Second place solution of Amazon KDD Cup 2022: ESCI Challenge for Improving Product Search

Xiaolei Qin
qinxiaolei@corp.netease.com
Netease Games AI Lab
Hangzhou, China

Nan Liang
liangnan@corp.netease.com
Netease Games AI Lab
Hangzhou, China

Hongbo Zhang
zhanghongbo@corp.netease.com
Netease Games AI Lab
Hangzhou, China

Wuhe Zou
zouwuhe@corp.netease.com
Netease Games AI Lab
Hangzhou, China

Weidong Zhang
zhangweidong02@corp.netease.com
Netease Games AI Lab
Hangzhou, China

ABSTRACT

How to improve the search experience is a hot topic that has been studied for decades in both academia and industry. KDD Cup 2022 ESCI Challenge organized by Amazon provides a large dataset with four categories of ESCI to improve semantic matching of queries and products. In this paper, we present our solution on the three tasks in this competition. Since the dataset is multilingual, we adopt DeBERTa-v3-large, mDeBERTa-v3-base, RoBERTa-large and XLM-RoBERTa-large as our basic models, and train a model for each language separately. In task1, we also train multilingual models to increase diversity and use an ensemble for these models. We mainly focus on task1 and try to generalize the solution to task2 and task3. Using our method, our team achieved 0.9036 on the private test of task1 and won the second place. Due to time and computation resource constraints, we didn't do much optimization work for task2 and task3, and won 8th and 6th respectively.

KEYWORDS

KDD Cup, Language Model, Product Search, Semantic Matching, Multilingual

ACM Reference Format:
Xiaolei Qin, Nan Liang, Hongbo Zhang, Wuhe Zou, and Weidong Zhang. 2022. Second place solution of Amazon KDD Cup 2022: ESCI Challenge for Improving Product Search. 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 INTRODUCTION

Amazon is one of the most popular worldwide online shopping platforms, and the Search System plays an important role in users' online shopping experience. Improving the relevance of search results can significantly improve the customer experience and their engagement with search. In recent years, with the development of deep learning and natural language processing, text semantic matching has made significant progress. However, there are still some challenges that need to be further addressed. Most challenges are reflected in this year's competition hosted by Amazon – the KDD Cup 2022 ESCI Challenge.

In this challenge, the organizer provides the "Shopping Queries DataSet" [11], a large dataset of difficult search queries and products with ESCI relevance judgements(Exact, Substitute, Complement, Irrelevant). The dataset is multilingual, including English, Japanese and Spanish. Given a user specified query, the tasks are to rank the products by semantic matching(task1), classify each product as being an Exact, Substitute, Complement or Irrelevant match(task2), and identify the substitute products(task3).

Our work is organized as follows. In Section 2, we summarize the related works. Section 3 details our methods that achieve competitive results in all three tasks. In Section 4, we present some attempts we have made. Although they are not used in the final submission due to different reasons, they may provide some useful insights.

2 RELATED WORK

2.1 Pretrained Language Model

Large Pretrained Language models have recently become mainstream in the area of natural language processing. Most of them use self-supervised learning to learn the deep semantic meaning of words and contexts. After pretraining, the model can be easily adapted to downstream tasks and achieve huge improvements. BERT [4] is the first bi-directional pretrained language model. After that, many different pretrained models have been proposed. In this work, we adopt RoBERTa [10] and DeBERTa [7, 8] and their multilingual versions as our text encoder.

RoBERTa RoBERTa-large model is pretrained on the reunion of five English datasets using a masked language modeling (MLM) objective and consists of 24-layer, 1024-hidden size, 16-heads. It optimizes some hyper-parameters and pretraining objective, these changes sharply enhance the model performance in all tasks as compared to BERT.

XLM-RoBERTa [3] XLM-RoBERTa-large model is the multilingual version of RoBERTa and is pretrained on 2.5TB of filtered CommonCrawl data containing 100 languages(CC100).

DeBERTa DeBERTa focuses intensively on positional encoding and improves the BERT and RoBERTa models using disentangled attention and enhanced mask decoder. The DeBERTa-V3 model, using ELECTRA-style pretraining [2] with Gradient Disentangled Embedding Sharing, significantly improves the model performance on downstream tasks and consists of 24-layer, 1024-hidden size, 304M parameters.

mDeBERTa mDeBERTa is the multilingual version of DeBERTa and is pretrained with CC100 multilingual data which is the same as XLM-RoBERTa.

2.2 Bi-Encoder & Cross-Encoder

The key to this challenge is to measure the semantic similarity between user query and products. Bi-Encoder and Cross-Encoder are two different methods to do so. Bi-Encoders produce for a given text a semantic embedding and then calculate the similarity between texts by cosine similarity. In contrast, for a Cross-Encoder, both texts are input into the model simultaneously, and the similarity is directly output after sufficient interaction. Generally, Bi-Encoders are used as the retrieval to get a candidate set quickly from a large dataset, and Cross-Encoders are then used as the re-ranker to score the relevancy of all these candidates with higher accuracy. In this competition, there is already a fixed candidate set for each query, so we adopt Cross-Encoder for better results.

2.3 Ensemble

Ensemble [6] is one of the most powerful techniques in practice to improve the performance of deep learning models. By simply averaging the output of a few independently trained neural networks over the same training data set, it can significantly boost the prediction accuracy over the test set comparing to each individual model [1].

3 APPROACH

3.1 Dataset

There are different versions of dataset for the three tasks. Task2 and task3 share the same larger dataset and task1 is a reduced version of what is deemed to be "easy" queries. Machine Learning algorithms generally benefit from more data, therefore, when training task1, we also added the data of task2 and task3.

We didn't do much data preprocessing, but it is worth mentioning that we found that some of the product brand and color name have already appeared in the product title, the brand name basically appears at the beginning of the title, and the color name appears at the end. Accordingly, we concatenate the product title, brand and color name to generate new title in this way if title or color name does not appear in title. In addition, we removed the html symbols in the product description by regular expression.

3.2 Model Architectures

The overall architecture of our method is shown in Figure 1. The first step is to obtain semantic representation of query and product by fine-tuning a pretrained language model where we use RoBERTa-large and DeBERTa-v3-large for English, mDeBERTa-v3-base and XLM-RoBERTa-large for Japanese and Spanish. For task1, we also use Spanish and Japanese data to train a mDeBERTa-v3-base model and use the three languages data to train a xlm-RoBERTa-large model. Then we feed the last hidden states of the pretrained models into a Bi-LSTM layer. Finally, the output of Bi-LSTM is fed into the dense layer to get the prediction.

3.3 Train Strategies

3.3.1 5-fold cross-validation. We train all the models by 5-fold cross-validation, but we do not use all the 5-fold results in the final submission due to the limitation of the online submission system (the inference time and repository size).

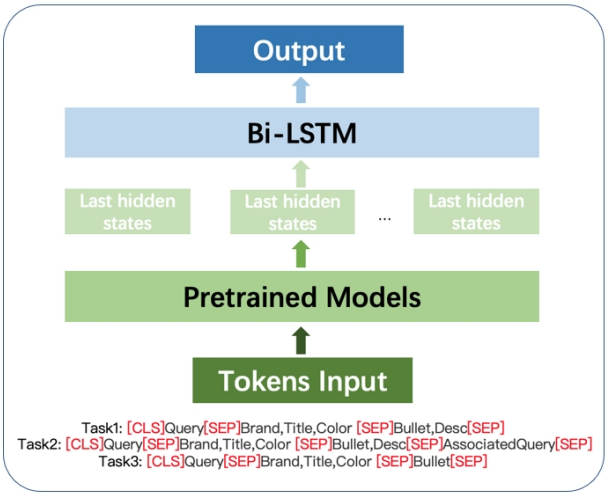

Figure 1: Model architecture.

3.3.2 Loss Functions. For task1, we use MSE loss function (the four categories are mapped as 1.0, 0.1, 0.01 and 0), we also tried pairwise loss, but the model trained by MSE converges better and faster. For task2 and task3, we use Cross Entropy loss function, which is better than Focal Loss, Weighted Cross Entropy and Cross Entropy with label smoothing.

3.3.3 Domain-adaptive pretraining. Domain-adaptive pretraining (a second phase of pretraining in domain) leads to performance gains [5]. In our methods, we continued to pretrain XLM-RoBERTa-large and RoBERTa-large using the competition dataset and achieved better result in task1. Specifically, we continued to pretrain Roberta-large using English dataset, and xlm-roberta-large using all three languages dataset combined. In this way, we can use not only the train dataset, but also the product information that do not appear in the train dataset.

3.3.4 Ensemble. For task1, we train several different versions of the models and apply weighted average to get better ensemble result, the details are shown in Table 1. Due to the limitation of time and computation resources, we only use 3 of 5 folds task2 models for final submission, and 2 of 5 folds task3 models.

3.3.5 multi-task learning. On task2 and task3, we use multi-task learning and achieve about 0.1%~0.2% improvement on local validation set. In task2, we jointly train four 2-class classifiers while training the main 4-class classifier. In task3, we jointly train the four classifier and the classifier of substitute label. The loss functions are as follows:

$$L_{task2} = 0.5 * L_{esci} + 0.125 * (L_e + L_s + L_c + L_i) \quad (1)$$

$$L_{task3} = 0.5 * L_{esci} + 0.5 * L_s \quad (2)$$

3.3.6 Hyperparameters. The hyperparameters in our method are as follows:

Table 1: Some Submission Results of Task1

US	JP	ES	JP+ES	US+JP+ES	Public	Private
deberta-v3-large * 1 of 5folds	mdeberta-v3-base * 1 of 5folds	mdeberta-v3-base * 1 of 5folds	-	-	0.8946	0.8919
deberta-v3-large * 5folds	mdeberta-v3-base * 5folds	mdeberta-v3-base * 5folds	-	-	0.9003	0.8985
deberta-v3-large * 4 of 5folds + roberta-large * 2 of 5folds	mdeberta-v3-base * 4 of 5folds + xlm-roberta-large * 2 of 5folds	mdeberta-v3-base * 4 of 5folds + xlm-roberta-large * 2 of 5folds	-	-	0.9019	0.9002
deberta-v3-large * 4 of 5folds + pretrained roberta-large * 2 of 5folds	mdeberta-v3-base * 4 of 5folds + pretrained xlm-roberta-large * 2 of 5folds	mdeberta-v3-base * 4 of 5folds +pretrained xlm-roberta-large * 2 of 5folds	-	-	0.9017	0.9026
deberta-v3-large * 4 of 5folds + pretrained roberta-large * 4 of 5folds	mdeberta-v3-base * 2 of 5folds	mdeberta-v3-base * 2 of 5folds	mdeberta-v3-base * 5folds	pretrained xlm-roberta-large * 5folds	**0.9047**	**0.9036**

- Max length of input: 400
- Learning rate for pretrained model: 7e-6 for Deberta-v3-large and 2e-5 for others
- Learning rate for BiLSTM and MLP parameters: 1e-3
- Batch size: 32
- Freeze embedding parameters
- Linear schedule with warm up: 5% warmup steps

3.4 Inference acceleration

We speed up the model inference by PyTorch amp and presort input text according to the token length.

4 DISCUSSION

In this part, we want to share some ideas that we tried but didn't use in the final submission, hoping to provide some useful insights, maybe there are better ways to use them and achieve better results.

4.1 Cross features

The dataset consists of Query-Product pairs, and there are some products that appear in different queries, which we believe should be useful.

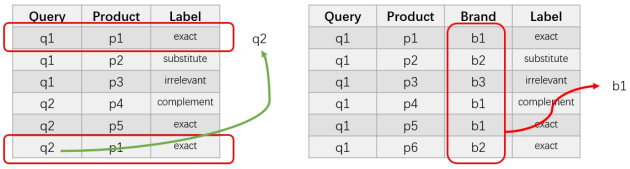

Figure 2: Cross features.

As shown on the left of Figure 2, product p1 appears in queries q1 and q2, so q2 should be useful supplementary information when predicting q1 and p1. Similarly, we can obtain associated queries for each product and it should be noted that these queries must only be obtained from the train dataset to prevent label leakage. Through this method, we achieved more than 0.1% improvement on the local validation set, but not much improvement on the public leaderboard. In the end we only used this feature in the submission of task2.

Another cross feature is shown on the right of Figure2. Through data analysis, we found that among all pairs of the same query, the brand name with the most occurrences is more likely to be Exact, such as b1 in the figure. However, we didn't have enough time to experiment this feature in our methods in the end.

4.2 Data augmentation

We experimented with translation and text generation to augment the data. Specifically, we translated the Japanese and Spanish dataset into English and also trained a BART [9] text generation model to generate query from product title and bullet point, but these methods didn't work due to the extra noise probably introduced.

4.3 Post-processing

In the challenge, we found that given a query, the more non-Exact categories in the candidate products, the lower the NDCG score is likely to be. These Query-Product pairs can be considered as hard cases. The intuition of post-processing is to pick out these hard pairs according to the score distribution predicted by stage-1 models, and then train a stage-2 model with the query and product title of multiple pairs, an example is shown in Figure 3. Ultimately, we did not achieve higher NDCG score than stage-1 model in this way, so we think maybe better modeling methods and more experiments are needed.

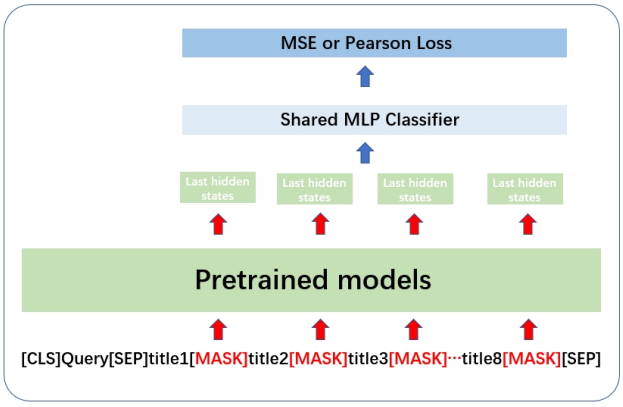

Figure 3: Post-processing Model Architecture.

5 CONCLUSION

In this paper, we introduce our final submitted solution, which won the 2nd place in the KDD Cup 2022 ESCI Challenge Task 1, and also obtained competitive results on Task2 and Task3 without much

targeted optimization (8th and 6th respectively). We present how we train multilingual models with multiple pretrained models and strategies, and ensemble different models to enhance the model. In addition, we also share the ideas that we think are worth further discussion and experimentation.

REFERENCES

[1] Zeyuan Allen-Zhu and Yuanzhi Li. 2020. Towards Understanding Ensemble, Knowledge Distillation and Self-Distillation in Deep Learning. *ArXiv* abs/2012.09816 (2020).

[2] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *ArXiv* abs/2003.10555 (2020).

[3] 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*.

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

[5] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. *ArXiv* abs/2004.10964 (2020).

[6] Lars Kai Hansen and Peter Salamon. 1990. Neural Network Ensembles. *IEEE Trans. Pattern Anal. Mach. Intell.* 12 (1990), 993–1001.

[7] Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing. *ArXiv* abs/2111.09543 (2021).

[8] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. *ArXiv* abs/2006.03654 (2021).

[9] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *ACL*.

[10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv* abs/1907.11692 (2019).

[11] C. Krishna Reddy, Lluís Màrquez i Villodre, Francisco B. Valero, Nikhil S. 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. *ArXiv* abs/2206.06588 (2022).