A Winning Solution of KDD CUP 2022 ESCI Challenge for Improving Product Search

Jinzhen Lin
jinzhen.ljz@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Lanqing Xue
lanqing.xlq@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Zhenzhe Ying
zhenzhe.yzz@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Changhua Meng
changhua.mch@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Weiqiang Wang
weiqiang.wwq@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Haotian Wang
wht209287@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

Xiaofeng Wu
congyu.wxf@antgroup.com
Ant Group
Hangzhou, Zhejiang, China

ABSTRACT

Nowadays, e-commerce has become one of the main ways for people to shop. To improve users' shopping experience and platform revenue, an accurate search ranking algorithm is needed. To improve product search, Amazon published a large-scale shopping queries dataset and hosted *KDD Cup 2022 ESCI challenge for improving product search.*

In this technical report, we present our solution to this challenge. Realizing that short queries will have a great impact on the ranking accuracy, we use a query-based TF-IDF procedure to extract keywords from product titles, product bullet points, and product descriptions. Then, we use those keywords as query features. This query augmentation technique can greatly improve the ranking and classification accuracy. We also use self-distillation, post-processing, ensemble, and some other technologies. And finally, with our solution, our team *day-day-up* won 1st place at task2 and task3, and 3rd place at task1, among 1699 participants [1].

KEYWORDS

Product Search, TF-IDF, Self Distillation, ESCI Challenge

ACM Reference Format:
Jinzhen Lin, Lanqing Xue, Zhenzhe Ying, Changhua Meng, Weiqiang Wang, Haotian Wang, and Xiaofeng Wu. 2022. A Winning Solution of 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]For more details, please see https://www.aicrowd.com/challenges/esci-challenge-for-improving-product-search/leaderboards

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

1 INTRODUCTION

1.1 Background

E-commerce has gradually replaced traditional shopping because it has higher convenience. And the e-commerce platform is trying to improve users' shopping experience. The search experience is an important part of the user experience. If the platform has a higher search ordering accuracy, users can search for the products they want faster. On the one hand, it saves users time. On the other hand, it also helps to improve the platform's revenue.

Much work has been focused on the search ranking problem in recent years. To further improve the product search ordering accuracy, Amazon published a large-scale shopping queries dataset [6] and hosted *KDD Cup 2022 ESCI challenge for improving product search.*

1.2 Dataset and Problem

Considering the limitations of binary relevance, the dataset uses four classes to represent different correlations. The four classes are Exact (E), Substitute (S), Complement (C) and Irrelevant (I). The dataset is multilingual and contains queries in English, Japanese, and Spanish.

The ESCI challenge contains three different tasks:

- Task1: **Query-Product Ranking**
  Given a query and a set of products, we should rank all of the products, products with higher relevance to the query should be ranked higher. The metric of task1 is Normalized Discounted Cumulative Gain (nDCG), and the gains of the four classes (Exact, Substitute, Complement and Irrelevant) are 1.0, 0.1, 0.01 and 0.0, respectively.
- Task2: **Multiclass Product Classification**
  In this task, we should find the relevance class (Exact, Substitute, Complement or Irrelevant) of each query and product pair. This is exactly a multi-class classification problem. And the metric of this task is Micro-F1. Note that in multi-class classification problem, the Micro-F1 is exactly the classification accuracy.

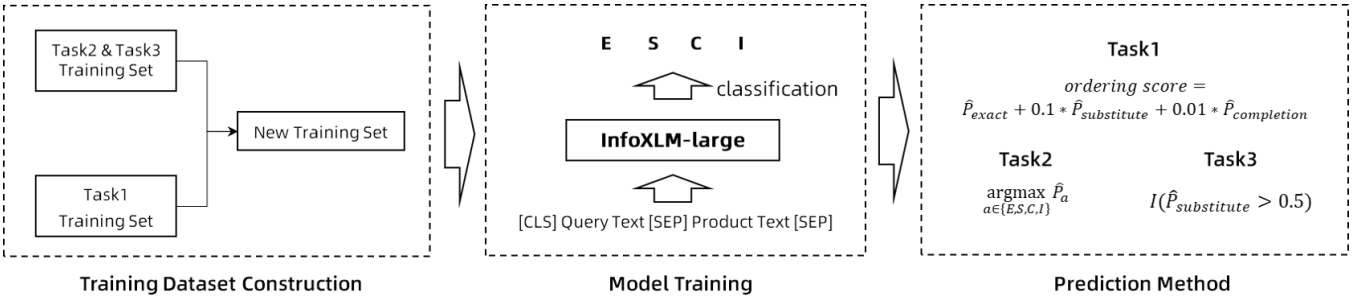

Figure 1: Overall Architecture

- Task3: **Product Substitute Identification**
  In this task, we should find whether the product is a substitute for a given query. This is also a classification problem. And the metric of this task is also Micro-F1 (which is exactly the classification accuracy).

The datasets of task2 and task3 are the same. Task1 has a different dataset with a different sample size and different label distribution.

The summaries of two datasets (with train dataset, public test dataset and private test dataset) are shown in Table 1 and 2. And the label distribution (in %) of the two datasets are shown in Table 3. More details about the dataset can be found in [1].

Table 1: Summary of task1 dataset (Small dataset)

Language	# Queries	# Judgements	Avg. Depth
English	29844	601462	20.2
Spanish	8049	218826	27.2
Japanese	10407	297882	28.6
Overall	48300	1118170	23.2

Table 2: Summary of task2 and task3 dataset (Large dataset)

Language	# Queries	# Judgements	Avg. Depth
English	97345	1819105	18.7
Spanish	15180	356578	23.5
Japanese	18127	446055	24.6
Overall	130652	2621738	20.1

Table 3: Dataset label distribution (in %)

Dataset	E	S	C	I
Task1 (Small)	43.72	34.33	5.13	16.82
Task2 and Task3 (Large)	65.20	21.91	2.89	10.00

2 METHOD

2.1 Overall Architecture

As shown in Figure 1, we choose InfoXLM large [1] as our backbone model since the datasets are multilingual, and InfoXLM-large is a pretrained model with excellent cross-language understanding performance. Since the data of all three tasks contain labels with the same definition, we concatenate the training set of all three tasks as a new training set after deduplication. Then all three tasks use the same model trained on the new training set. We implement our model with PyTorch [5] and huggingface transformers [7].

The output of the model can be submitted to different tasks after different processing:

- Task1 (Ranking): Order the product by the formula 1.

$$score = \hat{P}_{exact} + 0.1 * \hat{P}_{substitute} + 0.01 * \hat{P}_{completion} \quad (1)$$

  The class weight is exactly the gain of four labels.
- Task2 (Classification): Take the label with the highest prediction probability as the prediction result
- Task3 (Classification): Check whether $\hat{P}_{substitute}$ is greater than 0.5 (or other thresfold) and the prediction result is obtained

2.2 Query-Based TF-IDF Procedure

The queries are short text, which is very unfavorable for understanding the meaning of queries. Therefore, we use a query-based TF-IDF procedure to extract keywords from the text of products. Specifically, as shown in Figure 2, we take the titles of all products corresponding to the query as a document and then use TFIDF to extract keywords. We also get the keywords of product_bullet_point and product_description for each query. In this way, the extracted keywords can be used as the query feature. And it can be put into the input text.

In addition, we also add the brand and color names of all products with the same query to the model. In intuition, if a word in the query represents a brand, but we don't point it out, it will affect the prediction results of some goods that are not of this brand.

This query augmentation technique greatly benefits our model performance, especially for task2 and task3. We get an improvement of more than 0.01 from it (task2). This technique has less gain on task1 because the task1 focuses on ordering different products with the same query, so the features of the query are less important, and the features of products are more important.

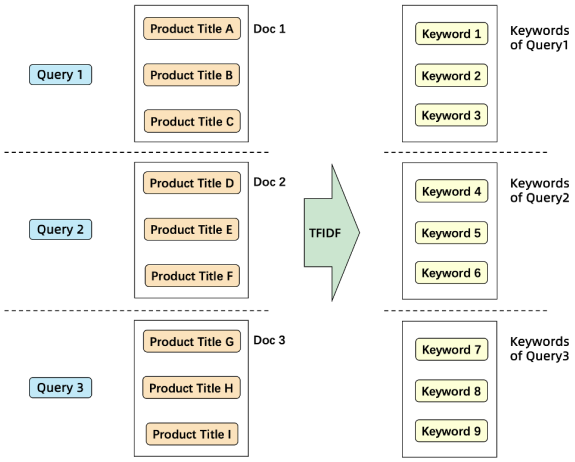

Figure 2: Query-based TF-IDF procedure

2.3 Self Distillation

We notice that there are some noise data in the dataset. To improve the model robustness and overcome the impact of noise data, we use a self-distillation technique.

We first get the prediction probability on the training set through 10-fold cross-validation on the training data and then take the mean of prediction probability and the true label probability as a soft label. For example, suppose the prediction probability of one sample is $(0.4, 0.3, 0.2, 0.1)$, and the true label is 0 (that is, the true label probability is $(1, 0, 0, 0)$), then we have a soft label $(0.7, 0.15, 0.1, 0.05)$. Then we use the soft label for model training.

Such an approach can significantly improve the model's robustness and overcome the noise data's impact. With this method, my task2 score of a single model can improve by 0.003.

2.4 Post Processing

At the beginning of the competition, some samples from the test sets of task2 and task3 are included in the training set of task1. This is serious data leakage, and after the leakage is fixed by removing some test set samples, we find that the label distribution of the test dataset has a significant change.

Based on this, we first significantly improve our task2 and task3 score by adjusting the threshold and increasing the probability of a particular label (for example, add 0.1 to the prediction probability of label 0). And after that, we future use LightGBM [3] to replace the manual design post-processing rules. The features of our LightGBM model are the prediction probabilities of all four labels, the query locale, the boolean value that identifies whether the sample is from the public test dataset of task1, etc.

2.5 Ensemble, Inference Acceleration and Other Technologies

To future improve our scores, we use ensemble technology. The ensemble method is simple probability averaging. We use 4 models in task2 and task3 and use 8 models in task1. To increase the difference between models and produce a better ensemble effect, we use the following methods:

- Add `token_type_ids` and `token_type_emebddings`. The InfoXLM model is based on RoBERTa model [4], which doesn't use `token_type_ids` and `token_type_emebddings` like BERT [2]. We initialize `token_type_emebddings` with a normal distribution with mean 0 and standard deviation 0.01. Then `token_type_ids` is set to 0 for the tokens of query features and set to 1 for the tokens of product features.
- For some models, we replace the pooler output with the first emebddings of last hidden states.
- Some models use cased text for training and the others use uncased text.

To save inference time, we use the following methods:

- We read 1024 samples from the dataloder at a time and ordered them according to the number of non-padded tokens. Then the 1024 samples were split into 16 pieces. In this way, shorter texts can have a shorter prediction time.
- For model ensemble, not all models need to make complete predictions. For example, suppose we have four models, and the mean prediction probability of the first three models is (0.7, 0.1, 0.1, 0.1). Then the fourth model does not need to predict this sample because its prediction results can not change the final prediction anyway. Even if the prediction probability of the fourth model for this sample is (0.0, 1.0, 0.0, 0.0), the mean prediction probability of the four models is still (0.525, 0.325, 0.075, 0.075), and the final prediction result is still the first label. Based on this idea, we can reduce many unnecessary predictions in the prediction of the third and fourth models.

We also crawl the titles of the products in English, Spanish, Japanese, and Chinese and added them as product features. We get a gain of 0.001 through the crawled title.

3 EXPERIMENT RESULTS

Table 4, Table 5 and Table 6 are the top10 score of all the tree task3. With our solution, our team *day-day-up* won 1st place in task2 and task3, and won 3rd place in task1 private score leaderboard.

Table 4: Top 10 score of task1. Our team "day-day-up" won 3rd in private score leaderboard and 2nd place in public score leaderboard.

Rank	Team Name	NDCG (Private)	NDCG (Public)
1	www	0.9043	0.9057
2	qinpersevere	0.9036	0.9047
3	**day-day-up**	**0.9035**	**0.9056**
4	GraphMIRAcles	0.9028	0.9036
5	ZhichunRoad	0.9025	0.9035
6	ETS-Lab	0.9014	0.9025
7	ALONG	0.9014	0.8999
8	ljr333	0.9008	0.9012
9	NeuralMind	0.9007	0.9012
10	zackchen	0.8998	0.9030

Table 5: Top 10 score of task2. Our team "day-day-up" won 1st both in private score leaderboard and public score leaderboard.

Rank	Team Name	F1 (Private)	F1 (Public)
1	**day-day-up**	**0.8326**	**0.8320**
2	ETS-Lab	0.8325	0.8303
3	Uni	0.8273	0.8281
4	cmb-ai	0.8251	0.8234
5	MetaSoul	0.8207	0.8182
6	www	0.8204	0.8209
7	ZhichunRoad	0.8194	0.8176
8	qinpersevere	0.8191	0.8181
9	zackchen	0.8189	0.8212
10	LYZD-fintech	0.8183	0.8177

Table 6: Top 10 score of task3. Our team "day-day-up" won 1st both in private score leaderboard and public score leaderboard.

Rank	Team Name	F1 (Private)	F1 (Public)
1	**day-day-up**	**0.8790**	**0.8766**
2	ETS-Lab	0.8771	0.8749
3	Uni	0.8754	0.8744
4	cmb-ai	0.8734	0.8708
5	LYZD-fintech	0.8708	0.8688
6	qinpersevere	0.8701	0.8684
7	wookiebort	0.8687	0.8673
8	ZhichunRoad	0.8686	0.8678
9	NTT-DOCOMO-LABS-GREEN	0.8677	0.8655
10	rein20	0.8668	0.8652

4 CONCLUSION AND FUTURE WORK

In this technical report, we present our solution to KDD CUP 2022 ESCI challenge for improving product search. Our solution won 1st place for task2 and task3, and 3rd place for task1.

In the future, we can try more knowledge fusion methods to understand the query better. In addition, this challenge is an NLP competition and uses text only, and we can also continue to study how to use multimodal information in product search ranking.

REFERENCES

[1] Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An Information-Theoretic Framework for Cross-Lingual Language Model Pre-Training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Online, 3576–3588. https://doi.org/10.18653/v1/2021.naacl-main.280

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[3] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017).

[4] 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 preprint arXiv:1907.11692* (2019).

[5] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. http://arxiv.org/abs/1912.01703 arXiv:1912.01703 [cs, stat].

[6] 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. (2022). arXiv:2206.06588

[7] 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*. Association for Computational Linguistics, Online, 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6