Developing Prefix-Tuning Models for Hierarchical Text Classification

Lei Chen and **Houwei Chou**
Rakuten Institute of Technology (RIT)
Boston, MA
{lei.a.chen,houwei.chou}@rakuten.com

Xiaodan Zhu
Ingenuity Labs Research Institute & ECE
Queen's University, Canada
xiaodan.zhu@queensu.ca

Abstract

Hierarchical text classification (HTC) is a key problem and task in many industrial applications, which aims to predict labels organized in a hierarchy for given input text. For example, HTC can group the descriptions of online products into a taxonomy or organizing customer reviews into a hierarchy of categories. In real-life applications, while Pre-trained Language Models (PLMs) have dominated many NLP tasks, they face significant challenges too—the conventional fine-tuning process needs to modify and save models with a huge number of parameters. This is becoming more critical for HTC in both global and local modelling—the latter needs to learn multiple classifiers at different levels/nodes in a hierarchy. The concern will be even more serious since PLM sizes are continuing to increase in order to attain more competitive performances. Most recently, prefix tuning has become a very attractive technology by only tuning and saving a tiny set of parameters. Exploring prefix turning for HTC is hence highly desirable and has timely impact. In this paper, we investigate prefix tuning on HTC in two typical setups: local and global HTC. Our experiment shows that the prefix-tuning model only needs less than 1% of parameters and can achieve performance comparable to regular full fine-tuning. We demonstrate that using contrastive learning in learning prefix vectors can further improve HTC performance.

1 Introduction

Hierarchical text classification (HTC) is a key task in many industrial applications. Typically, a large number of labels are defined and organized in a taxonomic tree. How to accurately and efficiently predict texts into label paths in the label hierarchies is an important capacity in high demand. For example, many e-commerce applications need to assign an online product to a path in the label hierarchy, e.g., *beverage→coffee→instant coffee* or *beverage→tea→oolong tea*. Identifying these label paths allows the information to be easily accessed by down-stream applications and human users.

In the past few years, Pre-trained Language Models (PLMs) have become a dominant solution for most natural language processing (NLP) applications. However, PLM models often contain a very large number of parameters, and the model sizes keep increasing, which can put a heavy burden on HTC applications. As an example, HTC often benefits from building a number of local models to fully utilize label hierarchies. Instead of training one model as in *global* HTC modelling, *local* HTC models rely on and leverage several inner classifiers (Peng et al., 2018). Figure 1 shows that when building a local HTC model for separating various types of drinks into the beverage category, the model sizes dramatically increase along with the increase of hierarchy levels. (More discussion about local and global HTC models can be found in Section 2).

Most recently, prefix tuning (Li and Liang, 2021; Lester et al., 2021) has become a very attractive technology by only tuning and saving a tiny set of parameters compared that of a fully fine-tuned model. Exploring prefix turning for HTC is hence highly desirable and has timely impact. In this paper, we investigate prefix tuning on HTC in two typical setups: local and global HTC. Our experiment shows that the prefix-tuning model only needs less than 1% of parameters and can achieve performance comparable to regular full fine-tuning. We demonstrate that using contrastive learning in learning prefix vectors can further improve HTC performance.

In brief, our contributions are summarized as follows:

- To the best of our knowledge, this is the first systematic study to develop prefix-tuning for HTC

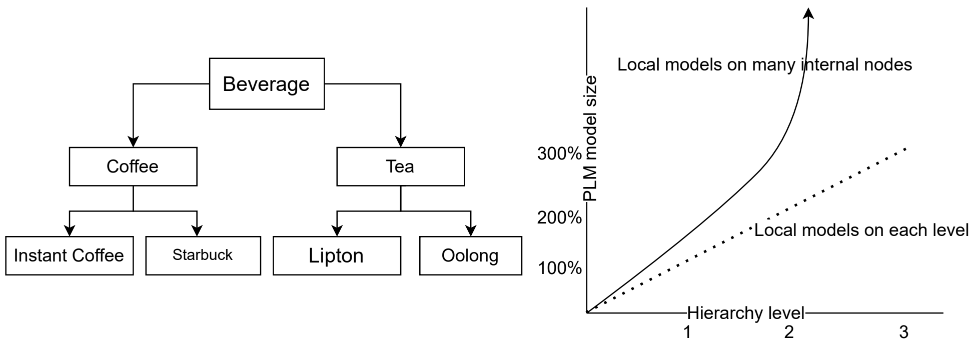

Figure 1: An illustration to show that local HTC model will face a size issue when using PLM models to be classifiers.

- Following local HTC modelling, we examine different architectures to leverage prefix vectors learned at different levels of label hierarchies and provide results about our best practice.

- In the global HTC strategy, we propose to add a self-training step built on a contrastive learning (CL) loss and this shows to improve performance.

- We provide detailed results on two HTC datasets and the analyses to show how the models work.

2 Related work

There are two major means of handling label hierarchies for HTC, i.e., the local and global approach (Zhou et al., 2020). The local approach builds a number of classifiers on different label levels or on many internal nodes in a label hierarchy but the global approach develops a single classifier to predict all labels that are flattened from the label hierarchy.

Shimura et al. (2018) developed convolution neural network (CNN) based local models at each level of label hierarchies and proposed to use the trained CNN at the higher level to initialize the CNN at a lower level. This *transferring* approach that considers inter-connections among the CNN models in a hierarchy showed to improve HTC performance. Regarding the global HTC, a straightforward method is flattening labels' hierarchical structure into a flat list and modelling the HTC simply to a multi-label classification task. Recently, a trending method is utilizing a structure encoder to retain the label hierarchy to better utilize mutual information among labels. (Zhou et al., 2020) used a structure encoder, either a tree LSTM or a graph convolution network (GCN), to consider labels' prior hierarchy information when learning label representations. PLMs have become a foundational paradigm on building various NLU tasks. For example, BERT (Devlin et al., 2019) has been applied to tackle the HTC task (Chen et al., 2021; Wang et al., 2022).

In parallel, contrastive learning (CL) has been found to be effective in providing high-quality encoders in a simple self-learning way. For example, in computer vision, SimCLR (Chen et al., 2020) uses the consistence between an anchor image and its transformed version and the in-consistence between the anchor and other instances in a batch (in-batch negative instances) to guide encoder training. Inspired by the success of SimCLR in computer vision, CL-based textual representation learning has become a hot research topic in NLP. SimCSE (Gao et al., 2021) uses dropout operations existing in Transformer (Vaswani et al., 2017) to provide self-augmentation and can learn effective text representations.

The CL training has been applied on the HTC task. (Chen et al., 2021) embeds both text inputs and labels (in a hierarchy) into an unified semantics space and solving the HTC via *vector matching*. When training the text encoder, a CL setup is used and it considers label hierarchy information when forming contrastive pairs. (Wang et al., 2022) uses a CL setup to train a high-quality text encoder. Label hierarchy is firstly encoded by a Graphormer (Ying et al., 2021) and the encoded label information is used to generate text variations for providing positive pairs in the CL.

Although fine-tuning PLM models enable many down-stream natural language understanding (NLU) tasks to achieve high performance, this paradigm faces a challenge in real deployment compared to other light weight models, e.g., CNN. Also, PLMs contain a large number of parameters and the model sizes have been exploding in recent years for reaching more competitive performance and the trend is continuing. When deploying the fine-tuned PLM models, all model parameters (updated in the fine-tuning process) need be stored. When many such PLM models need be stored, for example, for local HTC modelling, the required models sizes can be very large. To use PLMs in a more space-efficient way, previous efforts, such as fine-tuning only several top layers in a PLM or fine-tuning an adapter, are proposed (He et al., 2022). Unlike them, prefix-tuning (Li and Liang, 2021; Lester et al., 2021) only learns prefix vectors to trigger a PLM, which is frozen and cannot be tuned, to output the text representations fitting to the targeted domain better. In addition, (Liu et al., 2022) extended the prefix-tuning on NLU tasks by using prefix vectors on each PLM layer and dropped several components in a conventional prompt-tuning, e.g., verbalizer.

3 Exploring Effective Prefix Tuning for Hierarchical Classification

Let $\mathbf{x}$ denote the text input, Y a label hierarchy and y a specific category label path in Y. HTC aims to solve a multi-label categorization task: given textual input $\mathbf{x}$, HTC learns to predict possible label paths y in the hierarchy Y. As discussed above, when developing HTC in industrial applications, PLM-based models face model-size issues, which is becoming more serious as PLM sizes are continuing to increase. To tackle the challenge, we investigate soft prefix prompt (SPP) tuning on HTC. We propose to explore the models in two typical approaches. In Section 3.1 below, we explore a transferring approach to better train SPP vectors among different label levels in the local HTC modelling. Section 3.2 explores global HTC models in which we propose to add a CL-based self-training step.

3.1 SPP tuning considering hierarchical information

Figure 2 depicts two approaches of fine-tuning a PLM model for text classification. The left subfigure shows the conventional [CLS]-tuning, in which the [CLS] token is appended in front of the input text x. The entire text sequence goes through multiple Transformer layers in a PLM model. Built on that, the hidden output $h_{[CLS]}$ at the final layer serves as the representation for x. The $h_{[CLS]}$ passes through a linear classifier layer (denoted as CLF in Figure 2) to make predictions. Using the fine-tuning data, losses can be fed back into the model and all parameters in both the classifier head and the PLM model are accordingly tuned. Unlike that, the right subfigure highlights the process of (SPP) tuning (Liu et al., 2022), in which the entire PLM model is frozen and will not be updated during fine-tuning. For the embedding layer and each of PLM layers, tunable SPP vectors, which have a much smaller sizes compared to the PLM, are tuned to trigger the frozen PLM to output a more informative $h_{[CLS]}$ for prediction.

When using the SPP-tuning, the HTC model focuses on a set of SPP vectors. In the local HTC model, these SPP vectors on different locations/layers in a hierarchy may have some inter-constraints and therefore training them by considering their topological relationship in the hierarchy is our first consideration.

Specifically, Figure 3 depicts how we perform SPP-tuning on adjacent hierarchy layers. Subfigure (a) shows a basic solution in which SPP vectors on different levels of the hierarchy are trained independently without considering any inter-level connections. However, subfigure (b) shows that the trained SPP vector at a higher level is used to initialize a part of SPP vector at a lower level. The motivation is that knowledge learned in the upper layer prefix can help inform the low layer decision. In our study, we propose to assign lower-level SPP vectors longer than the SPP vectors at a higher level since the former needs handle more labels. In addition, we propose and investigate the architecture in subfigure (c) where the SPP vector at the higher level is transformed to a longer vector to initialize the lower-level SPP vector by using a fully-connected neural network.

3.2 Global model using contrastive learning when doing SPP-tuning

The other typical setup is investigating global HTC models. As shown in the right subfigure of Figure 2, when training SPP vectors, the loss after the classifier layer is used in supervised learning. Unlike the local modelling, here we do not transfer prefix among different layers. When develop the model, inspired by the success of using self-learning to

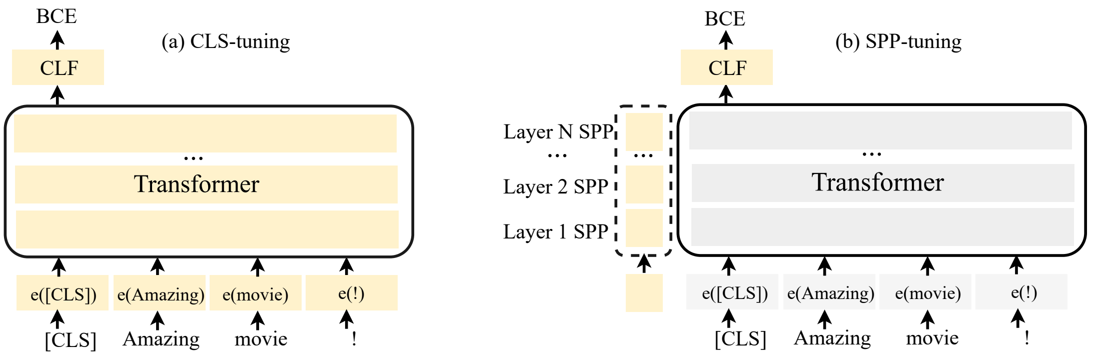

Figure 2: (a) shows conventional [CLS]-tuning for using PLM models. Note that all of parameters in a PLM need tuning and are shown in a light yellow color. In a contrast, (b) shows Soft Prefix Prompt (SPP) tuning, in which a frozen PLM model is used and only small-sized SPP vectors (on embedding input and each PLM layer) are tuned.

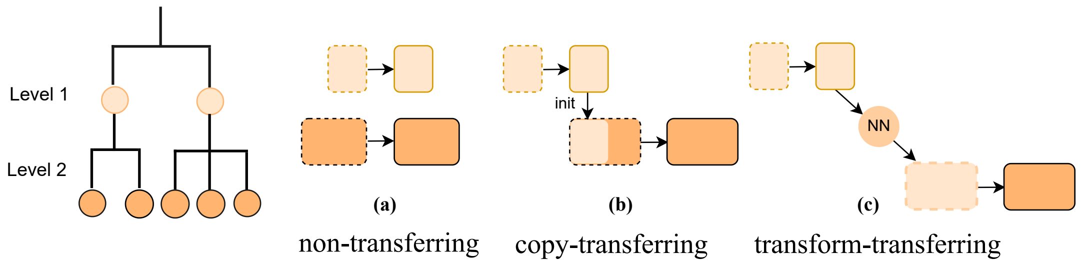

Figure 3: When solving HTC using a local model strategy, there are several ways to train prefix vectors for local models on each label level. (a) shows that vectors from upper and lower levels can be trained separately, (b) shows that lower level vectors can be initialized based on trained vectors at an upper level, and (c) shows that lower level vectors can be initialized based on the trained vectors at an upper level and go through a neural network (NN) transformation. Rectangles at the two levels refer to SPP vectors, from being initialized (enclosed in dash lines) to being fine-tuned (enclosed in solid lines).

learn proper representations, contrastive learning is found to be beneficial when being applied with SPP in the global modelling.

Specifically in our SPP-tuning setup, we follow the SimCSE (Gao et al., 2021) contrastive approach to feed inputs into a PLM model twice to obtain a data anchor and its positive pair.

For a text title $\mathbf{x}$, we append the SPP vector (V_{spp}) to [CLS]. Then we obtain a text representation $\mathbf{t}$ with a BERT encoder $BERT(*, d)$ where d is a dropout mask, and a projection function g, which uses a simple multiple layer perception (MLP) structure.

$$\mathbf{t} = g(BERT(V_{spp} : [CLS] : \mathbf{x}, d)) \quad (1)$$

To obtain a positive pair, SimCSE runs the same text title throughout the Transformer encoder pipeline with a different dropout mask d^+.

$$\mathbf{t}^+ = g(BERT(V_{spp} : [CLS] : \mathbf{x}, d^+)) \quad (2)$$

For the i^{th} text title, the training objective of SimCSE is as follows:

$$\mathcal{L}_i = -log \frac{exp(sim(\mathbf{t}_i, \mathbf{t}_i^+)/\tau)}{\sum_{j=1}^{N, j \neq i} exp(sim(\mathbf{t}_i, \mathbf{t}_j)/\tau)} \quad (3)$$

For a mini-batch of N text titles, where $sim(*, *)$ represents a similarity computation and τ is the temperature. The total loss computed by SimCSE, $\mathcal{L}_{simCSE}$, is an average among all text titles in the mini-batch, $\sum_{i=1}^{N} \mathcal{L}_i / N$. By running an optimization to keep reducing $\mathcal{L}_{simCSE}$, the SPP vectors on multiple layers of the BERT model can be tuned prior to applying the supervised fine-tuning. To the best of our knowledge, this is the first work to apply CL pre-training on an NLU task in SPP-tuning.

4 Experiments

4.1 Datasets and Evaluation

We perform our study on the widely used Web of Science (WoS) (Kowsari et al., 2017) and Amazon review dataset (McAuley et al., 2015; He and

Dataset	levels	Train size	Dev size	Test size	classes
WoS	2	33,070	7,518	9,397	141
Amazon Beauty	5	116,240	29,061	62,273	241

Table 1: Our experiments use both academic benchmark data set, WoS, which has been widely used in previous HTC research (Chen et al., 2021; Wang et al., 2022), and also an industry data from Amazon review dataset.

McAuley, 2016), focusing on the Beauty category for comparison and analysis. WoS contains abstracts of published papers from Web of Science and Amazon review contains titles of online products. For each instance in WoS, there is only a single label path. However, for each instance in Amazon Beauty, there could be more than one possible label paths. Regarding these two datasets, more statistic details are reported in Table 1. Similar to previous works, we measure the experimental results by using micro-F1 (denoted as mi-F1) and macro-F1 (denoted as ma-F1) to value performances per instance and per label respectively.

4.2 HTC models

We consider a variety of HTC models:

- CLS-tuning: A global model using a binary cross entropy (BCE) loss to train a HTC model as a multi-label text classifier.

- Local SPP-tuning: A local model consisting of two multi-label text classifiers trained with SPP. Between SPP vectors at the top and bottom label levels, there are three ways to train: (a) **non-transferring** refers to the basic strategy described in Section 3.1, training two sets of SPP vectors independently, (b) **copy-transferring** refers to using the trained SPP vector at the top level to initialize the corresponding portion in the longer SPP vector at the bottom level, and (c) **transform-transferring** refers to using a neural network to transform the SPP vector at the upper level to longer vector to initialize the SPP vector at lower level. Since the label hierarchy in the WoS data contains exactly two levels, we built local models on each level.

- Global SPP-tuning: a global model trained by using the SPP-tuning, and the BCE loss is used to train the SPP vectors

- Global SPP-tuning with CL: before training SPP vectors by the BCE loss, as described in Section 3.2, a contrastive learning (CL) self-learning step is used to better initialize SPP vectors.

For the PLM model, we used BERT-base provided in the Hugging Face's Transformer library (Wolf et al., 2020). The batch size is set to be 48. The optimizer is Adam with a learning rate of $1e^{-2}$ for the SPP-tuning and the learning rate of $2e^{-5}$ for CLS-tuning[1]. We implemented all above-mentioned models based on the source code[2] provided by (Liu et al., 2022) in PyTorch. We trained these models in an end-to-end way on the training set for up to 40 epochs and use an early stop if there was no any performance gain for consecutive 5 epochs on the development set. SPP vector length is an important hyper-parameter controlling SPP-tuning performance. Typically for simple NLU tasks, short SPP vectors, e.g., shorter than 20, could work sufficiently. Hence, we did a grid search among SPP vector lengths from 5 to 40 and found optimal SPP vector lengths for local and global models respectively. When conducting CL pre-training, we set the batch size to be 64 to maintain enough in-batch negative samples and used a temperature (τ) of 0.1. We conducted the CL pre-training for 10 epochs.

4.3 Results

Table 2 reports the result of comparing the three training strategies for building HTC models built on SPP-tuning. When using SPP-tuning to train a global model on the WoS data, we can find that by only using 0.46% of model parameters as used in CLS-tuning, we can achieve a performance even higher than that obtained by using the CLS-tuning. Among the three local HTC models based on SPP-tuning, the non-transferring approach yields the lowest performance, even worse than the result of using CSL-tuning. The transform-transferring approach works better than non-transferring and

[1](Liu et al., 2022) uses higher learning rate than what is used by CLS-tuning. We found this choice makes our SSP-tuning work.

[2]`https://github.com/THUDM/P-tuning-v2`

Model	Detail	Mi-F1	Ma-F1	Parameters(%)
CLS-tuning	global, BCE	85.89	79.22	100%
SPP-tuning	global, BCE, $\lvert V_{spp}\rvert = 30$	86.84	79.77	0.46%
SPP-tuning	local, non-transferring, $\lvert V_{spp}^{top}\rvert = 10$ $\lvert V_{spp}^{bottom}\rvert = 20$	86.84	79.12	0.46%
SPP-tuning	local, transform-transferring, $\lvert V_{spp}^{top}\rvert = 10$ $\lvert V_{spp}^{bottom}\rvert = 20$	86.93	79.33	0.46%
SPP-tuning	local, copy-transferring, $\lvert V_{spp}^{top}\rvert = 10$ $\lvert V_{spp}^{bottom}\rvert = 20$	87.24	79.98	0.46%
SPP-tuning	local, copy-transferring, $\lvert V_{spp}^{top}\rvert = 10$ $\lvert V_{spp}^{bottom}\rvert = 30$	87.34	80.09	0.66%

Table 2: HTC models on WoS dataset. By using a BERT-base, various fine-tuning methods, i.e., CLS-tuning, global model using SPP-tuning, and local models using SPP-tuning, are compared.

		WoS			Amazon Beauty		
Model	Detail	Mi-F1	Ma-F1	Parameters (%)	Mi-F1	Ma-F1	Parameters (%)
CLS-tuning	BCE	85.89	79.22	100.00%	87.49	63.38	100.00%
SPP-tuning	BCE	86.84	79.77	0.46%	88.00	61.36	0.50%
SPP-tuning	CL$\rightarrow$BCE	87.55	80.07	0.46%	88.14	62.07	0.50%

Table 3: Global HTC models on both WoS and Amazon datasets. When training by SPP-tuning, we proposed adding a CL pre-training stage and this turns out to improve HTC performance.

the best performance is from copy-transferring approach. When keep increasing V_{spp}^{bottom} from 20 to 30, the performance can be improved further. It shows that SPP vectors trained at a higher level need be used intact when initializing SPP vectors at lower levels. Also, both transferring approaches work better than non-transferring.

Table 3 reports the results of comparing the two types of training losses when training a global model based on SPP-tuning. We can see that on the two data sets, WoS and Amazon Beauty, the SPP-tuning only using the BCE loss is worse than the proposed model that leverages CL pre-training.

Note that when only using 0.5% of the parameters used in CLS-tuning, on the Amazon Beauty data with 241 labels, the SPP-tuning model has a performance comparable to CLS-tuning. A slight gain is actually observed on micro-F1, although macro-F1 has some drop from 63.38% to 61.36%. We show that after adding CL self-training prior to fine-tuning SPP vectors, macro-F1 is still lower than what we can obtain when using CLS-tuning. This is worth more investigations to evaluate SPP-tuning approach comprehensively, on labels with both sufficient and sparse training instances.

5 Conclusions and Future Work

HTC is a key task in many industrial applications. The conventional fine-tuning process needs to modify and save models that have a large number of parameters. This has become a more significant concern as PLM sizes are continuing to increase in the foreseeable future. In this paper, we investigate prefix tuning on HTC in two typical setups: local and global HTC. To the best of our knowledge, this is the first systematic study towards developing prefix-tuning for HTC in these typical architectures.

In local HTC modelling, we examine different architectures to leverage prefix vectors learned at different levels of label hierarchies and provide results about our best practice. We found that SPP vectors trained at a higher level can be utilized to initialize a portion of SPP vectors at a lower level of the hierarchy and such a vector transferring strategy is beneficial. For SPP vectors, using them intact works better than using their transformed version. In the global HTC strategy, we propose to add a self-training step built on a contrastive learning (CL) loss. On both WoS and Amazon datasets, such a CL pre-training is found to be helpful on improving model performance.

For future work, we will extend the current work to study long-tailed labels which is very common in many applications. Also, how to use labels' hierarchical information that can be represented by structural encoders is worth studying in SPP-tuning.

References

Haibin Chen, Qianli Ma, Zhenxi Lin, and Jiangyue Yan. 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)*, pages 4370–4379, Online. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a Unified View of Parameter-Efficient Transfer Learning.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Kamran Kowsari, Donald E. Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S. Gerber, and Laura E. Barnes. 2017. HDLTex: Hierarchical deep learning for text classification. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. 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)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 world wide web conference*, pages 1063–1072.

Kazuya Shimura, Jiyi Li, and Fumiyo Fukumoto. 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*, pages 811–816, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, \Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30:5998–6008.

Zihan Wang, Peiyi Wang, Lianzhe Huang, Xin Sun, and Houfeng Wang. 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)*, pages 7109–7119, Dublin, Ireland. Association for Computational Linguistics.

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. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888.

Jie Zhou, Chunping Ma, Dingkun Long, Guangwei Xu, Ning Ding, Haoyu Zhang, Pengjun Xie, and Gongshen Liu. 2020. Hierarchy-aware global model for hierarchical text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1106–1117.