequence of tokens of length n is represented as:

$$x = \{[\text{CLS}], x_1, x_2, ..., x_{n-2}, [\text{SEP}]\} \tag{1}$$

BERT encodes input tokens and outputs encodings corresponding to each token. We use encoding corresponding to [CLS] token as input to the path-guided hierarchy encoder and linear classifier.

$$P = \text{BERT}(x) \tag{2}$$

Where, $P \in \mathbb{R}^{n \times d_h}$, d_h is hidden dimension and n is number of tokens. Hidden state corresponding to [CLS] token is $p = P_{[CLS]}$.

4.2 Hierarchy Encoder

[20] proposed a contrastive learning-guided hierarchy in text encoder and [22] suggested a path-guided hierarchy to improve the HTC task. Experiments suggest that probability class distributions learnt by these two methods are complementary to each other. In order to leverage the representation learnings of both HTC approaches, we have proposed a hybrid method capable of **simultaneously learning two sets of classifiers, one using path-guided hierarchy classifier and the other using linear classifier on top of hierarchy encoded text using contrastive learning** as shown in Figure 3.

4.2.1 Path Guided Hierarchy Classifier

Path-encoded hierarchy helps in forcing the hierarchy structure between levels. [22, 21] created the hierarchy on top of text encoder. We have proposed the configuration of linear layers with ReLU activation to learn better hierarchy relations between children and parents.

Formally, let $p \in \mathbb{R}^{d_h}$ be the pooled output (hidden state corresponding to [CLS] token) from BERT text encoder as described in section 4.1, C_h be the set of classes of the h^{th} hierarchical level, $|H|$ the total number of hierarchical levels, and $|C|$ the total number of classes. Let A_1^P denote the activations for the first level using BERT pooled output, given by:

$$A_1^P = \phi(W_1^P p + b_1^P) \tag{3}$$

where $W_1^P \in \mathbb{R}^{C_1 \times d_h}$ is a weight matrix and $b_1^P \in \mathbb{R}^{C_1 \times 1}$ is the bias vector, which are the parameters for learning level 1 classifier based on text and hierarchy encoded pooled output, and ϕ is a non-linear activation function, in our work we have used ReLU. Similarly, activation A_h^P for level h using pooled output is given by:

$$A_h^P = \phi(W_h^P p + b_h^P) \tag{4}$$

where $W_h^P \in \mathbb{R}^{C_h \times d_h}$ is a weight matrix and $b_h^P \in \mathbb{R}^{C_h \times 1}$ is the bias vector.

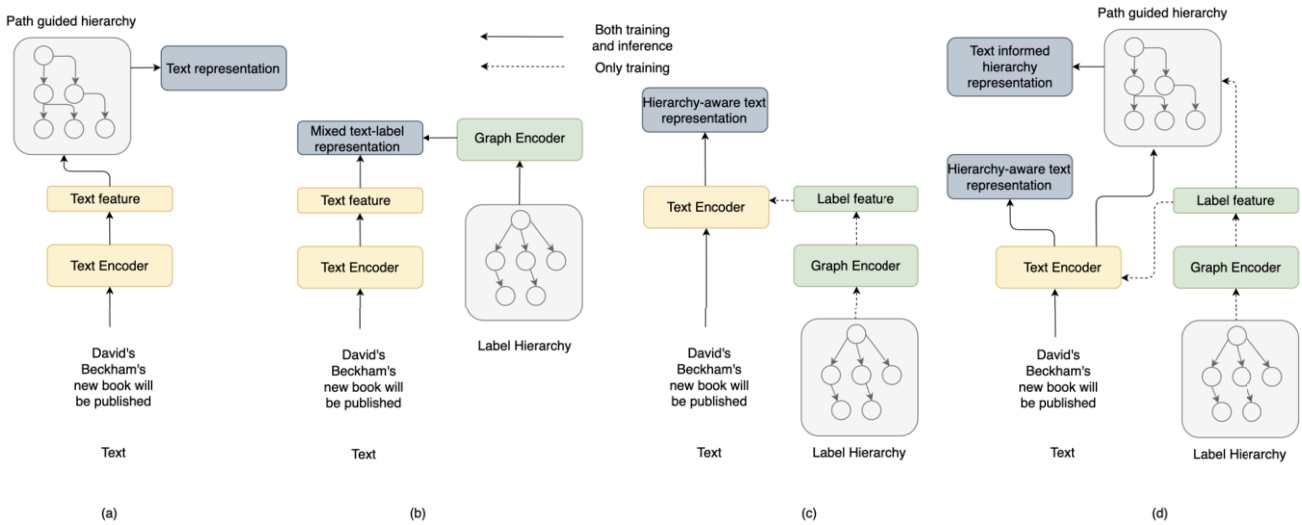

Figure 2. Different ways of introducing hierarchy information. (a) Previous work of modelling path-guided hierarchy, on top of text encoder [22]. (b) Previous work of modelling text and labels separately and finding a mixed representation [28, 5]. (c) Previous work of incorporating hierarchy information into text encoder for a hierarchy-aware text representation [20]. (d) Our work where the model learns two classifiers, one with text-encoded hierarchy and another with path-guided hierarchy, and pooled output from both classifiers is used during inference.

We have introduced a linear layer, that maps activation from level $h-1$ to k hidden neurons, and then a linear layer to map k activation to C_h classes at level h. These hidden layers help in learning a better representation of the parent and child nodes relationship. Let A_h^{h-1} denote activation for the h level using $h-1$ level activation.

$$A_k^{h-1} = \phi(W_k^{h-1} A_{h-1}^P + b_k^{h-1}) \tag{5}$$

$$A_h^{h-1} = \phi(W_h^k A_k^{h-1} + b_h^k) \tag{6}$$

where $W_k^{h-1} \in \mathbb{R}^{k \times C_{h-1}}$, $W_h^k \in \mathbb{R}^{C_h \times k}$ are weight matrices and $b_k^{h-1} \in \mathbb{R}^{k \times 1}$, $b_h^k \in \mathbb{R}^{C_h \times 1}$ are the bias vectors.

Let A_h be the final activation for level h, which is sum of activation based on pooled output, A_h^p and activation based on previous level A_h^{h-1}. For level 1, A_1^0 is a vector of all ones.

$$A_h = A_h^P \oplus A_h^{h-1} \tag{7}$$

Following previous work [20], we flatten the hierarchy for multi-label classification therefore the output of each classifier is equal to the total number of classes $|C|$. Hence, all activations from each level are concatenated and passed through sigmoidal activation to get class probabilities, P_c of path-guided hierarchy classifier.

$$P_c = \sigma(A_1 \odot A_2 \odot \cdots \odot A_{|H|}) \tag{8}$$

4.2.2 *Positive Sample Generation for Contrastive Learning*

The goal for the positive sample generation is to keep a fraction of tokens while retaining the labels. Given a token sequence as Equation 1, the token embedding of BERT is defined as:

$$e_1, e_2, ..., e_n = \text{BERT\_emb}(x) \tag{9}$$

The scale-dot attention weight between token embedding and label feature is first calculated to determine the importance of a token on a label,

$$q_i = e_i W_Q, k_j = l_j W_k, A_{ij} = \frac{q_i k_j^T}{\sqrt{d_h}} \tag{10}$$

The query and key are token embeddings and label features respectively, and $W_Q \in R^{d_h \times d_h}$ and $W_K \in R^{d_h \times d_h}$ are two weight matrices. Thus, for a certain x_i, its probability of belonging to label y_j can be normalized by a Softmax function.

Next, given a label y_j , we can sample key tokens from that distribution and form a positive sample $\hat{x}$. To make the sampling differentiable, we replace the Softmax function with Gumbel-Softmax [8] to simulate the sampling operation:

$$P_{ij} = gumbel\_softmax(A_{i1}, A_{i2}, ..., A_{ik})_j \tag{11}$$

Notice that a token can impact more than one label, so we do not discretize the probability as one-hot vectors in this step. Instead, we keep tokens for positive examples if their probabilities of being sampled exceed a certain threshold γ, which can also control the fraction of tokens to be retrained. For multi-label classification, we simply add the probabilities of all ground-truth labels and obtain the probability of a token x_i regarding its ground-truth label set y as:

$$P_i = \sum_{j \in y} P_{ij} \tag{12}$$

Finally, the positive sample $\hat{x}$ is constructed as:

$$\hat{x} = \{x_i \; if \; P_i > \gamma \; else \; \mathbf{0}\} \tag{13}$$

where $\mathbf{0}$ is a special token that has an embedding of all zeros so that key tokens can keep their positions. The select operation is not differentiable, so we implement it differently to make sure the whole model can be trained end-to-end.

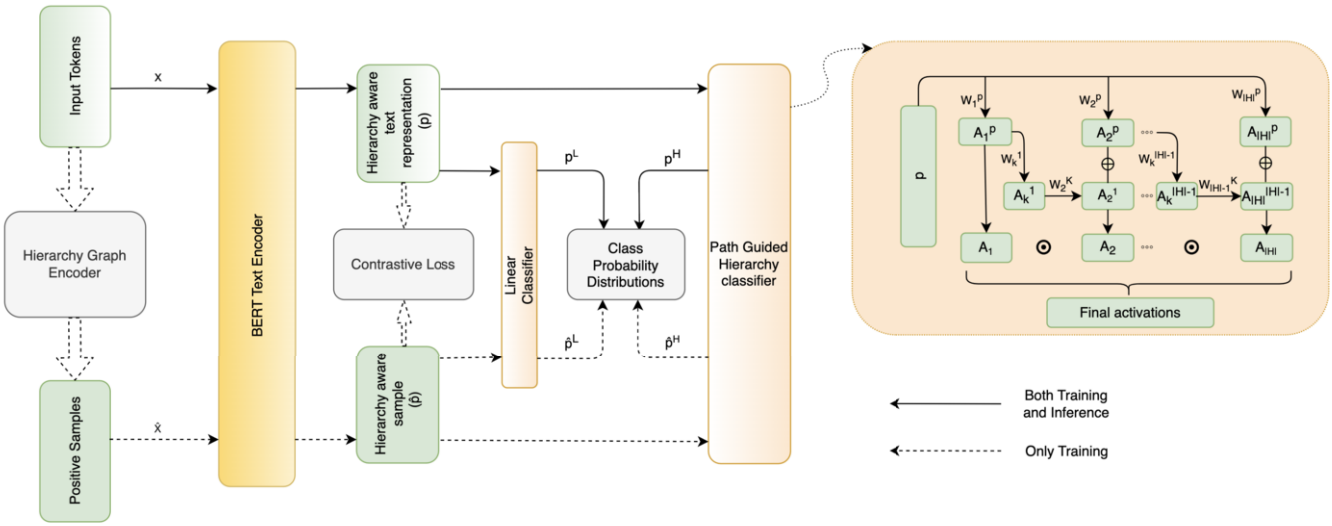

Figure 3. HTC-CLIP architecture. It contains a BERT based encoder, which generates a hierarchy-aware text representation and a positive sequence output representation. Both representations are passed through two classifiers (linear classifier and path-guided hierarchy classifier) to minimize binary cross-entropy loss, which helps in generating text-aware hierarchy representations.

4.2.3 *Contrastive Learning for Text Encoder*

Contrastive loss helps in bringing token sequence and their positive counterpart representation closer, and the examples which are not from the same pair will be moved farther away. Encoding hierarchy information in text encoder using contrastive loss is proven to improve the result for HTC [20], we have used the same implementation for contrastive learning for text encoder. First, positive samples ($\hat{x}$) are generated for contrastive loss as discussed in 4.2.2. The positive sample is fed to the same BERT as the original one in section 4.1.

$$\hat{P} = \text{BERT}(\hat{x}) \tag{14}$$

We get a sequence representation $\hat{p}$ with the first token corresponding to [CLS] before being classified.

With a batch of N hidden states of positive pairs $(p_i, \hat{p}_i)$, with nonlinear activation ReLU:

$$\begin{aligned} c_i &= W_2\text{ReLU}(W_1 p_i) \\ \hat{c}_i &= W_2\text{ReLU}(W_1 \hat{p}_i) \end{aligned} \tag{15}$$

where $W_1 \in \mathbb{R}^{d_h \times d_h}$, $W_2 \in \mathbb{R}^{d_h \times d_h}$.

For a batch of size N, we generate N positive examples so there will be total 2N pairs. For a given utterance there will be 2 positive pairs and remaining 2(N-1) pairs will be negative examples. Thus, in a batch of size N with 2N examples $\mathbf{Z} = \{z \in \{c_i\} \cup \{\hat{c}_i\}\}$, we compute the NT-Xent loss [4] for z_m as:

$$L_m^{con} = -log \frac{exp(sim(z_m, \mu(z_m)/\tau)}{\sum_{i=1, i\neq m}^{2N} exp(sim(z_m, z_i)/\tau)} \tag{16}$$

where sim is the cosine similarity function as:

$$sim(u, v) = u \cdot v / \|u\| \|v\| \tag{17}$$

and μ is a matching function as:

$$\mu(z_m) = \begin{cases} c_i, & \text{if } z_m = \hat{c}_i \\ \hat{c}_i, & \text{if} z_m = c_i \end{cases} \tag{18}$$

τ is a temperature hyperparameter.

The total contrastive loss is the mean loss of all examples:

$$L_{con} = \frac{1}{2N} \sum_{m=1}^{2N} L_m^{con} \tag{19}$$

4.3 *Contrastive Learning Informed Classifiers*

To improve the representation of linear classifier and path-guided hierarchical classifier, the constructed positive sample BERT representation $\hat{p}_i$ is passed through each classifier separately.

The probability of text representation $\hat{p}_i$ on label j is:

$$\hat{p}_{ij}^L = \sigma(\text{Linear}(\hat{p}_i))_j \tag{20}$$

$$= \sigma(W_L \cdot \hat{p}_i + b_L)_j \tag{21}$$

$$\hat{p}_{ij}^H = \sigma(\text{Hierarchical}(\hat{p}_i))_j \tag{22}$$

Where $W_L \in \mathbb{R}^{|C| \times d_h}$, $b_L \in \mathbb{R}^{|C| \times 1}$. σ is sigmoid function. The hierarchical classifier is explained in section 4.2.1.

Positive sample representation, $\hat{p}_i$ learned using contrastive loss tunes classifiers' weights during training using binary cross entropy loss as explained in the next section. The path-guided classifier's weights tuning helps in encoding text representation in a path-guided network.

4.4 *Classification and Objective Function*

Similar to positive constructed sequence output $(\hat{p}_i)$, the hidden feature (p_i) from the text encoder is fed into a linear and hierarchical classifier. Outputs from classifiers are fed into sigmoid (σ) activation to calculate class probability distribution.

Dataset	|L|	Depth	Avg(|L_i|)	Train	Val	Test
WOS Dataset	141	2	2.0	30,070	7,518	9,397
NYT Dataset	166	8	7.6	23,345	5,834	7,292

Table 1. Statistics of three datasets for hierarchical multi-label text classification. |L|: Number of target classes. Depth: Maximum level of hierarchy. Avg(|L_i|): Average Number of classes per sample. Train/Val/Test: Size of train/validation/test set.

The probability of text representation p on label j is:

$$p_{ij}^{L} = \sigma(\text{Linear}(p_i))j \tag{23}$$

$$p_{ij}^{H} = \sigma(\text{Hierarchical}(p_i))j \tag{24}$$

For multi-label classification, we use the binary cross-entropy loss function for text i on label j,

$$L_{ij}^{C} = -y_{ij}\log(p_{ij}) - (1-y_{ij})\log(1-p_{ij}) \tag{25}$$

$$L^{C} = \sum_{i=1}^{N}\sum_{j=1}^{k} L_{ij}^{C} \tag{26}$$

where y_{ij} is the ground truth. L^C will be L_L^C, L_H^C for p_{ij} as p_{ij}^L, p_{ij}^H respectively.

The classification loss of the positive sample representation $\hat{L}_L^C$, $\hat{L}_H^C$ can be calculated similarly by $\hat{p}_{ij}^L$, $\hat{p}_{ij}^H$ using Equation 20, 22 and 26.

The final loss function is the combination of classification loss of original text, classification loss of the constructed positive samples, and the contrastive learning loss:

$$L = L_L^C + L_H^C + \hat{L}_L^C + \hat{L}_H^C + \lambda L_{con} \tag{27}$$

where λ is a hyperparameter controlling the weight of contrastive loss. During testing, we use the text encoder and path-guided encoder for classification and the model degenerates to a BERT encoder with two classification heads. The maximum pool output of both the linear classifier and path-guided classifier is used as a class probability. This output is used for assigning classes for a given input text.

5 Experiments

In order to assess the effectiveness of the proposed architecture and compare it against existing models, we have run several experiments.

5.1 Experiment Setup

Datasets and Evaluation Metrics We performed experiments on Web-of-Science (WOS) [10] and NYT [16] datasets for comparison and analysis. WOS dataset includes abstracts of published papers from Web of Science. NYT is an archive of manually categorized newswire stories. For WOS and NYT datasets we have followed the train/val/test distribution of [20] and [28]. Statistics of these datasets are listed in Table 1.

Evaluation Metrics We measure the experimental results by Micro-F1 and Macro-F1. Micro-F1 is calculated from the overall precision and recall of all the instances, while Macro-F1 is equal to the average F1-score of labels.

Implementation Details We use bert-base-uncased from Transformers [23, 20] as the base architecture for text encoder. As [20] suggested, for Graphormer, attention head is set to 8 and feature size d_h to 768. We use 1 GPU of Nvidia A100 for computing. We set the batch size to 56. Similar to [20], we use Adam optimizer with a learning rate of 3x10^{-5}. The threshold γ is set to 0.02 on WOS, 0.005 on NYT dataset. The loss weight λ is set to 0.05 on WOS, 0.3 on NYT dataset. The temperature of the contrastive module is fixed to 1 [20]. The hidden layer size (k) for the path-guided hierarchy is set to 128 for all three datasets. We implemented our model in PyTorch and trained end-to-end, and stopped training if the Macro-F1 does not increase for 6 epochs. We evaluated the test subset with the model having the best Macro-F1 on the validation subset.

Comparison Models We compare the performance of our HTC-CLIP model with a few recent works on HTC as strong baselines. HGCLR [20], HiAGM [28], HTCInfoMax [5] and HiMatch [3]. HiAGM applies soft attention to text features and label features for the mixed feature. HTCInfoMax improves HiAGM by regularizing the label representation with a prior distribution. HiMatch matches text representation with label representation in a joint embedding space and uses joint representation for classification. HGCLR directly embeds the hierarchy into a text encoder. HGCLR is the state-of-the-art before our work. Except for HiMatch and HGCLR, all of the above approaches adopt TextRCNN [11] as text encoder therefore we used [20] implementation of these with BERT for a fair comparison. Results are shown in Table 2.

Weight Parameters Size Table 3 shows the size of weight parameters. We observe that the number of parameters due to the path-guided hierarchy network in our proposed model doesn't increase the weight parameters significantly compared to HGCLR.

5.2 Experiment Results

Table 2 reports the performance of our approach against other methods. On WOS dataset, our proposed HTC-CLIP model shows 2.06% and 2.44% improvement on Micro-F1 and Macro-F1 respectively compared to BERT. Our HTC-CLIP model also achieves competitive improvement over HGCLR and Hi-Match in terms of both Macro-F1 and Micro-F1. We could not reproduce the BERT and HGCLR results reported in [20], so we report the results of our implementation of BERT and HGCLR. We ran the experiment 5 times on WOS dataset and the standard deviation for Micro-F1 is 0.26 and Macro-F1 is 0.17. The mean of Micro-F1 and Macro-F1 for HGCLR are statistically different from that for HTC-CLIP as per t-test. HGCLR mean is more than three standard deviations away from that of HTC-CLIP for both Micro and Macro F1 scores. This shows that in comparison to the state-of-the-art model, our model's results are statistically better.

On NYT, our approach shows 1.00% and 2.74% improvement on Micro-F1 and Macro-F1 respectively compared to BERT and performs significantly better than previous methods on both of the above measurements. The results show that HTC-CLIP achieves consistent improvement on the performance of Hierarchical Text Classification among WOS and NYT datasets.

Model	WOS Dataset		NYT Dataset	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1
BERT (Our implement)	85.80	79.20	78.22	65.85
BERT+HiAGM [20]	86.04	80.19	78.64	66.76
BERT+HTCInfoMax [20]	86.30	79.97	78.75	67.31
BERT+HiMatch [3]	86.70	81.06	-	-
HGCLR [20] (Our implement)	87.01	80.30	78.34	66.22
HTC-CLIP (Ours)	**87.86**	**81.64**	**79.22**	**68.59**

Table 2. Experimental results comparing the proposed model (HTC-CLIP) with other state-of-the-art models on WOS and NYT datasets. For a fair comparison, we implemented the same baseline as the BERT encoder. We cannot reproduce the BERT and HGCLR results reported in [20], so we report the results of our implementation of BERT and HGCLR.

Model	WOS	NYT
BERT(Our implement)	110.9	110.9
HGCLR(Our implement)	120.5	120.6
HTC-CLIP	120.7	120.8

Table 3. Number of weight parameters in the proposed model with other state-of-the-art models on WOS and NYT datasets. The numbers displayed in the table are in millions.

5.3 Analysis

In this section, we investigate the independent effect of each component in our proposed model.

Ablation Models	Micro-F1	Macro-F1
BERT	85.80	79.20
HTC-CLIP	**87.58**	**80.88**
-r.m. $\hat{L}_H^C$	86.65	80.09
-r.m. $\hat{L}_L^C$	86.95	80.13
-r.m. $\hat{L}_H^C$& $\hat{L}_L^C$	86.18	79.16
-r.m. l.c. and $\hat{L}_H^C$	86.70	79.75

Table 4. Performance of HTC-CLIP on the validation set of WOS after removal of BCE loss components from Equation 27. $\hat{L}_H^C$ and $\hat{L}_L^C$ are BCE loss of positive sample output representation learned through a contrastive loss for linear and path-guided hierarchy classifiers respectively, l.c. stands for a linear classifier, r.m. stands for remove

5.3.1 Effect of Contrastive Learning on Classifiers

To demonstrate the advantage of contrastive information, we tested our model with and without BCE loss of constructed positive sequence output representation on the WOS Dataset. We first removed $\hat{L}_H^C$ from Equation 27, results are shown in Table 4. We observed that Macro-F1 and Micro-F1 scores drop significantly after removing $\hat{L}_H^C$. BCE loss of positive sequence output learned through contrastive loss helps the path-guided hierarchy classifier in learning the text representation in its weights (section 4.3). We also removed the $\hat{L}_L^C$ and saw a similar trend of decreased Macro-F1 and Micro-F1 scores. As evident from Table 4, BCE loss of positive sequence output representation learned through contrastive loss helps in tuning weights and boosts performance in both linear classifier as well as path-guided hierarchy classifier.

Ablation Models	Micro-F1	Macro-F1
BERT	85.80	79.20
HTC-CLIP	**87.58**	**80.88**
-r.m. h.c.	86.69	80.38
-r.m. l.c.	87.16	80.11
-r.m. hidden layers from h.c.	86.38	79.88

Table 5. Performance of HTC-CLIP on the validation set of WOS after removal of classifiers. r.m. stands for remove, h.c. stands for path-guided hierarchy classifier and l.c. stands for the linear classifier.

5.3.2 Effect of Path Guided Hierarchy

To study the influence of path-guided hierarchy, we tested our model on the WOS Dataset after removing the path-guided hierarchy classifier. Without the path-guided hierarchy classifier, we found a drop in Macro-F1 and Micro-F1 scores. Results are shown in Table 5. We also experimented with the modified version of path-guided hierarchy after removing the hidden layers (Equation (5) and (6), Figure 3). As shown in Table 5, these layers contribute positively to the model and help in learning a better representation of the hierarchy.

Ablation Models	Micro-F1	Macro-F1
BERT	85.80	79.20
Only h.c. model	87.16	80.11
Only l.c. model	86.69	80.39
Both l.c. & h.c. with avg pool	87.35	80.69
Both l.c. & h.c. with max pool	87.01	80.42
Both l.c. & h.c. with max pool during inference (HTC-CLIP)	**87.58**	**80.88**

Table 6. Performance of HTC-CLIP on the validation set of WOS after changing some components. h.c. stands for path-guided hierarchy classifier and l.c. stands for linear classifier

5.3.3 Effect of Pooling Classifiers' Outputs

We trained the path-guided hierarchy model and linear classifier model with text-encoded hierarchy using contrastive loss. In our model, we propose to use both path-guided hierarchy and linear classifier in a single architecture, and a method to combine the output probabilities during inference. As seen from Table 6, our proposed architecture works better than individually trained models. This shows that the individual models capture complementary information and can be effectively combined into one architecture to achieve improved performance. We find that adding or taking the maximum of the probabilities of classifier outputs during training

yields better results than both the individual models, but is not able to outperform our proposed architecture where we do max pooling during inference.

6 Conclusion

In this paper, we present hierarchical text classification using contrastive learning-informed path-guided hierarchy (HTC-CLIP). The method combines the strength of two existing approaches: contrastive learning guided hierarchy in text encoder and path guided hierarchy. Our paper shows that the two previous approaches capture complementary information and can be effectively combined into one architecture to achieve improved performance. Our approach empirically achieves consistent improvements over the state-of-the-art on two public benchmark datasets. All of the components we designed are proven to be effective.

References

[1] Siddhartha Banerjee, Cem Akkaya, Francisco Perez-Sorrosal, and Kostas Tsioutsiouliklis, 'Hierarchical transfer learning for multi-label text classification', in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6295–6300, Florence, Italy, (July 2019). Association for Computational Linguistics.

[2] Haibin Chen, Qianli Ma, Zhenxi Lin, and Jiangyue Yan, 'Hierarchy-aware label semantics matching network for hierarchical text classification', in *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)*, pp. 4370–4379, Online, (August 2021). Association for Computational Linguistics.

[3] Haibin Chen, Qianli Ma, Zhenxi Lin, and Jiangyue Yan, 'Hierarchy-aware label semantics matching network for hierarchical text classification', in *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)*, pp. 4370–4379, Online, (August 2021). Association for Computational Linguistics.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, 'A simple framework for contrastive learning of visual representations', in *Proceedings of the 37th International Conference on Machine Learning*, eds., Hal Daumé III and Aarti Singh, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, (13–18 Jul 2020).

[5] Zhongfen Deng, Hao Peng, Dongxiao He, Jianxin Li, and Philip Yu, 'HTCInfoMax: A global model for hierarchical text classification via information maximization', in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3259–3265, Online, (June 2021). Association for Computational Linguistics.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina 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)*, pp. 4171–4186, Minneapolis, Minnesota, (June 2019). Association for Computational Linguistics.

[7] Siddharth Gopal and Yiming Yang, 'Recursive regularization for large-scale classification with hierarchical and graphical dependencies', in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, p. 257–265, New York, NY, USA, (2013). Association for Computing Machinery.

[8] Eric Jang, Shixiang Gu, and Ben Poole, 'Categorical Reparameterization with Gumbel-Softmax', *arXiv e-prints*, arXiv:1611.01144, (November 2016).

[9] Rie Johnson and Tong Zhang, 'Effective use of word order for text categorization with convolutional neural networks', in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 103–112, Denver, Colorado, (May–June 2015). Association for Computational Linguistics.

[10] Kamran Kowsari, Donald E. Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S. Gerber, and Laura E. Barnes, 'Hdltex: Hierarchical deep learning for text classification', in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 364–371, (Dec 2017).

[11] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao, 'Recurrent convolutional neural networks for text classification', in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, p. 2267–2273. AAAI Press, (2015).

[12] Yuning Mao, Jingjing Tian, Jiawei Han, and Xiang Ren, 'Hierarchical text classification with reinforced label assignment', in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 445–455, Hong Kong, China, (November 2019). Association for Computational Linguistics.

[13] Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han, 'Weakly-supervised hierarchical text classification', in *AAAI*, (2019).

[14] Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang, 'Large-scale hierarchical text classification with recursively regularized deep graph-cnn', in *Proceedings of the 2018 World Wide Web Conference*, WWW '18, p. 1063–1072, Republic and Canton of Geneva, CHE, (2018). International World Wide Web Conferences Steering Committee.

[15] Hao Peng, Jianxin Li, Senzhang Wang, Lihong Wang, Qiran Gong, Renyu Yang, Bo Li, S Yu Philip, and Lifang He, 'Hierarchical taxonomy-aware and attentional graph capsule rcnns for large-scale multi-label text classification', *IEEE Transactions on Knowledge and Data Engineering*, **33**(6), 2505–2519, (2019).

[16] Evan Sandhaus. The New York Times Annotated Corpus - Linguistic Data Consortium, 2008. [Online; accessed 0000-00-00].

[17] Kazuya Shimura, Jiyi Li, and Fumiyo Fukumoto, 'HFT-CNN: Learning hierarchical category structure for multi-label short text categorization', in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 811–816, Brussels, Belgium, (October-November 2018). Association for Computational Linguistics.

[18] Carlos N Silla and Alex A Freitas, 'A survey of hierarchical classification across different application domains', *Data Mining and Knowledge Discovery*, **22**(1), 31–72, (2011).

[19] Yangqiu Song and Dan Roth, 'On dataless hierarchical text classification', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, (2014).

[20] Zihan Wang, Peiyi Wang, Lianzhe Huang, Xin Sun, and Houfeng Wang, 'Incorporating hierarchy into text encoder: a contrastive learning approach for hierarchical text classification', in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7109–7119, Dublin, Ireland, (May 2022). Association for Computational Linguistics.

[21] Jônatas Wehrmann, Rodrigo C. Barros, Silvia N. das Dôres, and Ricardo Cerri, 'Hierarchical multi-label classification with chained neural networks', in *Proceedings of the Symposium on Applied Computing*, SAC '17, p. 790–795, New York, NY, USA, (2017). Association for Computing Machinery.

[22] Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros, 'Hierarchical multi-label classification networks', in *International Conference on Machine Learning*, pp. 5075–5084. PMLR, (2018).

[23] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush, 'Transformers: State-of-the-art natural language processing', in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, (October 2020). Association for Computational Linguistics.

[24] Jiawei Wu, Wenhan Xiong, and William Yang Wang, 'Learning to learn and predict: A meta-learning approach for multi-label classification', in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4354–4364, Hong Kong, China, (November 2019). Association for Computational Linguistics.

[25] Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang, 'SGM: Sequence generation model for multi-label classification', in *Proceedings of the 27th International Conference on Com-*

putational Linguistics, pp. 3915–3926, Santa Fe, New Mexico, USA, (August 2018). Association for Computational Linguistics.

[26] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu, 'Do transformers really perform badly for graph representation?', in *Advances in Neural Information Processing Systems*, eds., M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, volume 34, pp. 28877–28888. Curran Associates, Inc., (2021).

[27] Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li, 'Adversarial attacks on deep-learning models in natural language processing: A survey', *ACM Trans. Intell. Syst. Technol.*, **11**(3), (apr 2020).

[28] Jie Zhou, Chunping Ma, Dingkun Long, Guangwei Xu, Ning Ding, Haoyu Zhang, Pengjun Xie, and Gongshen Liu, 'Hierarchy-aware global model for hierarchical text classification', in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1106–1117, Online, (July 2020). Association for Computational Linguistics.