KDD CUP 2022 MULTICLASS PRODUCT CLASSIFICATION: TEAM MetaSoul SOLUTION

Zhichao Feng
Beijing University of Posts and Telecommunications
Beijing, China
fengzc@bupt.edu.cn

Jiawei Lu
Beijing University of Posts and Telecommunications
Beijing, China
lujiawei@bupt.edu.cn

Junwei Cheng
Beijing University of Posts and Telecommunications
Beijing, China
cjwse205@bupt.edu.cn

Ke Hou
Beijing University of Posts and Telecommunications
Beijing, China
houke1237@bupt.edu.cn

Kaiyuan Li
Beijing University of Posts and Telecommunications
Beijing, China
tsotfsk@bupt.edu.cn

Pengfei Wang
Beijing University of Posts and Telecommunications
Beijing, China
wangpengfei@bupt.edu.cn

Yadong Zhu
Beijing Shuyuanling Technology Co., Ltd.
Beijing, China
sunkai@dmetasoul.com

ABSTRACT

Multi-product classification is a task of Amazon's ESCI challenge, which gives queries and corresponding lists of products and requires contestants to classify each product as an exact, substitute, complementary, or irrelevant match for the query. Due to the uneven distribution of the product's class and the noisy information in the results, it's hard to classify the products into different relations. While in this paper, we introduce our solution for this multi-product classification task. We firstly build a single tower bert-based model to classify the relationship between query and product, then utilize many tricks, including adversarial method,multi-sample dropout, and weight averaging evaluation method, to enhance the model's robustness or accuracy. Also, we use the weighted average method to ensemble the model parameters of different training steps together first and then the output of different models as the final result. Our solution achieves an overall Micro-F1 of 0.8207, which wins fifth place in the final leaderboard[1].

KEYWORDS

BERT, Query-Product Classification, Multilingual, Data Challenge

ACM Reference Format:
Zhichao Feng, Jiawei Lu, Junwei Cheng, Ke Hou, Kaiyuan Li, Pengfei Wang, and Yadong Zhu. 2022. KDD CUP 2022 MULTICLASS PRODUCT CLASSIFICATION: TEAM MetaSoul SOLUTION. In *KDD Cup 2022 Workshop: ESCI Challenge for Improving Product Search, August 17, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 4 pages.

[1]Our code is publicly available at https://github.com/guijiql/kddcup2022

KDDCup '22, August 17, 2022, Washington, DC, USA

1 INTRODUCTION

In this paper, we focus on the second task of the KDD Cup, a multi-class product classification task [9] which aims to predict the relationship between the query and the product retrieved for this query using the query statement and the product metadata. We need to address the noisy information in the results, the difficulty of understanding the query intent, and the too much variety of the items to finally solve this problem. To achieve the target, we utilized single-tower models based on Bert for classification. Specially, we used the multilingual versions of DeBERTa and RoBERTa as backbones. We also added several tricks and post-process models by model parameters ensemble on this basis.

The outline of the rest of the paper is as follows. First, we describe the methods we used in the challenge, including model methods, training methods, evaluation methods and post-processing. Next, we discuss the performance of our methods. Finally, we draw a conclusion through discussing our empirical findings.

2 METHOD

In this part, we introduce our methods in detail. The structure of our model is shown in Fig 1. Firstly, we perform word tokenization on the preprocessed query and product text. Then we put them into the bert to get embedding information. At this part, we also use several kinds of bert and we'll introduce them later. Finally we use the [CLS] token of bert with tricks to classify the input data. Our models will be introduced from three following aspects: model method, training method and evaluation method.

2.1 Model Method

DeBERTRa V3. DeBERTa[5] improves the BERT class models using disentangled attention and enhanced mask decoder. To go further, DeBERTa V3 uses the ELECTRA-Style pre-training method which replace MLM task with RTD(Replaced Token Detect) task

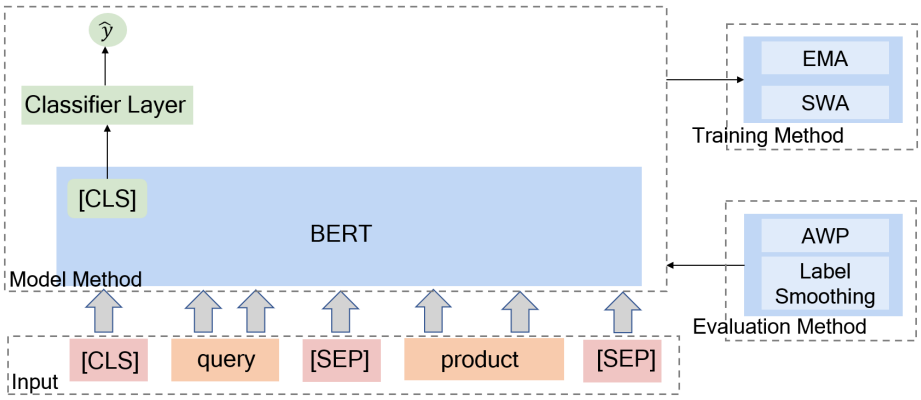

Figure 1: The structure of our model. Our backbone is a single-tower model. On the right of the figure are some of the tricks we tried. The output of classifier layer is probability of four classes.

accompanied by Gradient Disentangled Embedding Sharing and outperforms initial DeBERTa on many downstream tasks. We use mDeBERTa V3 base which comes with 12 layers and a hidden size of 768 to encode the multilingual query and product information, we also use DeBERTa V3 large which comes with 24 layers and a hidden size of 1024 to encode the English information.

XLM-RoBERTa. XLM-RoBERTa[1] model fine-tunes not only with traditional MLM(Masked Language Modeling) method, but also with the CLM[2](Casual Language Modeling) which uses the preceding words to predict next word's probability and TLM[2](Translation Language Modeling) to guide model aligning the representation among multi-languages. Besides, it also uses sharing sub-word dictionary to increase the tokens distributed to low-resource language and ease the prejudice of high resource language. We use the XLM-RoBERTa which comes with 12 layers and a hidden size of 768 to encode the multilingual information.

Multi-Sample Dropout. Multi-sample dropout[6] is an enhanced dropout technique that can accelerate training and improve generalization comparing to the original Dropout. It creates multiple dropout samples which calculate loss separately and use the average of the previous step to obtain the final loss. Multi-sample dropout can implement without much extra cost by sharing weights among the duplicating layers after the dropout layer.

2.2 Training Method

Adversarial Weight Perturbation. In the training phase, we utilized adversarial training. Adversarial training is common in NLP tasks. Usually, it's used to perturb the input or model parameters to construct adversarial examples and improve the robustness of the model during training. This method can also be considered as a regularization method to reduce overfitting and improve the generalization[3]. In this challenge, we tried adversarial weight perturbation(AWP) [10]. AWP perturbs both input and model parameters, and constrains the size of the perturbation to the times of parameter itself. The AWP formula we used is shown as equation 1:

$$w_i \mapsto w_i + \delta w_i = w_i + \gamma \frac{\nabla_i}{\|\nabla_i\|} \|w_i\|, \quad |\delta w_i| \leq \epsilon |w_i| \tag{1}$$

where w_i denotes the i-th parameter of model, ∇_i denotes the gradient of it.

Label Smoothing. To address the problem of overfitting and overconfidence, we'll introduce the Label Smoothing method. One-hot encoded label, commonly used, is not calibrated and always owns higher predicted probabilities, making the model less adaptive and more confident about its predictions. While Label Smoothing introduces noise for the labels to solve this problem, it regularizes a classification model with k output values by replacing the hard 0 and 1 classification targets with targets of $\frac{\epsilon}{k-1}$ and $1-\epsilon$ respectively[8].

2.3 Evaluation Method

Exponential Moving Averages. Exponential Moving Averages (EMA) [4] is a type of moving average method, which applies more weight to the most recent data pints than those which happened in past. In other words, it is likely to give more importance to the last experience or memories than to older ones, the formula of EMA is as Equation 2.

$$v_t = \beta \cdot v_{t-1} + (1-\beta) \cdot \theta_t \tag{2}$$

θ_t represents the model weights at time t, v_t represents the average of previous θ before t which is also called shadow weights and β represents the weighted weight. In the process of gradient descent, the shadow weight v_t will be maintained but not participate in the training. The basic assumption is that the model weight will wobble at the actual optimum point in the last n steps. Thus we take the average of the last n steps to make the model more robust.

Stochastic Weight Averaging. Stochastic Weight Averaging(SWA) [7] uses simple averaging of multiple points along the trajectory of SGD, with a cyclical or constant learning rate to achieve a better optimization result.Intuition for SWA comes from that the local minimum generated at the end of each learning rate cycle tend to

Table 1: Summary of the sampled dataset, including the number of unique queries, the number of judgements, and the average number of judgements per query.

Language	#Queries	#Judgements	Avg. Depth
English	7,479	140,220	18.7
Spanish	1,143	26,422	23.1
Japanese	1,378	33,446	24.3
US+ES+JP	10,000	200,088	20.0

accumulate on the edge regions of the loss surface and the loss value on these edge regions is small. By averaging several such points, it is quite possible to obtain an even lower loss, global universal solution

Instead of ensemble several models, two models are used to achieve SWA. The first model(w_{SWA}) stores the running average of model weights and the second model(w) traverses the weight space and explores it by using a cyclical learning rate schedule. At the end of each learning rate cycle, the current weights of the second model will be used to update the weight of the running average model by taking weighted mean between the old running average weights and the new set of weights from the second model.

$$w_{SWA} \leftarrow \frac{w_{SWA} \cdot n_{models} + w}{n_{models} + 1} \tag{3}$$

By following this approach, we only need to train one model and store two models in memory during training. For prediction, we just need to predict on the second modelw_{SWA}.

2.4 Post Processing

Inspired by EMA, we consider it is beneficial to maintain moving averages of the trained model parameter, which belongs to different training steps. When most of the model parameters are generally consistent within a time window, it indicates that the parameters around this place are relatively confident, so the moving average of the parameter will bring significant benefits to the stability and accuracy of the model. So we use the weighted averages of last several checkpoint model parameters instead of the last trained values [7].

3 EXPERIMENT

3.1 Setup

During the early stage, we trained our models on 10,000 randomly sampled queries of official training dataset, which is called sampled dataset, to accelerate the verification of our ideas. The summary of sampled dataset is shown in Table 1. It can be seen that the proportion of queries across different languages and the respective average depth are about the same as full dataset, which is shown in Table 2. The sampled dataset is divided into two parts: training and valid sets with 80% and 20% percentages.

For classification, we used microsoft/mdeberta-v3-base [2] (short for mDeBERTa), microsoft/deberta-v3-large [3] (short for DeBERTa-Large) and joeddav/xlm-roberta-large-xnli [4] (short for XLM-RoBERTa)

[2]https://huggingface.co/microsoft/mdeberta-v3-base
[3]https://huggingface.co/microsoft/deberta-v3-large
[4]https://huggingface.co/joeddav/xlm-roberta-large-xnli

Table 2: Summary of the full dataset, including the number of unique queries, the number of judgements, and the average number of judgements per query.

Language	#Queries	#Judgements	Avg. Depth
English	68,139	1,272,626	18.7
Spanish	10,624	249,721	23.5
Japanese	12,687	312,397	24.6
US+ES+JP	91,450	1,834,744	20.1

Table 3: Performance of mDeBERTa with different learning rates on sampled dataset.

learning rate	micro-F1(%)
1e-6	**74.11**
5e-6	74.05
2e-5	73.32

from HuggingFace. Among them, DeBERTa-Large is only for English queries. The other two models are for all languages. The hyperparameters used in the experiment are as follows: the learning rate of mDeBERTa ranges in [1e-6, 5e-6, 2e-5], and the learning rates of Deberta-Large and XLM-RoBERTa are 5e-6. We used the AdamW optimizer in all experiments, 6 epochs, and evaluated models for four times per epoch.

In the submission stage, we used full official training dataset and divided it by the same ratio. For mDeBERTa, we used a fixed learning rate 5e-6, and other settings are the same as the early stage.

3.2 Performance on Sampled Dataset

In this part, we show the results of mDeBERTa model on sampled dataset. Although this isn't the exact result, it could still show the trend that whether the method works.

The performance of mDeBERTa model with different learning rates is summarized in Table 3. We find that the learning rate has a great influence on the model performance. Though the micro-F1 under learning rate 1e-6 is slightly higher, we still used the learning rate 5e-6 in the following experiments considering the rate of convergence.

The effect of different tricks on the model is shown in Table 4 . It is clearly shown that AWP can get nearly 0.9% improvement through adversarial training, while it also brings the problem of slower training speed. As for the evaluation methods, SWA and EMA brings an improvement of 0.1% and can also accelerate model fitting. With SWA or EMA, model can fit in one epoch, which is much more efficient than 3 epochs of ordinary training. What's more, multi-sample dropout get over 0.2% improvement, while label smoothing get over 0.1%improvement,and both of them don't have much effect on speed.

3.3 Offline Performance on Full Dataset

With full data, we used three base models with different model structures or pre-training data. The results are shown in Table 5. The optimal single model we got was mDeBERTa with AWP. Although EMA and SWA were positive when used separately, they

Table 4: Performance of models with different tricks on sampled dataset.

model	micro-F1(%)
mDeBERTa	74.05
mDeBERTa + EMA	74.15
mDeBERTa + SWA	74.18
mDeBERTa + AWP	74.99
mDeBERTa + Multi-Sample Dropout	74.26
mDeBERTa + Label Smoothing	74.16

Table 5: Offline micro-F1(%) of models on full dataset. SM represents single model and MPE represents model parameters ensemble.

model	SM	MPE
mDeBERTa + AWP	76.37	76.41
DeBERTa-Large(only English)	77.3	77.4
XLM-RoBERTa + multidropout	75.55	75.76

Table 6: Online performance of models on full dataset. Ensemble consist of the three models in Table 5.

model	Public micro-F1(%)	Private micro-F1(%)
mDeBERTa + AWP	81.48	81.72
Ensemble	81.82	82.07

might be negative when combined with multi-dropout or label smoothing. Similarly, limited by the resources, multi-dropout and label smoothing can't be used at the same time. So we ended up not using them. Moreover, we simply added the parameters of the two best performing model checkpoints by weight to get model parameters ensemble. Our results showed that it led to different extents of improvement.

3.4 Online Performance on Full Dataset

In code submission stage, we used the three models mentioned in Table 5. Limit by online inference time, we used DeBERTa-Large for English queries prediction, XLM-RoBERTa for Spanish and Japanese queries prediction, and mDeBERTa for all queries prediciton. We took the weighted summation of predicted scores as our final prediction. The ensemble result is shown in Table 6. Our approach ranked 5th at last.

4 CONCLUSION

In this paper, we fully introduce our solution for the multi-product classification task. And we find that multilingual bert-based model can encode the product and query text well. Moreover, adversarial training method can obviously enhance our model's robustness and improve the prediction accuracy. Weighted average method is deployed to ensemble different training steps' parameters firstly and then the prediction of different models. Combining all those together, our model effectively alleviate the multi-product classification problem.

REFERENCES

[1] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *ACL*. 8440–8451.

[2] Alexis Conneau and Guillaume Lample. 2019. *Cross-Lingual Language Model Pretraining*. Curran Associates Inc., Red Hook, NY, USA.

[3] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *ICLR*.

[4] Seng Hansun. 2013. A new approach of moving average method in time series analysis. In *CoNMedia*. 1–4.

[5] Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv* (2021).

[6] Hiroshi Inoue. 2019. Multi-Sample Dropout for Accelerated Training and Better Generalization. *CoRR* (2019).

[7] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. 2018. Averaging Weights Leads to Wider Optima and Better Generalization. In *UAI*. 876–885.

[8] Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help?. In *NeurIPS*. 4696–4705.

[9] Chandan K. Reddy, Lluís Màrquez, Fran Valero, Nikhil Rao, Hugo Zaragoza, Sambaran Bandyopadhyay, Arnab Biswas, Anlu Xing, and Karthik Subbian. 2022. Shopping Queries Dataset: A Large-Scale ESCI Benchmark for Improving Product Search.

[10] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. 2020. Adversarial Weight Perturbation Helps Robust Generalization. In *NeurIPS*.