Hierarchical Text Classification as Sub-hierarchy Sequence Generation

SangHun Im, GiBaeg Kim, Heung-Seon Oh, Seongung Jo, Dong Hwan Kim

School of Computer Science and Engineering, Korea University of Technology and Education (KOREATECH)
{tkrhkshdqn, fk0214, ohhs, oowhat, hwan6615}@koreatech.ac.kr

Abstract

Hierarchical text classification (HTC) is essential for various real applications. However, HTC models are challenging to develop because they often require processing a large volume of documents and labels with hierarchical taxonomy. Recent HTC models based on deep learning have attempted to incorporate hierarchy information into a model structure. Consequently, these models are challenging to implement when the model parameters increase for a large-scale hierarchy because the model structure depends on the hierarchy size. To solve this problem, we formulate HTC as a sub-hierarchy sequence generation to incorporate hierarchy information into a target label sequence instead of the model structure. Subsequently, we propose the **Hi**erarchy **DEC**oder (HiDEC), which decodes a text sequence into a sub-hierarchy sequence using recursive hierarchy decoding, classifying all parents at the same level into children at once. In addition, HiDEC is trained to use hierarchical path information from a root to each leaf in a sub-hierarchy composed of the labels of a target document via an attention mechanism and hierarchy-aware masking. HiDEC achieved state-of-the-art performance with significantly fewer model parameters than existing models on benchmark datasets, such as RCV1-v2, NYT, and EURLEX57K.

Introduction

Hierarchical text classification (HTC) uses a hierarchy such as a web taxonomy to classify a given text into multiple labels. Moreover, classification tasks are essential in real-world applications because of the tremendous amount of data on the web that should be properly organized for applications such as product navigation (Kozareva 2015; Cevahir and Murakami 2016) and news categorization (Lewis et al. 2004; Sandhaus. 2008).

Recent HTC research using deep learning can be categorized into local and global approaches. In the local approach (Peng et al. 2018; Kowsari et al. 2017; Shimura, Li, and Fukumoto 2018; Banerjee et al. 2019; Dumais and Chen 2000; Wehrmann, Cerri, and Barros 2018), a classifier is built for each unit after the entire hierarchy is split into a set of small units. Subsequently, the classifiers are applied in sequence according to a path from a root to target labels in a top-down manner. In contrast, in the global approach (Zhao et al. 2018; Wang et al. 2021; Peng et al. 2021; Wang et al. 2022; Zhou et al. 2020; Mao et al. 2019; Chen et al. 2021; Sinha et al. 2018; Deng et al. 2021; Wu, Xiong, and Wang 2019; Yang et al. 2018), a classifier for all labels in the entire hierarchy is built, excluding the hierarchy structure through flattening. A document hierarchy information can be obtained using a structure encoder and merged with text features from a text encoder. The global approach achieves superior performance to the local approach owing to the effective design of the structure encoders, such as the graph convolution network (Kipf and Welling 2017) and Graphormer (Ying et al. 2021).

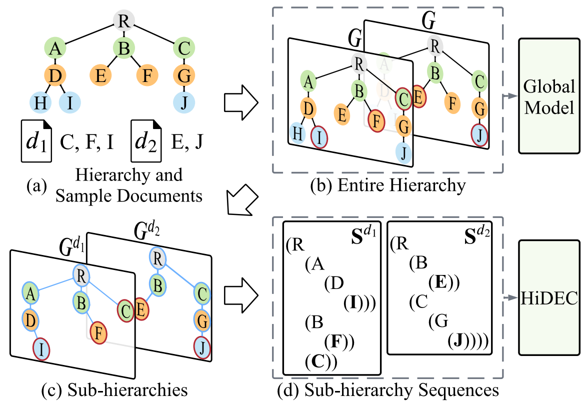


Figure 1: Example of converting the target labels of two documents to the sub-hierarchy sequences. The existing global model uses the entire hierarchy twice. In contrast, the proposed HiDEC uses the sufficiently small sub-hierarchies relevant to the documents twice.

However, the global approach has scalability limitations because the structure encoders require a disproportionate number of parameters as the size of the hierarchies increases. For example, HiAGM (Zhou et al. 2020) and HiMatch (Chen et al. 2021), recent GCN-based structure encoders, require a weight matrix to convert text features to label features. In HGCLR (Wang et al. 2022), edge and spatial encodings were employed to represent the relationship between two nodes.

In existing global models, a large model size is inevitable because they attempt to incorporate the entire hierarchy information with respect to the document labels into a model structure. In contrast, employing a sub-hierarchy comprising a set of paths from a root to each target label is sufficient because most labels are irrelevant to a target document. To this end, we formulate HTC as a sub-hierarchy sequence generation using an encoder-decoder architecture to incorporate the sub-hierarchy information into a target label sequence instead of the model structure. Figure 1 shows the differences between the proposed approach and the existing global models. For example, given a hierarchy and two documents with different labels in Figure 1-(a), a global model attempts to capture the entire hierarchy information with respect to the document labels, as shown in Figure 1-(b). For further improvement, the document labels are converted into a sub-hierarchy sequence using a depth-first search on an entire hierarchy and parse tree notation, as shown in Figure 1-(c) and -(d).

Based on this idea, we propose a **Hi**erarchy **DEC**oder (HiDEC)[1], which recursively decodes the text sequence into a sub-hierarchy sequence by sub-hierarchy decoding while remaining aware of the path information. The proposed method comprises hierarchy embeddings, hierarchy-aware masked self-attention, text-hierarchy attention, and sub-hierarchy decoding, similar to the decoder of Transformer (Vaswani et al. 2017). Hierarchy-aware masked self-attention facilitates learning all hierarchy information in a sub-hierarchy sequence. A hierarchy-aware mask captures the sub-hierarchy information by considering the dependencies between each label and its child labels from the sub-hierarchy sequence at once in the training step. Subsequently, two features generated from the hierarchy-aware masked self-attention and a text encoder are merged through text-hierarchy attention. Note that HiDEC does not require additional parameters for the structure encoder as used in global models, such as HiAGM, HiMatch, and HGCLR. Consequently, the parameters of the model increase linearly with respect to the classes in a hierarchy. In the inference step, sub-hierarchy decoding is recursively applied to expand from parent to child labels in a top-down manner. As a result, all parent labels at the same depth are expanded simultaneously. Thus, the maximum recursions are the depth of the entire hierarchy and not the sequence length.

A series of experiments show that HiDEC outperforms state-of-the-art (SOTA) models on two small-scale datasets, RCV1-v2 and NYT, and a large-scale dataset, EURLEX57K. Consequently, HiDEC achieves better performance with significantly fewer parameters on the three benchmark datasets. Thus, the proposed approach can solve scalability problems in large-scale hierarchies.

The contributions of this paper can be summarized as follows:

- This paper formulates HTC as a sub-hierarchy sequence generation using an encoder-decoder architecture. We can incorporate the hierarchy information into the sub-hierarchy sequence instead of the model structure, as all dependencies are aware by parse tree notation.
- This paper proposes a Hierarchy DECoder (HiDEC) that recursively decodes the text sequence into a sub-hierarchy sequence by sub-hierarchy decoding while remaining aware of the path information.
- This paper demonstrates the superiority of HiDEC by comparing SOTA models on three benchmark HTC datasets (RCV1-v2, NYT, and EURLEX57K). The results reveal the role of HiDEC in HTC through in-depth analysis.

[1]Code is available on https://github.com/SangHunIm/HiDEC

Related Work

The critical point to HTC is the use of hierarchy information, denoted as relationships among labels. For example, the relationships include root-to-target labels (path information), parent-to-child, and the entire hierarchy (holistic information).

Research on HTC can be categorized into local and global approaches. In the local approach, a set of classifiers are used for small units of classes, such as for-each-class (Banerjee et al. 2019), for-each-parent (Dumais and Chen 2000; Kowsari et al. 2017), for-each-level (Shimura, Li, and Fukumoto 2018), and for-each-sub-hierarchy (Peng et al. 2018). In (Kowsari et al. 2017), HDLTex was introduced as a local model that combined a deep neural network (DNN), CNN, and RNN to classify child nodes. Moreover, HTrans (Banerjee et al. 2019) extended HDLTex to maintain path information across local classifiers based on transfer learning from parent to child. HMCN (Wehrmann, Cerri, and Barros 2018) applied global optimization to the classifier of each level to solve the exposure bias problem. Finally, HR-DGCNN (Peng et al. 2018) divided the entire hierarchy into sub-hierarchies using recursive hierarchical segmentation. Unfortunately, applying this to large-scale hierarchies is challenging because many parameters are required from a set of local classifiers for small units of classes.

In the global approach, the proposed methods employed a single model with path (Zhao et al. 2018; Wang et al. 2021; Peng et al. 2021; Mao et al. 2019; Sinha et al. 2018; Deng et al. 2021) or holistic (Zhou et al. 2020; Chen et al. 2021; Wang et al. 2022) information. For instance, HNATC (Sinha et al. 2018) obtained path information using a sequence of outputs from the previous levels to predict the output at the next level. In HiLAP-RL (Mao et al. 2019), reinforcement learning was exploited in that HTC was formalized as a pathfinding problem. Moreover, HE-AGCRCNN (Peng et al. 2021) and HCSM (Wang et al. 2021) used capsule networks. Recent research (Zhou et al. 2020; Chen et al. 2021; Wang et al. 2022) has attempted to employ holistic information of an entire hierarchy using a structure encoder with GCN (Kipf and Welling 2017) and Graphormer (Ying et al. 2021). HiAGM (Zhou et al. 2020) propagated text through GCN, HiMatch (Chen et al. 2021) improved HiAGM by adapting semantic matching between text and label features from text features and label embeddings. In addition, HGCLR (Wang et al. 2022) attempted to unify a structure encoder with a text encoder using a novel contrastive learning method, where BERT (Devlin et al. 2019) and Graphormer were employed for the text and structure encoders, respectively. Therefore, classification was performed

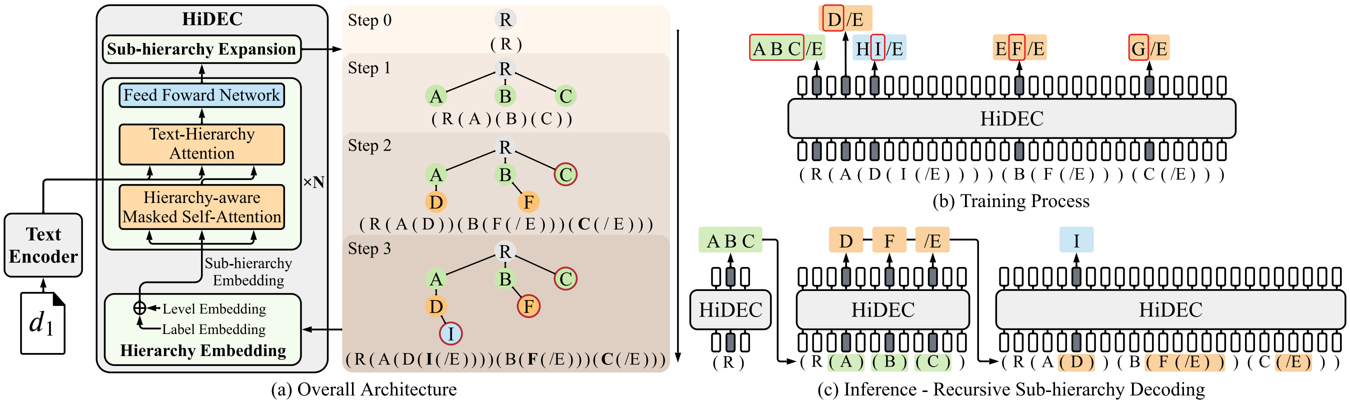


Figure 2: (a): The overall architecture of HiDEC. A feature vector from a text encoder is decoded to a sub-hierarchy sequence, which is expanded one level at once by starting from the root. (b) and (c): Illustration of input and output in training and inference, respectively. In step 3 of (a) and the right of (c), HiDEC correctly generates the sub-hierarchy (sequence) we expected. /E means "[END]" token.

using a hierarchy-aware text feature produced by the text encoder. Finally, it reported SOTA performance on RCV1-v2 and NYT but is infeasible in large-scale hierarchies because of the large model size caused by incorporating holistic information into the model structure.

Proposed Methods

The HTC problem can be defined using a tree structure. A hierarchy is represented as a tree $G = (V, \vec{E})$ where $V = \{v_1, v_2, \ldots, v_C\}$ is a set of C-labels in the hierarchy, and $\vec{E} = \{(v_i, v_j) | v_i \in V, v_j \in child(v_i)\}$ is a set of edges between a label v_i and a child v_j of v_i. $D = \{d_1, d_2, \ldots, d_K\}$ is a collection of K documents. A document d_k has a sub-hierarchy $G^{d_k} = (V^{d_k}, \vec{E}^{d_k})$ converted from assigned labels where $V^{d_k} = L^{d_k} \cup \{v_i^{d_k} | v_i^{d_k} \in ancestor(v_j^{d_k}), v_j^{d_k} \in L^{d_k}\}$ and $\vec{E}^{d_k} = \{(v_i, v_j) | v_j \in V^{d_k}, v_i \in parent(v_j)\}$, where $L^{d_k} = \{v_1^{d_k}, v_2^{d_k}, \ldots, v_t^{d_k}\}$ is a label set of document d_k. In other words, G^{d_k} is constructed using all the labels assigned to d_k and their ancestors. $\hat{G}_0^{d_k} = (\{v_{root}\}, \emptyset)$ is the initial sub-hierarchy of HiDEC, which has a root and no edges. Based on $\hat{G}_0^{d_k}$, recursive hierarchy decoding is defined by expanding $\hat{G}_p^{d_k}$ for p times from p=0. The goal of training HiDEC is given by $\hat{G}_p^{d_k} = G^{d_k}$.

In Figure 2-(a), the overall architecture of HiDEC is presented with a demonstration of the recursive hierarchy decoding. The remainder of this section presents the details of the proposed model.

Text Encoder

In the proposed model, a text encoder can use any model that outputs the text feature matrix of all the input tokens, such as GRU and BERT (Devlin et al. 2019). For simplicity, let us denote d_k as $\mathbf{T} = [w_1, w_2, \ldots, w_N]$ where w_n is a one-hot vector for an index of the n-th token. Initially, a sequence of tokens was converted into word embeddings $\mathbf{H}^0 (= \mathbf{W}^0\mathbf{T}) \in \mathbb{R}^{N \times e}$ where $\mathbf{W}^0$ is the weight matrix of the word embedding layer, and e is an embedding dimension. Given $\mathbf{H}^0$, the hidden state $\mathbf{H}$ from the text encoder can be computed using Equation 1:

$$\mathbf{H} = \text{TextEncoder}(\mathbf{H}^0). \quad (1)$$

Hierarchy DECoder (HiDEC)

Hierarchy Embedding Layer The sub-hierarchy embeddings, as shown in Figure 1, is obtained by initially constructing a sub-hierarchy sequence from a document d_k. This process consists of two steps. First, a sub-hierarchy $G^{d_k} = (V^{d_k}, \vec{E}^{d_k})$ of d_k is built with its target labels. Second, a sub-hierarchy sequence $\mathbf{S}$ following a parse tree notation is generated from G^{d_k}. Three special tokens, "(", ")", and "[END]", are used to properly represent the sub-hierarchy. The tokens "(" and ")" denote the start and end of a path from each label, respectively, whereas the "[END]" token indicates the end of a path from a root. For example, **S**=[(R (A (D (I ([END]))))) (B (F ([END]))) (C ([END])))] is constructed in Figure 1 with a label set [C,F,I]. Once again, the tokens in $\mathbf{S}$ are represented as one-hot vectors for further processing. Subsequently, these tokens can be represented as $\bar{\mathbf{S}} = [s_1, s_2, \ldots, s_M]$ where $s_i = \mathbb{I}_v$ is a one-hot vector for a label v and the special tokens. Finally, the sub-hierarchy embeddings $\mathbf{U}^0$ are constructed after explicitly incorporating the level information, similar to Transformer's position encoding (Vaswani et al. 2017), using Equations 2 and 3:

$$\bar{\mathbf{U}}^0 = \mathbf{W}^s\bar{\mathbf{S}}, \quad (2)$$

$$\mathbf{U}^0 = \text{LevelEmbedding}(\bar{\mathbf{U}}^0). \quad (3)$$

Hierarchy-Aware Masked Self-Attention This component is responsible for capturing hierarchy information, similar to the structure encoder in global models (Zhou et al. 2020; Chen et al. 2021; Wang et al. 2022). However, only a sub-hierarchy from the entire hierarchy, which is thought to be highly relevant information for classification, is used based on the self-attention mechanism used by Transformer

Algorithm 1: Recursive Hierarchy Decoding in Inference

Indices: Hierarchy depth P, Number of attentive layers R
Input: Text feature matrix from text encoder $\mathbf{H}$
Output: Predicted label set L

//HiDEC
$L = \emptyset$
$\hat{G}_0 = (\{v_{root}\}, \emptyset)$
for $p = 0, \dots, P-1$ **do**
  //Sub-hierarchy embedding
  Convert $\hat{G}_p$ to sub-hierarchy sequence $\mathbf{S}_p$
  Compute $\mathbf{U}^0$ from $\mathbf{S}_p$ with Eq.2, 3
  Generate masking matrix $\mathbf{M}$ with Eq.5
  //Attentive layers
  for $r = 0, \dots, R-1$ **do**
    $\mathbf{U}^{r+1}$ = Attention($\mathbf{U}^r, \mathbf{H}, \mathbf{M}$) with Eq.4, 6, 7
  end for
  $\mathbf{U} = \mathbf{U}^R$
  //Sub-hierarchy expansion
  for $s_i \in S_p$ **do**
    if $s_i \notin$ special token set **then**
      $v_i = s_i$
      for $v_j \in child(v_i)$ **do**
        $c_{ij} = U_i \cdot \mathbf{W}^S \cdot \mathbb{I}_{v_j}$ with Eq.8
        $p_i = \text{sigmoid}(c_i)$ with Eq.9
        Get $\hat{y}_i$ from p_i by thresholding
        for $v_k \in \hat{y}_i$ **do**
          $V_p = V_p \cup \{v_k\}$
          $\vec{E}_p = \vec{E}_p \cup \{(v_i, v_k)\}$
        end for
      end for
    end if
  end for
  $\hat{G}_{p+1} = (V_p, \vec{E}_p)$
end for
//Label assignment
for $i = 0, \dots, |V_P|$ **do**
  if $v_i \in leaf(\hat{G}_P)$ **then**
    $L = L \cup \{v_i\}$
  else if $v_i ==$"[END]" **then**
    $L = L \cup \{parent(v_i)\}$
  end if
end for
return L

(Vaswani et al. 2017). To compute self-attention scores, we applied hierarchy-aware masking to incorporate hierarchy information. The self-attention mechanism of Transformer was exploited with a minor modification concerning hierarchy-aware masking. The hierarchy-aware masked self-attention of r-th layer is computed using Equation 4:

$$\dot{\mathbf{U}}^r = \text{MHA}(\mathbf{W}_Q^r \mathbf{U}^{r-1}, \mathbf{W}_K^r \mathbf{U}^{r-1}, \mathbf{W}_V^r \mathbf{U}^{r-1}, \mathbf{M}), \quad (4)$$

where MHA is the multi-head attention, the same as that of Transformer. $\mathbf{W}_Q^r$,$\mathbf{W}_K^r$,$\mathbf{W}_V^r$ are projection weight matrices for the query, key, and value, respectively. Moreover, $\mathbf{M}$ is the hierarchy-aware mask defined as follows:

$$\mathbf{M}_{ij} = \begin{cases} -1e9 & \text{if } v_i \notin ancestor(v_j) \\ 0 & \text{else} \end{cases}. \quad (5)$$

We ignore the dependency between two labels if they are not the same label and not an ancestor by setting $\mathbf{M}_{ij} = -1e9$. This setting makes the model attend to the path information relevant to the input documents and ignores the hierarchy information at the lower-level labels than each label to learn a sub-hierarchy sequence at once in a training step. Note that the dependencies of the three special tokens with respect to the other tokens, including themselves, are considered.

Text-Hierarchy Attention In text-hierarchy attention, we can compute the attention scores of labels by dynamically reflecting the importance of tokens in an input document. A new sub-hierarchy matrix $\ddot{\mathbf{U}}^r$ of r-th layer is computed by combining the text feature matrix $\mathbf{H}$ from the encoder and $\dot{\mathbf{U}}^r$ without a masking mechanism using Equation 6:

$$\ddot{\mathbf{U}}^r = \text{MHA}(\mathbf{W}_Q^r \dot{\mathbf{U}}^r, \mathbf{W}_K^r \mathbf{H}, \mathbf{W}_V^r \mathbf{H}, -). \quad (6)$$

Subsequently, the output of r-th layer $\mathbf{U}^r$ is obtained using a position-wise feed-forward network (FFN) using Equation 7:

$$\mathbf{U}^r = \text{FFN}(\ddot{\mathbf{U}}^r). \quad (7)$$

Consequently, the output of the final layer, $\mathbf{U}$, in HiDEC is used in the sub-hierarchy expansion.

Sub-Hierarchy Decoding Sub-hierarchy decoding is crucial in generating a sub-hierarchy using recursive hierarchy decoding. This results in a target sub-hierarchy if HiDEC functions as expected. For each label, the classification to child labels is performed using the sub-hierarchy matrix $\mathbf{U}$ using Equations 8 and 9:

$$c_{ij} = U_i \cdot \mathbf{W}^S \cdot \mathbb{I}_{v_j} \quad \forall v_j \in child(v_i), \quad (8)$$

$$p_i = \text{sigmoid}(c_i), \quad (9)$$

where c_{ij} is a similarity score of child v_j under a parent v_i, and p_i is the probability of a child v_j obtained using a task-specific probability function such as sigmoid. The three special tokens are excluded when selecting the parent v_i.

We reduced the label space of HTC by focusing on the child labels of a parent label of interest. During training, we use binary cross-entropy loss functions as shown in Equation 10:

$$\mathcal{L} = -\frac{1}{MJ} \sum_{i=0}^{M} \sum_{j=0}^{J} y_{ij} \log(p_{ij}) + (1 - y_{ij}) \log(1 - p_{ij}), \quad (10)$$

where $J = |child(v_i)|$ indicates the number of child labels for parent v_i. Moreover, y_{ij} and p_{ij} denote a target label of j-th child label of v_i and its output probability, respectively.

At the inference time, recursive hierarchy decoding is performed using a threshold. The details of the recursive hierarchy decoding are described in Algorithm 1. The number of decoding steps is the same as the maximum depth of the hierarchy. At each decoding step, all tokens except the special tokens are expanded. Decoding ends if the tokens are leaf labels or "[END]". Finally, the labels associated with "[END]" or leaf labels are assigned to the input as predictions.

Dataset	$\|L\|$	Depth	Avg	Train	Val	Test
RCV1-v2	103	4	3.24	20,833	2,316	781,265
NYT	166	8	7.60	23,345	5,834	7,292
EURLEX57K	4,271	5	5.00	45,000	6,000	6,000

Table 1: Data statistics. $|L|$ denotes the number of labels. Depth and Avg are the maximum hierarchy depth and the average number of assigned labels for each text, respectively.

Experiments

Datasets and Evaluation Metrics

Table 1 lists the data statistics used in the experiments. For the standard evaluation, two small-scale datasets, RCV1-v2 (Lewis et al. 2004) and NYT (Sandhaus. 2008), and one large-scale dataset, EURLEX57K (Chalkidis et al. 2019), were chosen. RCV1-v2 comprises 804,414 news documents, divided into 23,149 and 781,265 documents for training and testing, respectively, as benchmark splits. We randomly sampled 10% of the training data as the validation data for model selection. NYT comprises 36,471 news documents divided into 29,179 and 7,292 documents for training and testing, respectively. For a fair comparison, we followed the data configurations of previous work (Zhou et al. 2020; Chen et al. 2021). In particular, EURLEX57K is a large-scale hierarchy with 57,000 documents and 4,271 labels. Benchmark splits of 45,000, 6,000, and 6,000 were used for training, validation, and testing, respectively. We used Micro-F1 for three datasets and Macro-F1 for RCV1-v2 and NYT.

Implementation Details

After text cleaning and stopword removal, words with two or more occurrences were selected to retain the vocabulary. Consequently, the vocabulary sizes for the three benchmark datasets were 60,000.

For the text encoder, we opted the simplest encoder, a bidirectional GRU with a single layer, for a fair comparison to previous work (Zhou et al. 2020; Chen et al. 2021). The size of the hidden state was set to 300. The word embeddings in the text encoder were initialized using 300-dimensional GloVe (Pennington, Socher, and Manning 2014). In contrast to the GRU-based encoder, we used BERT, bert-base-uncased (Devlin et al. 2019), to demonstrate the generalization ability of the pre-trained encoder. The output hidden matrix from the last layer of BERT was used as the context matrix $\mathbf{H}$ in Equation 1.

For HiDEC, a layer with two heads were used for both GRU-based encoder and BERT. The label and level embeddings with 300- and 768-dimension for the GRU-based encoder and BERT, respectively, were initialized using a normal distribution with μ=0 and σ=$300^{-0.5}$. The hidden state size in the attentive layer was the same as the label embedding size. The FFN comprised two FC layers with 600- and 3,072-dimension feed-forward filter for the GRU-based encoder and BERT, respectively. Based on an empirical test, we removed the residual connection from the original Transformer decoder (Vaswani et al. 2017). The threshold for recursive hierarchy decoding was set to 0.5. A dropout with a probability of 0.5, 0.1, and 0.1 was applied to the embedding layer and behind every FFN and attention, respectively.

For optimization, Adam optimizer (Kingma and Ba 2015) was utilized with learning rate lr=1e-4, β_1=0.9, β_2=0.999, and eps=1e-8. The size of the mini-batch was set to 256 for GRU-based models. With BERT as a text encoder model, we set lr and the mini-batch size to 5e-5 and 64, respectively. The lr was controlled using a linear schedule with a warm-up rate of 0.1. Gradient clipping with a maximum gradient norm of 1.0 was performed to prevent gradient overflow.

All models were implemented using PyTorch (Paszke et al. 2019) and trained using NVIDIA A6000. The average score of the ten different models was utilized as the proposed model performance, where the model with the best performance was selected using Micro-F1 on the validation data.

Comparison Models

We selected various baseline models from recent work. For RCV1-v2 and NYT, TextRCNN (Lai et al. 2015), HiAGM (Zhou et al. 2020), HiMatch (Chen et al. 2021), HTCInfoMax (Deng et al. 2021), and HGCLR (Wang et al. 2022) were chosen. TextRCNN comprises bi-GRU and CNN layers and is a hierarchy-unaware model used as a text encoder in HiAGM and HiMatch. HiAGM combines text and label features from the text encoder and GCN-based structure encoder, respectively, using text propagation. HTCInfoMax and HiMatch improved HiAGM with a prior distribution and semantic matching loss, respectively. They can be improved by replacing the text encoder with PLM. HGCLR directly embeds hierarchy information into the text encoder using Graphormer during training. For EURLEX57K, BiGRU-ATT (Xu et al. 2015), HAN (Yang et al. 2016), CNN-LWAN (Mullenbach et al. 2018), BiGRU-LWAN (Chalkidis et al. 2019), and HiMatch (Chen et al. 2021) were chosen. BiGRU-ATT and HAN are strong baselines for text classification tasks with an attention mechanism. CNN and BiGRU-LWAN extended BiGRU-ATT with label-wise attention. Owing to the large model size, applying HiMatch to EURLEX57K is infeasible. According to the paper (Chen et al. 2021), the text-propagation module weakly influences the performance compared with the original approach. Therefore, we simplified HiMatch by removing the text-propagation module.

Experimental Results

Table 2 summarizes the performance of HiDEC and the other models on (a) two small-scale datasets, RCV1-v2 and NYT, and (b) a large-scale dataset, EURLEX57K. HiDEC achieved the best performance on the three datasets regardless of whether BERT was used (Devlin et al. 2019) as the text encoder. This highlights the effectiveness of HiDEC using sub-hierarchy information rather than entire-hierarchy information. In addition, we found that HiDEC highly benefits from PLM compared to other models, as the performance gains of HiDEC with and without BERT are relatively large. For example, in MicroF1 on RCV1-v2, the gain between HiDEC and HiDEC with PLM was 2.42. In contrast, the gain between HiMatch (Chen et al. 2021) and Hi-

Model	RCV1-v2		NYT	
	Micro	Macro	Micro	Macro
w/o Pretrained Language Models				
TextRCNN* (Zhou et al. 2020)	81.57	59.25	70.83	56.18
HiAGM (Zhou et al. 2020)	83.96	63.35	74.97	60.83
HTCInfoMax (Deng et al. 2021)	83.51	62.71	-	-
HiMatch (Chen et al. 2021)	84.73	64.11	-	-
HiDEC	**85.54**	**65.08**	**76.42**	**63.99**
w/ Pretrained Language Models				
BERT* (Wang et al. 2022)	85.65	67.02	78.24	65.62
HiAGM (Wang et al. 2022)	85.58	67.93	78.64	66.76
HTCInfoMax (Wang et al. 2022)	85.53	67.09	78.75	67.31
HiMatch (Chen et al. 2021)	86.33	68.66	-	-
HGCLR (Wang et al. 2022)	86.49	68.31	78.86	67.96
HiDEC	**87.96**	**69.97**	**79.99**	**69.64**

(a)

Model	EURLEX57K
	Micro
w/o Pretrained Language Models	
BiGRU-ATT* (Chalkidis et al. 2019)	68.90
HAN* (Chalkidis et al. 2019)	68.00
CNN-LWAN* (Chalkidis et al. 2019)	64.20
BiGRU-LWAN* (Chalkidis et al. 2019)	69.80
HiMatch	71.11
HiDEC	**71.23**
w/ Pretrained Language Models	
BERT*	73.20
HiDEC	**75.29**

(b)

Table 2: Performance comparison on three datasets, (a) RCV1-v2 and NYT, (b) EURLEX57K. * denotes models without hierarchy information.

Model	RCV1-v2	NYT	EURLEX57K
	103	166	4,271
w/o Pretrained Language Models			
TextRCNN*	18M	18M	19M
HiAGM	31M	42M	5,915M
HiMatch	37M	52M	6,211M
HiDEC	**20M**	**20M**	**21M**
w/ Pretrained Language Models			
BERT*	109M	109M	112M
HGCLR	**120M**	**121M**	425M
HiDEC	123M	123M	**127M**

Table 3: Model parameter comparison. * denotes models without hierarchy information.

Model	RCV1-v2		EURLEX57K
	Micro	Macro	Micro
HiDEC	**85.54**	**64.04**	**71.31**
+ Residual connection	84.69	60.32	69.97
− Hierarchy-aware masking	85.46	63.73	71.22
− Level Embedding	85.51	63.91	70.90

Table 4: Ablation studies on RCV1-v2 and EURLEX57K.

Match with PLM was 1.60. On EURLEX57K, training the hierarchy-aware models was infeasible because of the large model size[2] caused by the structure encoder for 4,271 labels, except for HiDEC and simplified HiMatch. However, this does not apply to BERT because of the model size. Similar to RCV1-v2 and NYT, HiDEC with BERT exhibited the best performance.

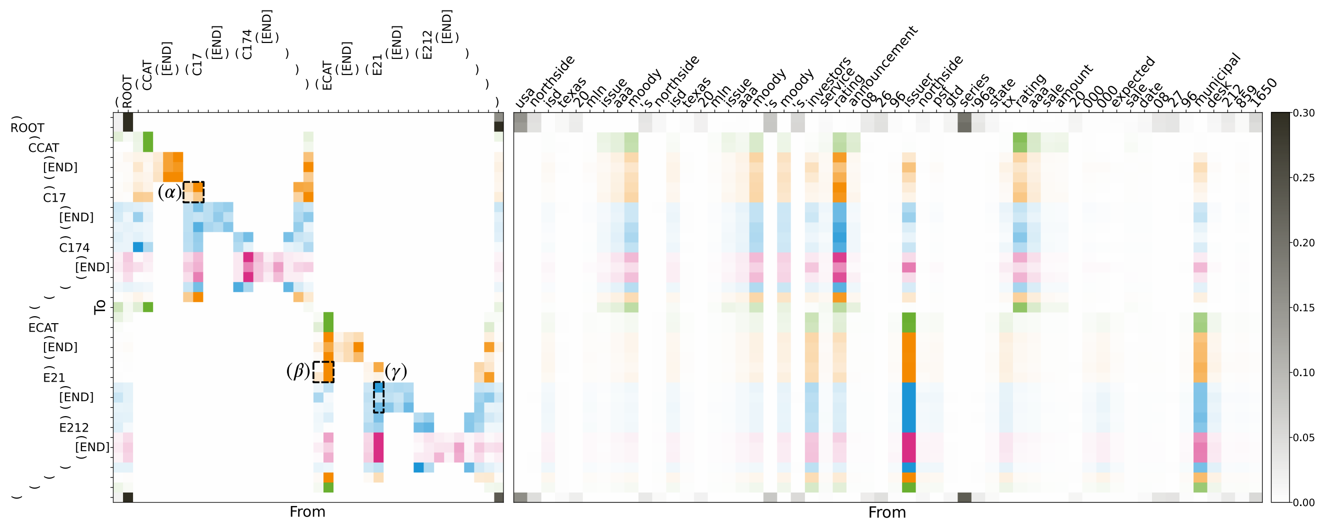
Model Parameters

Table 3 summarizes the parameters for different models on three benchmark datasets. The label sizes are RCV1-v2 (103) < NYT (166) << EURLEX57K (4,271). The table shows that the parameters of the existing models increase dramatically as the label size increases. Note that HiDEC requires significantly fewer parameters even though the label size increases. In the extreme case on EURLEX57K, HiDEC only requires 21M parameters, which are 295x smaller than HiMatch with 6,211M. Consequently, the parameters in HiDEC increase linearly with respect to the labels in a hierarchy because no extra parameters are required for the structure encoder and sub-tasks. HiAGM and HiMatch require merging parameters projecting text features into label features for text propagation. In addition, HGCLR needs edge and spatial encoding parameters for the structure encoder. However, HiDEC does not require these parameters because attentive layers play the same role. Only the label embeddings increase according to the label size in a hierarchy.

Ablation Studies

Table 4 shows the ablation studies of each component in HiDEC without PLM on RCV1-v2 and EURLEX57K. HiDEC differs from the original Transformer decoder (Vaswani et al. 2017) in the absence of a residual connection and the existence of hierarchy-aware masking and level embedding. We observed that adding the residual connection and eliminating hierarchy-aware masking and level embedding also had a negative effect. Among them, adding a residual connection had the most negative effect. We presumed that the essential information from previous features for HTC was hindered by the residual connection.

[2]Except for HiDEC and simplified HiMatch, we encountered out-of-memory with NVIDIA A6000 48GB for other models in training.



(a) Hierarchy-aware Masked Self-attention (b) Text-Hierarchy Attention

Figure 3: Heatmaps of the attention scores in HiDEC on RCV1-v2. (a) and (b) are heatmap of the hierarchy-aware masked self-attention scores and text-hierarchy attention scores, respectively. Attention scores over 0.3 are clipped in all the heatmaps and the similar colors indicate the same level.

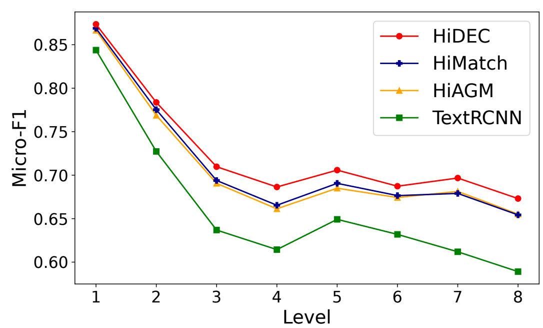


Figure 4: The level-wise performance of GRU-based HTC models on NYT.

Interpretation of Attentions

We investigated the roles of hierarchy-aware masked self-attention and text-hierarchy attention by visualizing the attention scores on RCV1-v2, as shown in Figure 3. The self-attention scores are shown in Figure 3-(a). In (α), the attention score between "(" and "C17" is relatively high where "(" is a starting path from "C17". The score shows that the special tokens "(" and ")" were appropriately associated with the corresponding labels. In (β), the dependency between child "E21" and parent "ECAT" was well-described because the attention scores for child labels under parent "ECAT" were high. In (γ), the results show a dependency between the label assignment sequence – ["(", "END", ")"] and the label "E21". From these three examples, we can conclude that hierarchy-aware masked self-attention effectively captures the path dependencies. Figure 3-(b) shows the attention scores between the input tokens and a sub-hierarchy sequence. Some tokens, such as the "rating" and "moody," have high attention scores for the descendants of "CCAT" and itself, where "CCAT" denotes "CORPORATE/INDUSTRIAL". In contrast, some tokens like "issuer," "municipal," and "investors" have high attention scores for the descendants of "EACT" and itself, where "EACT" denotes "ECONOMICS". This result indicates that the labels are associated with different tokens to different degrees.

Level-Wise Performance

Figure 4 depicts the level-wise performance of the models using a GRU-based text encoder on NYT. It shows the effectiveness of the hierarchy-aware models by comparing TextRCNN increases as the level increases. Among them, HiDEC consistently achieved the best performance at all levels. Note that significant improvements were obtained at the deep levels, implying that sub-hierarchy information is more powerful in capturing the structure information of a target document than the entire hierarchy information.

Conclusion

This paper addressed the scalability limitations of recent HTC models due to the large model size of the structure encoders. To solve this problem, we formulated HTC as a sub-hierarchy sequence generation using an encoder-decoder architecture. Subsequently, we propose Hierarchy DECoder (HiDEC) which recursively decodes the text sequence into a sub-hierarchy sequence by sub-hierarchy decoding while staying aware of the path information. HiDEC achieved state-of-the-art performance with significantly fewer model parameters than existing models on benchmark datasets, such as RCV1-v2, NYT, and EURLEX57K. In the future, we plan to extend the proposed model to extremely large-scale hierarchies (e.g., MeSH term indexing or product navigation) and introduce a novel training strategy combining top-down and bottom-up methods that can effectively use a hierarchy structure.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) (No. NRF-2019R1G1A1003312) and (No. NRF-2021R1I1A3052815).

References

Banerjee, S.; Akkaya, C.; Perez-Sorrosal, F.; and Tsioutsiouliklis, K. 2019. Hierarchical Transfer Learning for Multi-label Text Classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 6295–6300. Association for Computational Linguistics.

Cevahir, A.; and Murakami, K. 2016. Large-scale Multi-class and Hierarchical Product Categorization for an E-commerce Giant. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 525–535. The COLING 2016 Organizing Committee.

Chalkidis, I.; Fergadiotis, E.; Malakasiotis, P.; and Androutsopoulos, I. 2019. Large-Scale Multi-Label Text Classification on EU Legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 6314–6322. Association for Computational Linguistics.

Chen, H.; Ma, Q.; Lin, Z.; and Yan, J. 2021. 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)*, 4370–4379. Association for Computational Linguistics.

Deng, Z.; Peng, H.; He, D.; Li, J.; and Yu, P. 2021. 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*, 3259–3265. Association for Computational Linguistics.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. 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)*, 4171–4186. Association for Computational Linguistics.

Dumais, S.; and Chen, H. 2000. Hierarchical Classification of Web Content. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 256–263. Association for Computing Machinery. ISBN 1581132263.

Kingma, D. P.; and Ba, J. L. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Kowsari, K.; Brown, D. E.; Heidarysafa, M.; Meimandi, K. J.; Gerber, M. S.; and Barnes, L. E. 2017. HDLTex: Hierarchical Deep Learning for Text Classification. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 364–371.

Kozareva, Z. 2015. Everyone Likes Shopping! Multi-class Product Categorization for e-Commerce. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1329–1333. Association for Computational Linguistics.

Lai, S.; Xu, L.; Liu, K.; and Zhao, J. 2015. Recurrent Convolutional Neural Networks for Text Classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2267–2273. AAAI Press. ISBN 0262511290.

Lewis, D. D.; Yang, Y.; Rose, T. G.; Li, F.; and LEWIS, F. L. 2004. RCV1: A New Benchmark Collection for Text Categorization Research. *journal of Machine Learning Research*, 5: 361–397.

Mao, Y.; Tian, J.; Han, J.; and Ren, X. 2019. 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)*, 445–455. Association for Computational Linguistics.

Mullenbach, J.; Wiegreffe, S.; Duke, J.; Sun, J.; and Eisenstein, J. 2018. Explainable Prediction of Medical Codes from Clinical Text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 1101–1111. Association for Computational Linguistics.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Peng, H.; Li, J.; He, Y.; Liu, Y.; Bao, M.; Wang, L.; Song, Y.; and Yang, Q. 2018. Large-Scale Hierarchical Text Classification with Recursively Regularized Deep Graph-CNN. In *Proceedings of the 2018 World Wide Web Conference*, 1063–1072. International World Wide Web Conferences Steering Committee. ISBN 9781450356398.

Peng, H.; Li, J.; Wang, S.; Wang, L.; Gong, Q.; Yang, R.; Li, B.; Yu, P. S.; and He, L. 2021. Hierarchical Taxonomy-Aware and Attentional Graph Capsule RCNNs for Large-

Scale Multi-Label Text Classification. *IEEE Transactions on Knowledge and Data Engineering*, 33: 2505–2519.

Pennington, J.; Socher, R.; and Manning, C. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. Association for Computational Linguistics.

Sandhaus., E. 2008. The new york times annotated corpus LDC2008T19. Web Download. Linguistic Data Consortium.

Shimura, K.; Li, J.; and Fukumoto, F. 2018. 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*, 811–816. Association for Computational Linguistics.

Sinha, K.; Dong, Y.; Cheung, J. C.; and Ruths, D. 2018. A Hierarchical Neural Attention-based Text Classifier. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, 817–823. Association for Computational Linguistics. ISBN 9781948087841.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wang, B.; Hu, X.; Li, P.; and Yu, P. S. 2021. Cognitive structure learning model for hierarchical multi-label text classification. *Knowledge-Based Systems*, 218: 106876.

Wang, Z.; Wang, P.; Huang, L.; Sun, X.; and Wang, H. 2022. 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)*, 7109–7119. Association for Computational Linguistics.

Wehrmann, J.; Cerri, R.; and Barros, R. 2018. Hierarchical Multi-Label Classification Networks. In Dy, J.; and Krause, A., eds., *Proceedings of the 35th International Conference on Machine Learning*, volume 80, 5075–5084. PMLR.

Wu, J.; Xiong, W.; and Wang, W. Y. 2019. 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)*, 4354–4364. Association for Computational Linguistics.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In Bach, F.; and Blei, D., eds., *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, 2048–2057. PMLR.

Yang, P.; Sun, X.; Li, W.; Ma, S.; Wu, W.; and Wang, H. 2018. SGM: Sequence Generation Model for Multi-label Classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, 3915–3926. Association for Computational Linguistics.

Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; and Hovy, E. 2016. Hierarchical Attention Networks for Document Classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1480–1489. Association for Computational Linguistics.

Ying, C.; Cai, T.; Luo, S.; Zheng, S.; Ke, G.; He, D.; Shen, Y.; and Liu, T.-Y. 2021. Do Transformers Really Perform Badly for Graph Representation? In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P. S.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 28877–28888. Curran Associates, Inc.

Zhao, W.; Ye, J.; Yang, M.; Lei, Z.; Zhang, S.; and Zhao, Z. 2018. Investigating Capsule Networks with Dynamic Routing for Text Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3110–3119. Association for Computational Linguistics.

Zhou, J.; Ma, C.; Long, D.; Xu, G.; Ding, N.; Zhang, H.; Xie, P.; and Liu, G. 2020. Hierarchy-Aware Global Model for Hierarchical Text Classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1106–1117. Association for Computational Linguistics.