Enhancing User Behavior Alignment by Input-Level Model Cooperation and Model-Level Parameter Optimization

Yingyi Zhang[1,2,†], Zhipeng Li[1,2,†], Zhewei Zhi[1,2,†], Xianneng Li[1,2,*]
[1]School of Economics and Management, Dalian University of Technology
[2]Institute for Advanced Intelligence, Dalian University of Technology
{yingyizhang, lizhipeng, dzwer}@mail.dlut.edu.cn, xianneng@dlut.edu.cn

Abstract

In this paper, we investigate how to improve the large language model (LLM) in the user behavior alignment task, which is constrained by input confusion and process uncertainty. We propose a novel framework that employs input-level model cooperation and model-level parameter optimization. Specifically, in input-level model cooperation, we use the small language models to provide supplementary information to the LLM from both chain-of-thought and semantic similarity perspectives. In model-level parameter optimization, we first use data selection methods to train different models and then hybridize them to obtain the best one. The proposed framework was verified in the KDD Cup 2024 and achieved rank-2 performance, with code open-sourced at here[1].

CCS Concepts

• **Information systems** → **Recommender systems**; **Language models**.

Keywords

User Behavior Alignment, Large Language Model, Model Cooperation, Supervised Fine-Tuning

ACM Reference Format:

Yingyi Zhang, Zhipeng Li, Zhewei Zhi, Xianneng Li. 2024. Enhancing User Behavior Alignment by Input-Level Model Cooperation and Model-Level Parameter Optimization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'24)*. ACM, New York, NY, USA, 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

How to recommend products is a key challenge that an online e-commerce platform must face. Almost all researchers [4] indicate that the recommendation task is not trivial but a systematic problem with complex components varying from the product side to the user side. Compared to the fixed products whose features remain unchanged after being uploaded, users' preferences, intentions, and behaviors constantly evolve. Therefore, aligning the recommendation model with consumer behavior has become increasingly necessary. In practice, this alignment requires researchers to deeply understand user behavior for effective recommendations.

Traditional methods [3, 6] for understanding user behavior, which were prevalent before the advent of large language models (LLMs), involved a three-step process: *problem definition* → *mathematical formulation* → *model development*. Although these methods have yielded relatively robust performance through developing a unique model for each user behavior alignment, the advent of LLMs has fundamentally revolutionized these approaches. From the perspective of LLMs, which learn the distribution of text information in the real world, a single LLM can serve all user behavior alignment tasks in a zero-shot manner. Thus, the user behavior understanding process becomes *problem definition* → *input design* → *LLM parameter optimization*.

However, endowing a general LLM with the ability to understand users' behavior of a specific e-commerce platform remains an open challenge. Learning from a large amount of data is both an advantage of LLM and a drawback, as it makes them overly generalized. Thus, two challenges are aroused: *1) input confusion:* when the LLM encounters a task, the complexity of the learning corpus may prevent it from accurately understanding the task based on a concise description alone; *2) process uncertainty:* the LLM learns general knowledge, which can lead to uncertainty when answering specific domain questions.

Hence, in this paper, we study two research questions: *1) How can we make the LLM recognize the current task?* This involves making the domain-specific input more comprehensive so that the LLM can accurately understand the context it faces. *2) How can we enhance the LLM's ability to achieve the current task?* This involves improving the efficiency of the LLM's parameters to better answer domain-specific questions. To address the questions above, we propose a novel framework for enhancing user behavior understanding through input-level **M**odel **C**ooperation and model-level **P**arameter **O**ptimization (**MCPO**). This framework involves using a small model to supply information for the input of the large model and leveraging data selection and model hybridization techniques to optimize the parameters of the LLM.

2 Amazon KDD Cup 2024 track 3: User behavior alignment

In the user behavior alignment task of Amazon KDD Cup 2024, we are committed to using LLMs for mining the highly heterogeneous user behaviors such as browsing, purchasing, review, and query-then-clicking, to accomplish a variety of online shopping questions.

[1]https://gitlab.aicrowd.com/li_zhi_peng_/amazon-kdd-cup-2024-starter-kit/-/tree/submission-1.19_7?ref_type=tags
† These authors contributed equally to this research.
* The corresponding author.

KDD'24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

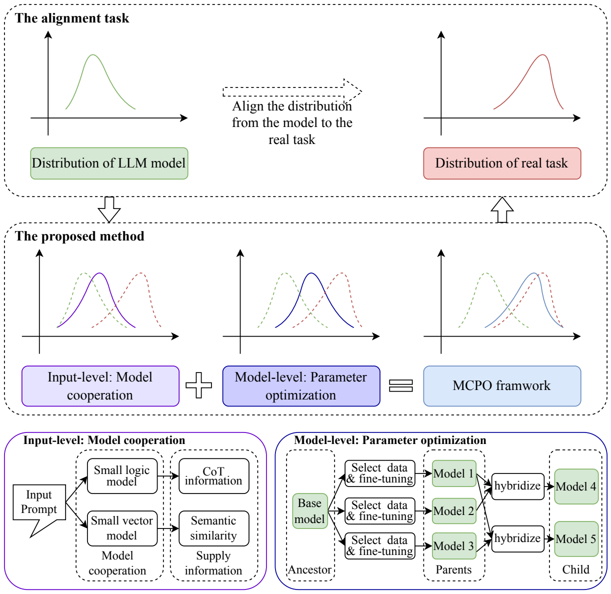

Figure 1: The architecture of our proposed framework, resolves the user behavior alignment task from 1) input-level model cooperation, and 2) model-level parameter optimization.

These questions contain multiple-choice questions (MCQ), retrieval questions (RTQ), generation questions (GENQ), and ranking questions (RKQ).

As shown in Figure 1, the overall task is to align the distribution of the LLM with the real task requirements. Given a LLM M_L and an input p_o, the output o can be defined as:

$$o = M_L(p_o). \tag{1}$$

However, the corpus textual distribution on which M_L is trained often differs from the task-specific distribution. To address this discrepancy, we develop two methods: input-level model cooperation and model-level parameter optimization. Additionally, we incorporate prompt engineering into these methods to further enhance performance.

3 Input-Level Model Cooperation

3.1 Chain-of-thought

LLMs often struggle to understand their tasks and the contextual information accurately with limited input. We add background knowledge by supplying chain-of-thought (CoT) information to address this. Specifically, we start with the original input prompt p_o and use two models, M_L (the LLM) and M_S (the small model). The goal is to generate a more informed input p_i by providing sufficient supplementary information to p_o.

Table 1: Result of supplying chain-of-thought

COT	MCQ Score (Round 1)
baseline(ecellm-M)	0.607
self-supplement	0.658
other supplement (phi-3)	0.690

Table 2: Result of applying semantic similarity

Ranking RAG	RKQ Score
baseline (llama3-70B)	0.737
+system_prompt	0.839
$k = 1$	0.890
$k = 3$	**0.901**
$k = 5$	0.888

We employ two technical methods: first, allowing the LLM to self-supplement the input information, and second, using smaller models specialized in logical reasoning to enhance the input information further. The informed input is:

$$p_i = \begin{cases} M_L(p_o|CoT) & \textit{self-supplement}; \\ M_S(p_o|CoT) & \textit{other supplement}. \end{cases} \tag{2}$$

The informed prompt p_i can then obtain CoT knowledge to be used as input to the LLM M_L. By making the input more task-specific and domain-informed, the model's output $o = M_L(p_o + p_i)$ distribution can better align with the task distribution.

3.2 Semantic similarity

For questions with options, the correct answer must exhibit semantic similarity. Since all entities are interconnected, a stronger relationship indicates more accuracy. Thus, the semantic similarity should be explicitly reflected in the input.

We apply a vector model M_V to calculate the similarity between the main body q of the question and each option $I = [i_1, \ldots, i_n]$. Firstly, we embed them into vectors:

$$e^q = M_V(q),\ e^{i_j} = M_V(i_j)\ j \in [1, 2, \ldots, n]. \tag{3}$$

Then, we calculate the similarity score between the main body and the options:

$$s^{i_j} = e^q \otimes e^{i_j}. \tag{4}$$

Finally, we rank the options by the similarity score and explicitly write the top k of them $I^r = [i^r_1, \ldots, i^r_k]$ into the prompt. The final input is $[p_o, I_r]$.

4 Model-level Parameter Optimization

4.1 Model selection

The distributions of different models vary, so we first need to select a model that better aligns with the distribution of user behavior alignment tasks. To this end, we compared several base models: the ecellm-M , ecellm-L [4], llama3-8B, llama2-13B, and llama3-70B models. However, the memory and runtime requirements of the llama3-70B far exceeded online limitations, necessitating a method to reduce memory demand and runtime. Therefore, we applied the AWQ[2] quantization as the solution.

4.2 Instruction data selection

To align the distribution of the llama3-70B model with the user behavior alignment task, we need to select appropriate data for fine-tuning. Intuitively, using comprehensive data would generally

Table 3: Results of model selection

Model	Score	Multiple Choice Score	Retrieval Score	Generation Score	Ranking Score
ecellm-L	0.5310	0.5543	0.4230	0.4953	0.7761
ecellm-M	0.5185	0.4970	0.4556	0.5568	0.7632
llama3-8b	0.4198	0.4549	0.3283	0.3269	0.6922
Wizard-Vicuna-30B-Uncensored-AWQ	0.3605	0.2984	0.3010	0.4844	0.6641
Qwen1.5-14B	0.2069	0.2003	0.2792	0.0267	0.5825
llama3-70B-AWQ	**0.6657**	**0.6406**	**0.7667**	**0.5755**	**0.8333**

Table 4: Result of data selection of fine-tuning and model hybridization

Model	Score	Multiple Choice Score	Retrieval Score	Generation Score	Ranking Score
baseline	0.665	0.640	0.766	0.575	0.833
$N = 2000$	0.693	0.687	0.789	0.543	**0.901**
$N = 4750$	0.710	0.692	**0.826**	0.584	0.883
$N = 6000$	**0.715**	0.697	0.811	**0.608**	0.888
$N = 6100$	0.699	0.680	0.795	0.586	0.897
$N = 10000$	0.696	0.676	0.804	0.586	0.863
$N = 20000$	0.695	0.700	0.768	0.551	0.873
Hybridized model (6000-20000)	0.709	**0.703**	0.796	0.584	0.868

yield better results. However, our experiments have shown that for base models with larger parameter sizes (such as 70B), smaller, refined datasets produce far superior results. Therefore, choosing the right amount of data for fine-tuning the base model is crucial. We divided the ECInstruct dataset [4] into training and validation datasets based on their labels and selected the first N data entries from both datasets for fine-tuning.

4.3 Fine-tuning method

For efficient fine-tuning in the limited shareware, we opt for the Quantized Low-Rank Approximation (QLoRA)[1] with Fully Sharded Data Parallelism (FSDP) approach[7], a fine-tuning method that ensures the effectiveness of fine-tuning while minimizing the need for VRAM.

FSDP is a parallelization technique in deep learning training that achieves efficient parallel training by distributing different parts of the model to different devices. QLoRA is a model compression technique that reduces the size and computational requirements of the model through quantization and low-rank approximation.

4.4 Model hybridize

Compared to fine-tuning, another approach for model parameter optimization is model hybridization. We primarily used the Spherical Linear Interpolation (SLERP) method for this purpose. The SLERP technique[5] is used to merge model checkpoints and represents an extension of simple weight averaging. Its success demonstrates that there is often a spherical path with a lower loss barrier compared to direct linear interpolation.

5 Prompt engineering

5.1 Multiple choice task

We performed the following steps. 1) Prompt1: We add the sentence "Please think cautiously. The answers or responses are very important. You should only answer a number of options. Do not say other words or explanation." 2) Prompt2: we modified the option descriptions in the input based on the fine-tuning dataset ECInstruct, such as "substitute" to "product1 and product2 are similar."

5.2 Retrieval task

We performed the following steps. 1) Prompt1: We add the sentence "This is a retrieval question. You are a highly skilled online shopping assistant and a professional product retrieval expert. Your goal is to help consumers quickly and accurately identify products that meet their specific needs. You provide a clear and concise list of retrieval results, including the product name, key attributes, and how they meet the requirements. Please analyze the following request and deliver accurate retrieval results." 2) Prompt2: We add the sentence "Based on your previous knowledge then generate the answer." 3) Prompt3: We add the sentence "generate the answer as clearly as possible". Unfortunately, we did not merge this prompt in the final submission.

5.3 Generation task

We performed the following steps. 1) Prompt1: We speculated on several scenarios and designed different roles and prompts for the LLM based on each context. such as "You are a marketing specialist. Write a detailed and persuasive product description based on the following instructions." The other context prompt details can be found in the code. 2) Prompt2: We add the sentence "Based on

Table 5: Results of prompt engineering

Prompt Type	Multiple Choice Task	Retrieval Task	Generation Task	Ranking Task
Baseline	0.697	0.811	0.496	0.888
Prompt1	0.704	0.834	0.608	0.901
Prompt1+Prompt2	**0.710**	0.838	0.622	0.905
Total Score	**0.710**	0.838	**0.630**	0.914
Prompt1+Prompt2+Prompt3	-	**0.840**	-	**0.918**

your previous knowledge then generate the answer." and change "Answer" to "Response (limit to 30 words)".

5.4 Ranking task

We performed the following steps. 1) Prompt1: "Based on your previous knowledge then generate the answer. You are a helpful online shopping assistant. Please answer the following question about online shopping and follow the given instructions." 2) Prompt2: "In this task, each question is associated with a requirement and a list of candidate items, and the model is required to re-rank all items according to how each item satisfies the requirement." 3) Prompt3: We add the sentence "Only respond with the ranking results. Do not say any word or explanations". Unfortunately, we did not merge this prompt in the final submission.

6 Experiment

6.1 Input-Level Model Cooperation

As shown in Table 1, the CoT supplement provides significant improvement in both self-supplement and other supplement (Phi-3) for MCQ. The results of using semantic similarity, as presented in Table 2, indicate that adding the top three semantically similar options can significantly enhance the RKQ ability of the LLM.

6.2 Model-level Parameter Optimization

The results of model selection are shown in Table 3. We found that models with a parameter size smaller than 20B could not achieve the desired effect. The initial llama3-70B model already demonstrated superior offline and online performance compared to the other models.

The results of data selection and fine-tuning are shown in Table 4. As N increases from small to large, the online performance of the fine-tuned model generally shows an initial increase followed by a decrease. Additionally, We found that as the amount of fine-tuning data increases, the model's performance on multiple-choice questions may improve, but its performance on generation questions may worsen. The peak overall performance is achieved with 6000 data entries (3000 from the training dataset and 3000 from the validation dataset). Interestingly, even a slight increase or decrease of 100 data entries can significantly reduce the model's performance.

The results of model hybridization are shown in Table 4. Aiming to improve scores for multiple-choice questions, we selected the best overall performance model (N = 6000) and the best multiple-choice performance model (N = 20000) for hybridization. The resulting 6000-20000-merge model did not perform as expected in offline testing, but its online performance was impressive. The online multiple-choice score reached 0.703, a result we had never achieved before, despite a decrease in the generation score.

6.3 Prompt engineering

The results of prompt engineering are shown in Table **??**, demonstrating that the purpose of prompt engineering is to simulate an intelligent and cautious online shopping assistant. This approach significantly enhances the alignment between the distribution of the LLM and the user behavior alignment tasks.

7 Conclusion

In this paper, we aim to solve the input confusion and process uncertainty of LLM in the user behavior alignment task. We propose a novel **MCPO** framework that employs input-level model cooperation and model-level parameter optimization. The experiments demonstrate that our framework significantly enhances the alignment capability of the LLM.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (NSFC) under Grant 72071029 and 72231010.

References

[1] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. arXiv:2305.14314 [cs.LG] https://arxiv.org/abs/2305.14314

[2] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration. arXiv:2306.00978 [cs.CL] https://arxiv.org/abs/2306.00978

[3] Cataldo Musto, Marco de Gemmis, Giovanni Semeraro, and Pasquale Lops. 2017. A Multi-criteria Recommender System Exploiting Aspect-based Sentiment Analysis of Users' Reviews. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (Como, Italy) *(RecSys '17)*. Association for Computing Machinery, New York, NY, USA, 321–325. https://doi.org/10.1145/3109859.3109905

[4] Bo Peng, Xinyi Ling, Ziru Chen, Huan Sun, and Xia Ning. 2024. eCeLLM: Generalizing Large Language Models for E-commerce from Large-scale, High-quality Instruction Data. In *Forty-first International Conference on Machine Learning*. https://openreview.net/forum?id=LWRI4uPG2X

[5] Ken Shoemake. 1985. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*. 245–254.

[6] Mingyue Zhang, Xuan Wei, Xunhua Guo, Guoqing Chen, and Qiang Wei. 2019. Identifying Complements and Substitutes of Products: A Neural Network Framework Based on Product Embedding. *ACM Trans. Knowl. Discov. Data* 13, 3, Article 34 (jun 2019), 29 pages. https://doi.org/10.1145/3320277

[7] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. 2023. PyTorch FSDP: Experiences on Scaling Fully Sharded Data Parallel. arXiv:2304.11277 [cs.DC] https://arxiv.org/abs/2304.11277