Bert with Dynamic Masked Softmax and Pseudo Labeling for Hierarchical Product Classification

Li Yang[1], Shijia E[2], Shiyao Xu[1], and Yang Xiang[1]

[1] Tongji University, Shanghai, China
[2] Tencent, Shanghai, China
{li.yang,xushiyao,shxiangyang}@tongji.edu.cn
e.shijia@gmail.com

Abstract. Hierarchical product classification (HPC) aims to assign predefined product categories stored in a hierarchical structure to product instances. Categories at different levels of a product tend to have dependencies. However, most previous studies either decompose the original problem into a set of flat classification sub-problems or deal with all categories simultaneously, lacking a focus on the dependencies among different category levels. In this paper, we propose a BERT-based ensemble model to address the HPC challenge in SWC2020–MWPD Task 2[1]. We devise a masked matrix for each category level based on the hierarchical category structure, which can dynamically filter out the child categories unrelated to the current parent category and eliminate the negative effect of category inconsistency. Further through a two-level ensemble strategy and pseudo labeling, our team *Rhinobird* wins first place with a weighted-average macro-F1 score of 88.62 on the testing dataset. Our source code is publicly available on Github.[2]

Keywords: Hierarchical Product Classification · BERT · Dynamic Masked Softmax · Pseudo Labeling.

1 Introduction

Recent years have seen significant use of semantic annotations in the e-commerce domain, where online shops (e-shops) are increasingly adopting semantic markup languages to describe their products to improve their visibility. Hierarchical product classification is a fundamental but challenging task of product semantic annotation, where product instances are assigned to multiple levels of categories which are stored hierarchically. Automated product classification based on the product offer information made on the web has become an essential tool for searching, retrieving, and managing the products.

For the hierarchical classification problem, prior studies predict only the categories of the last level by reducing the problem into a flat multi-class problem [2]. Unfortunately, these flat-based approaches ignore the hierarchical cat-

[1] https://ir-ischool-uos.github.io/mwpd/index.html#task2

[2] https://github.com/AlexYangLi/iswc2020_prodcls

egory structure information. To this end, some works have considered the hierarchical structure, which can be categorized into two approaches: 1) Local approaches [4] generate a unique classifier for each parent node in the category hierarchy; 2) Global approaches [3,5] deal with all the levels of categories by using a single classifier. The local approaches suffer the inherited disadvantage that the number of sub-models grows exponentially concerning the number of category levels. This is especially problematic for neural-based models with a large number of parameters.

In this paper, we propose an end-to-end global approach that overcomes the problem of exploding models and explicitly considers the dependencies among different category levels. We use BERT [1] as the base model to generate a rich semantic product representation and predict the categories level by level, conditioned on a dynamic masked matrix obtained based on the hierarchical category structure. The masked matrix serves as an information filter that dynamically filters out the child categories unrelated to the current parent category, which contributes to addressing the category inconsistency problem. To enhance the generalization ability of our model, we adopt a two-level ensemble strategy, which combines the results of 17 different BERT models to make the final decisions. Furthermore, we utilize pseudo labeling, a semi-supervised method that uses the unlabeled data to enhance the performance. Our solution achieves a weighted-average macro-F1 score of 88.62 on the testing dataset of SWC2020–MWPD Task2.

2 Dataset

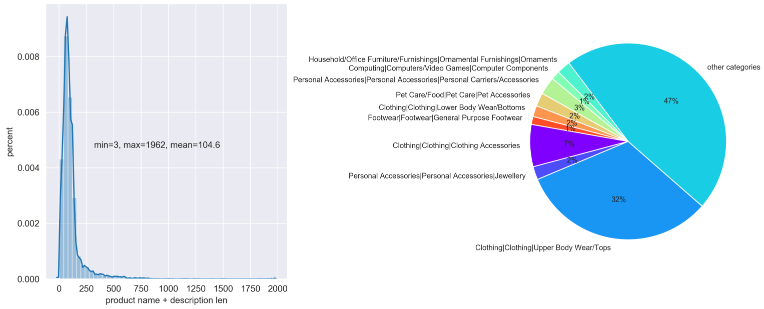

Fig. 1. Distribution of the length of product name and description, and the category labels of three levels on the training dataset.

For this challenge, the organizers have provided 10012 labeled products for training, 3008 labeled products for validation, and 3107 unlabeled products for testing. Each product instance is provided with its id, name, description, website-specific product category, and original web page URL. In this paper, we use only the name and description text to solve the product classification problem. The left sub-figure of Figure 1 shows the distribution of text length of product names and descriptions, and we can find that most of them are less than 250. The product labels are organized in three levels, of which the first level has 37 categories, the second level has 76 categories and the third level has 281 categories. We show the distribution of categories labels of three levels in the right sub-figure in Figure 1[3]. We can find that the category distribution is fairly imbalanced, which increases the difficulty of the task.

3 Model Description

In this section, we describe the details of our proposed approach. The overall framework and processing pipeline are shown in Figure 2, including the base model construction, model ensembling, and pseudo labeling for data augmentation.

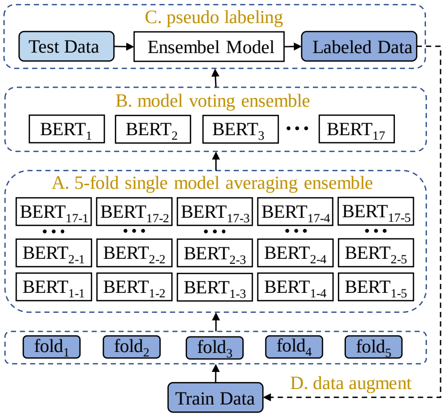

Fig. 2. The overall framework of our solution for hierarchical product classification.

3.1 Base Model

We use BERT as the base model in our solution. BERT is a language representation model designed to pre-trained deep bidirectional representations by jointly conditioning on both left and right context in all layers, which achieved state-of-the-art performance in several language understanding tasks.

In practice, we fine-tune the BERT model to generate a contextual product representation and then add three output layers on top of this representation to

[3] We only show the categories with top-10 frequencies for clarity, the “other” label accounts for the rest categories.

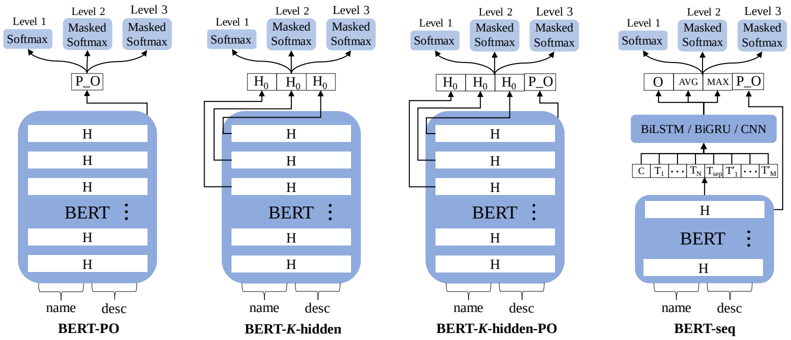

Fig. 3. Different ways of using BERT to generate a product representation.

jointly classify the products into the categories of three levels. For the input to BERT, we use the name and description text of products provided by the data. Instead of concatenating the name and description into one sentence, we regard them as one text pair and use a special symbol, "[sep]", to separate them before feeding into the BERT model. The intuition behind this is that the product name provides more concise and useful information about the product. We want to enhance the influence of the product name and expect that the model can distinguish them.

To generate rich semantic product representations, We make full use of the hidden states from the last or more hidden layers of BERT, resulting in 17 different BERT base models. Figure 3 presents four different ways:

(1) **BERT-PO** uses the pooler output of BERT as the product representation, which is the most common way to adopt BERT for classification problems.
(2) **BERT-K-hidden** concatenates the first hidden state from the last K hidden layers of BERT as the product representation. We range K from 1 to 5, resulting in 5 different models.
(3) **BERT-K-hidden-PO** concatenates the first hidden state from the last K hidden layers as well as the pooler output of BERT as the product representation. We range K from 1 to 5, resulting in 5 different models.
(4) **BERT-seq** uses the hidden states from the last hidden layer of BERT as the input of another sequence layer, and then concatenates the pooler output of BERT, with the last hidden output as well as the max-pooling and mean-pooling over the hidden states of sequence layer, as the final product representation. We try 5 different sequence layers: BiLSTM, BiGRU, CNN, BiLSTM+CNN, and BiGRU+CNN.

3.2 Dynamic Masked Softmax

Suppose the product representation generated by the BERT base model is denoted by R. We then feed it into three feed-forward neural layers, to compute the scores of categories for each level:

$$O^l = W^l R + b^l, \tag{1}$$

where W^l and b^1 are the weight matrix and bias term for the category level l. Hereafter, we can directly adopt a plain softmax layer to normalize the scores and select the category with the maximum probability. However, this approach chooses category independently for each level, ignoring the dependencies among different category levels, which can cause the problem of category inconsistency.

In this paper, we propose a category hierarchy based mask matrix to address the above problem by dynamically filtering out the child categories which are unrelated to the current parent category. Specifically, we take advantage of the pre-defined hierarchical category structure information and devise a mask matrix for each sub-level $M^l \in \{0,1\}^{N^{l-1} * N^l}$, where N^l is the total number of categories of the current level l and N^{l-1} is the amount of categories of the last level. In this matrix, each $M^l_{u,v} = 1$ indicates that the v-th category of level l is the child category of the u-th category of level $l-1$, otherwise $M^l_{u,v} = 0$. We then compute the normalized scores of categories of level l using the masked softmax function with M^l as follows:

$$P\left(y_v^l \mid s, \theta\right) = \frac{\exp\left(O_v^l\right) * M^l_{u,v} + \exp(-8)}{\sum_{v'=1}^{N} \exp\left(O_{v'}^l\right) * M^l_{u,v'} + \exp(-8)}, \tag{2}$$

where $P\left(y_v^l \mid s, \theta\right)$ is the probability of assigning the v-th categories of level l to the product with sentence s by the model with parameter θ, and u is the index of the parent category. Notably, the probability of the categories unrelated to the current parent category will be fairly small, and therefore, their negative effect can be eliminated.

With this design, we successfully introduce the hierarchical category structure information to the model. Besides, by this filtering mechanism, we can reduce the number of categories to be classified for each sub-levels and classify dynamically according to the predicted parent category, which can ensure the consistency of categories between different levels.

3.3 Model Ensemble

To combine different single models, we adopt a two-level ensemble strategy. In the first level, we first combine the original training and validation dataset and split them into five folds by using the cross-validation technique. For every single model that we design above, we train it for five times by sequential choosing one fold for validation and the rest four folds for training. We then average the probability outputs from these five single models with the same model architecture but trained on a different dataset. In the second level, we apply the voting ensemble strategy to the 17 averaged ensemble single models, by choosing the most voted category as the final prediction. With the two ensemble strategy, we can generate a robust model with better generalization ability than any single model.

3.4 Pseudo Labeling

To further improve the performance, we utilize pseudo labeling, a simple yet effective semi-supervised method that allows us to make full use of the unlabeled data. To be specific, after training the single models with the labeled data and ensembling them, we use the ensemble model to predict labels for the unlabeled testing data. Together with the pseudo-labeled data and training data, we then retrained the single models again and thus obtain a new ensemble model. Pseudo labeling can be regarded as an effective way of data augmentation, which can alleviate the over-fitting problem by increasing the amount of training data. Since the ensemble model performs better than any of the single models in most cases, the pseudo-labels predicted by the ensemble model can correct the bias made by the single models, thus leading to performance improvements.

4 Experiments

4.1 Experimental Setup

Evaluation Metric For the challenge, a weighted-average macro-F1 score (WAF1) will be calculated over all the categories for each classification level. Then the average of the WAF1 of the three levels will be calculated and used as the evaluation metric to rank the participating systems.

Parameter Settings Our proposed approach is implemented by Tensorflow. We use the *BERT-Base-Uncased* version of Google's pre-trained BERT models[4] for fine-tuning, which has 12 transformer layers, 12 attention heads, and 768 hidden sizes, and is trained on lower-cased English text. We use Adam as the optimizer and set the initial learning rate to be 2e-5. The batch size is 32. All the hidden states of BiLSTMs and BiGRU and feature maps of CNNs used in the sequence layer have the same dimension as BERT's. For training every single model, early-stopping is applied to avoid over-fitting: training will be stopped when no performance improvement is observed on the validation dataset after three epochs. We then store the single model with the best performance on the validation dataset.

4.2 Experimental Results of single models

We first present the experimental results of the single models in Table 1. Due to the limited space, we only show the performance of the best single model, BERT-5-hidden, which concatenates the first hidden state of the last five hidden layers as the product representation. As indicated from the table, the best performance among 17 BERT models can achieve a weighted-average macro-F1

[4] `https://storage.googleapis.com/bert_models/2020_02_20/uncased_L-12_H-768_A-12.zip`

Table 1. Performance of the single models on the testing dataset.

Model	**Performance**			
	Level 1	**Level 2**	**Level 3**	**Avg.**
BERT-5-hidden	**0.9027**	**0.8913**	**0.8250**	**0.8730**
BERT-5-hidden$_{\text{name}}$	0.8896	0.8835	0.8166	0.8633
BERT-5-hidden$_{\text{desc}}$	0.8794	0.8724	0.8020	0.8513
BERT-5-hidden$_{\text{sent}}$	0.8836	0.8827	0.8220	0.8661
BERT-5-hidden$_{\text{w/o mask}}$	0.8967	0.8897	0.8214	0.8693

Table 2. Performance of the ensemble models on the testing dataset.

Model	**Performace**			
	Level 1	**Level 2**	**Level 3**	**Avg.**
BERT-ensemble	**0.9108**	**0.9011**	**0.8468**	**0.8862**
BERT-ensemble$_{\text{w/o mask}}$	0.9111	0.8990	0.8413	0.8838
BERT-ensemble$_{\text{w/o pseudo}}$	0.9069	0.8968	0.8388	0.8808

score of 0.8730. To gain a better understanding of the impact of the BERT's input and dynamic masked softmax, we further perform an ablation study.

The difference between the 1st to 4th single models is the input they are trained with, where BERT-5-hidden$_{\text{name}}$ and BERT-5-hidden$_{\text{desc}}$ are the variants which only use the product's name or description as input, while BERT-5-hidden$_{\text{sent}}$ uses both name and description as input but concatenates them as one sentence. As shown in Table 1, BERT-5-hidden$_{\text{desc}}$ is inferior to BERT-5-hidden$_{\text{name}}$. It is because the description of a product contains more noise (*e.g.*, misspelled words, useless URLs, etc.), while the name can convey more concise and accurate information about a product. Compare to BERT-5-hidden$_{\text{name}}$ and BERT-5-hidden$_{\text{desc}}$, BERT-5-hidden$_{\text{sent}}$ achieves better performance, which demonstrates using both name and description as input can provide more useful and complementary semantic information for product classification. BERT-5-hidden distinguishes the name and description by regarding them as one text pair when adopting BERT to model them, thus performs better than BERT-5-hidden$_{\text{sent}}$.

Furthermore, we observe performance degradation when eliminating the use of dynamic masked softmax (BERT-5-hidden$_{\text{w/o mask}}$), highlighting the importance of incorporating hierarchical category structure information for multi-levels classification.

4.3 Experiment results of the ensemble model

Table 2 shows the experimental results of the ensemble models. Compared to the single models, our ensemble model brings a significant performance gain, which achieves a weighted-average macro-F1 score of 0.8662 and is ranked the first among all the participating systems. This illustrates the utility of our two-level ensemble strategy. Removing the use of dynamic masked softmax (BERT-ensemble$_{\text{w/o mask}}$) degrades the model's performance, which is consistent with the observation in the single models. We also develop another variant that does not utilize the pseudo-labeled data to retrain the single models, which we denote BERT-ensemble$_{\mathrm{w/o\,pseudo}}$. We can see from Table 2 that the ensemble model achieves worse performance without pseudo labeling. This strongly verifies that pseudo labeling plays a great role in enhancing the generalization ability of the model.

5 Conclusion

In this paper, we propose a BERT-based ensemble model for the HPC challenge of SWC2020–MWPD Task2. In our solution, we construct 17 different single BERT models to generate rich semantic product representations for classification and then adopt a two-level ensemble strategy to combine their predictions. To consider the hierarchical category structure, we devise a masked matrix for each category level that can dynamically filter out the child categories unrelated to the parent category. Besides, we apply pseudo labeling to make full use of the unlabeled test data for better performance. Our team *Rhinobird* win first place with a weighted-average macro-F1 score of 88.62 on the testing dataset.

Acknowledgments This work was supported by the National Key Research and Development Project of China (2019YFB1704402), and the 2019 Tencent Marketing Solution Rhino-Bird Focused Research Program.

References

1. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: 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. pp. 4171–4186 (2019)
2. Fall, C.J., Törcsvári, A., Benzineb, K., Karetka, G.: Automated categorization in the international patent classification. SIGIR Forum **37**(1), 10–25 (2003)
3. Huang, W., Chen, E., Liu, Q., Chen, Y., Huang, Z., Liu, Y., Zhao, Z., Zhang, D., Wang, S.: Hierarchical multi-label text classification: An attention-based recurrent network approach. In: Proceedings of the 28th ACM CIKM International Conference on Information and Knowledge Management. pp. 1051–1060 (2019)
4. Kowsari, K., Brown, D.E., Heidarysafa, M., Jafari Meimandi, K., Gerber, M.S., Barnes, L.E.: Hdltex: Hierarchical deep learning for text classification. In: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA). pp. 364–371 (2017)
5. Sinha, K., Dong, Y., Cheung, J.C.K., Ruths, D.: A hierarchical neural attention-based text classifier. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 817–823 (2018)