

DMoERM: Recipes of Mixture-of-Experts for Effective Reward Modeling

Shanghaoran Quan
Beihang University
shrquan@buaa.edu.cn

Abstract

The performance of the reward model (RM) is a critical factor in improving the effectiveness of the large language model (LLM) during alignment fine-tuning. There remain two challenges in RM training: 1) training the same RM using various categories of data may cause its generalization performance to suffer from multi-task disturbance, and 2) the human annotation consistency rate is generally only 60% to 75%, causing training data to contain a lot of noise. To tackle these two challenges, we introduced the idea of Mixture-of-Experts (MoE) into the field of RM for the first time. We propose the Double-Layer MoE RM (DMoERM). The outer layer MoE is a sparse model. After classifying an input into task categories, we route it to the corresponding inner layer task-specific model. The inner layer MoE is a dense model. We decompose the specific task into multiple capability dimensions and individually fine-tune a LoRA expert on each one. Their outputs are then synthesized by an MLP to compute the final rewards. To minimize costs, we call a public LLM API to obtain the capability preference labels. The validation on manually labeled datasets confirms that our model attains superior consistency with human preference and outstrips advanced generative approaches. Meanwhile, through BoN sampling and RL experiments, we demonstrate that our model outperforms state-of-the-art ensemble methods of RM and mitigates the *overoptimization* problem. Our code is available at: <https://github.com/quanshr/DMoERM>.

1 Introduction

After an initial stage of pre-training and subsequent instruction fine-tuning (Quan, 2024), large language models (LLMs) undergo a crucial stage of high-quality alignment fine-tuning based on Reinforcement Learning with Human Feedback (RLHF) to improve their abilities (Ouyang et al., 2022; Stiennon et al., 2020). During the RLHF process,

a reward model (RM) often needs to be trained, which acts as a proxy of human preferences and assigns scores to the outputs of the LLM. The scores are then used as reward signals to optimize the LLM through reinforcement learning (RL). In this process, the LLM and RM are interdependent and iteratively optimized, and the RM is expected to be highly consistent with human preferences. In addition, during the inference stage, the RM can also be augmented with Best-of- n (BoN) sampling strategies to further enhance the quality of the outputs of the LLM (Ouyang et al., 2022; Nakano et al., 2021).

Training of reward models relies on the data derived from human annotators who manually rank the varying outputs under a single input by their preferences. However, many studies have found that agreement rates among human annotators typically only range between 60-75% (Ziegler et al., 2019; Stiennon et al., 2020; Dubois et al., 2023), thereby introducing a minimum of 25% noise within the labeled dataset. One important reason for this phenomenon is the multifaceted nature of evaluation—it is often observed that one response may excel in one aspect while simultaneously falling short in another. This multifaceted evaluation conundrum has been exemplified in previous studies (Dai et al., 2023; Ganguli et al., 2022; Bai et al., 2022), which illustrate the inherent tensions between enhancing helpfulness and harmlessness. As these attributes can at times be inversely related, adjudicating between a response that is more helpful yet potentially less harmless poses a significant challenge for comparative assessment. We further validate this perspective through empirical studies.

In this study, we pioneer the integration of the Mixture-of-Experts (MoE) framework (Jacobs et al., 1991; Lepikhin et al., 2021; Du et al., 2022) into Reward Modeling. Our approach employs a double-layer MoE architecture. The outer layer

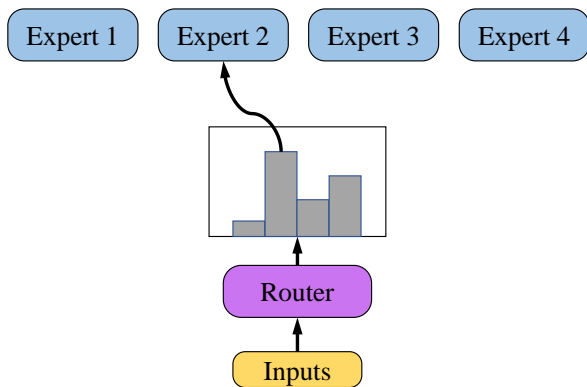


Figure 1: The outer MoE routes inputs to corresponding task-specific inner MoE.

comprises a sparse MoE model specifically designed to avoid multi-task disturbance (Standley et al., 2020). As shown in Figure 1, We categorize inputs into several distinct tasks and use a pre-trained router to route the inputs to their corresponding task-specific expert. This strategy can also facilitate distributed deployment and enhance the model’s capacity and capabilities without a commensurate increase in computational demands (Rajbhandari et al., 2022; Xue et al., 2022).

Subsequently, within each inner layer lies a dense MoE model, which is tailored to the specific capabilities set required for its category. For instance, in roleplay scenarios, we divide it into six core capabilities, including personality and emotional investment, conversational sense, empathy ability and so on (details are in Appendix Table 12). We obtain preference labels on these single capability points by calling a public API, which greatly reduces annotation costs and is sufficient to achieve satisfactory results in our experiments. Considering that capability points are equivalent to a decomposition of tasks in a low dimensional space, using low-rank adaptation (LoRA) (Hu et al., 2022) fine-tuning will be very suitable. Each LoRA fine-tuned model effectively becomes an *expert* in scoring on a singular capability point. Lastly, we aggregate the outputs from these expert models into a unified one-dimensional score with an MLP to determine the final reward value. We believe this methodology can improve the interpretability and performance of RMs since it is just like the *Chain of Thought (CoT)* for RMs. Both preference consistency and optimization evaluations indicate our model is more effective in optimizing LLMs and can mitigate the *overoptimization* problem against other state-of-the-art RM ensemble methods.

2 Related Work

2.1 Mixed-of-Experts

Mixture-of-Experts or MoEs was introduced early for machine learning applications (Jacobs et al., 1991; Jordan and Jacobs, 1994), where researchers control the allocation of different weights to different models through gate networks to mitigate interference between different types of samples. MoE can enhance the model’s generalization capability by decomposing complex tasks into several subtasks, which will help avoid multi-task disturbance (Standley et al., 2020) and meanwhile confer greater flexibility in development (Ma et al., 2018). Recently, much ongoing research has focused on the top- k (e.g., top-1 or top-2 in many works) activation MoE model (Ramachandran and Le, 2019; Clark et al., 2022; Dai et al., 2022), since it can be leveraged to enlarge parameter count and enhance model capability while keeping computational complexity nearly unchanged for both training and inference due to its sparse activating nature (Shazeer et al., 2017; Fedus et al., 2022). While MoE has achieved great success in the field of large generative language models (Shen et al., 2023; OpenAI, 2023), how to efficiently train more effective RM with MoE architecture remains largely unexplored.

2.2 Reward Model Ensembling

Reward Model Ensembling has been tried in the field of safe RLHF (Dai et al., 2023). The research is based on a widely observed imagination: the pursuit of greater helpfulness and harmlessness may often conflict in practice (Ganguli et al., 2022; Bai et al., 2022). Another weave of research ensemble RMs through multi-objective reward modeling (Ramé et al., 2023b,a) or weight-averaged reward modeling (Ramé et al., 2024), but they are struggling to formulate non-linear relationships. In addition, some research works (Coste et al., 2024; Eisenstein et al., 2023; Zhai et al., 2024) have found that training multiple reward models and aggregating them by changing the data training order, batch size, and learning rate can alleviate the overoptimization problem (Gao et al., 2023) of RM and increase its performance. However, the aggregation methods they chose were only 1) mean, 2) min, and 3) mean minus std, and the performance of aggregation was very dependent on the diversity of several models (Zhai et al., 2024), which required a lot of attempts in experiments.

3 Empirical Study

3.1 Multi-Task Training

It has been frequently observed that using irrelevant training data to train LLMs will cause their generalization performance to decrease in other tasks (Dong et al., 2023; Wen et al., 2023). This also applies to RMs. Dai et al. (2023) found that training RMs on harmless and helpfulness simultaneously will lead to the model getting sub-optimal results on both types of data. In order to further explore whether data of different categories interfere with each other, we selected the preference data for three tasks: roleplay, chitchat, and text creation. We train on different combinations of training sets and testing on all test sets. The results are shown in Table 1.

Dataset	#A	#B	#C
#A	56.7%	51.1%	49.8%
#B	52.4%	54.3%	50.3%
#C	50.8%	51.0%	58.1%
#A ∪ #B	54.2%	52.6%	50.2%
#A ∪ #C	53.3%	50.9%	55.6%
#B ∪ #C	51.7%	52.5%	56.4%
#A ∪ #B ∪ #C	52.0%	51.5%	54.8%

Table 1: The results of training on different combinations of training sets and testing on all test sets. #A, #B, and #C represent roleplay, chitchat and text creation respectively. The best values are written in bold.

We find that using separate category data give the best results in this category, and using data from other categories may affect the generalization ability under original task.

3.2 Annotation Consistency

The consistency rate of manually labeled preference data is generally only 60-75% (Ziegler et al., 2019; Stiennon et al., 2020; Dubois et al., 2023), which brings a lot of noise to the training data. Inspired by the fact that Chain-of-Thought (CoT) can improve the accuracy of reasoning (Wei et al., 2022; Zhou et al., 2023), we try to improve the consistency rate through CoT. We first conduct experiments on humans. We select the text creation subtask and randomly divide 200 pairs of responses into two groups. For the first group, we let three annotators directly rank the preference and record the average agreement rate between any two anno-

tators, which is represented as A. For the second group, we divide the text creation into five capability points: 1) intent conformity, 2) expressiveness, 3) readability, 4) content richness, 5) logic, and ask the annotators to first score the content in each single capability point, and then evaluate the overall preference. The average agreement rates on five capability points are represented as B-1 to B-5, respectively, and the final overall agreement rate is B-f. We record these results in Figure 2, and a screenshot of the annotation interface is shown in Figure 8 in the Appendix.

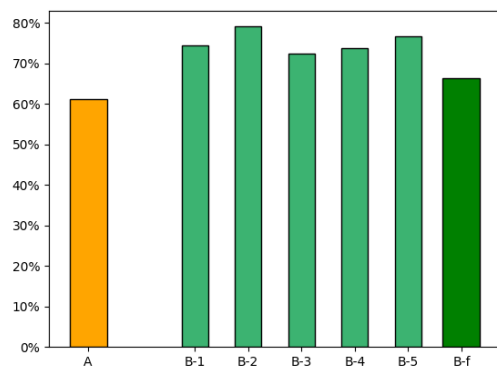


Figure 2: The results of consistency study.

We find that the consistency on capability points is significantly higher than the consistency of directly evaluating the overall preference, and the method of evaluating capability points first can increase the final overall consistency rate.

4 Methodology

4.1 Outer Layer MoE

Our first layer is a sparse MoE structure, and only the top-1 expert is activated each time. We divide the input into five categories according to tasks: text creation, roleplay, objective knowledge QA, subjective knowledge QA, and chitchat, and train an MoE for each category. When new input comes, we use a small frozen top-1 gating network pre-trained on category labels to act as the router. Formally, we have

$$chosen = \arg \max_{e=0}^4 E(t_e|x) \quad (1)$$

$$y = \text{RM}_{chosen}(x) \quad (2)$$

Here t_0, \dots, t_4 represent five tasks respectively, x represents input, $\text{RM}_0, \dots, \text{RM}_4$ represent the

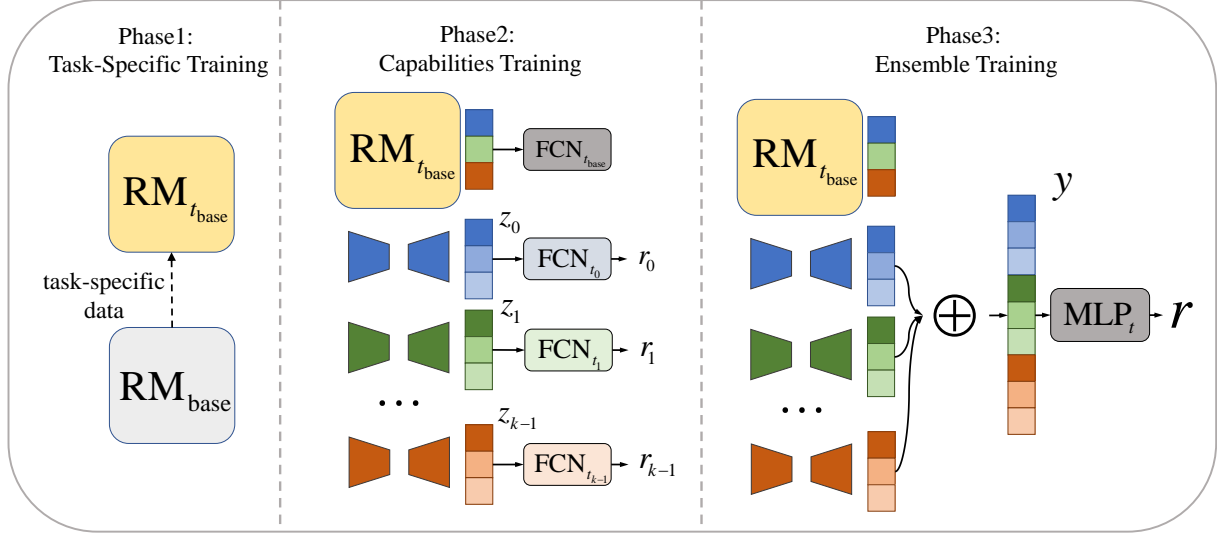


Figure 3: The training framework of each inner layer MoE. The LoRA components in the figure is only for illustration, as in actual experiments we will inject the LoRA layers into each layer of the transformers. Training details are in Section 4.2.2.

expert RMs corresponding to each task, and y represents the RM’s output.

4.2 Inner Layer MoE

4.2.1 Modeling

For each task, we first obtain a base task-specific model $RM_{t_{base}}$ from a general RM through training on task-specific preference data. Then, we divide the tasks into distinct capability points. Capability points are equivalent to a decomposition of tasks in a low-dimensional space. Define the input space as X , we need to learn an expert $RM_{t_i} : X \rightarrow Z_i$. In this work, we obtain RM_{t_i} by performing LoRA fine-tuning on $RM_{t_{base}}$.

Let $Z = Z_0, \dots, Z_{k-1}$. After we have learned k experts, we use an aggregation network to aggregate the output results of k experts to produce the final output; that is, we learn $RM_t : Z \rightarrow R$. This is a Markov process and $X \rightarrow Z \rightarrow R$ construct a homogeneous Markov chain.

We employ a FCN following each expert RM_{t_i} to serve as the value head to generate one-dimensional scores, which further maps to the range of $[0, 1]$ using the sigmoid activation function. Let $W^{(base)}$ represent the initial base model $RM_{t_{base}}$, $\Delta W^{(i)}$ represent the fine-tuned LoRA network learned for capability point i , and w_i and b_i represent the FCN network associated with capability point i . The score r_i for capability point i is then expressed as follows:

$$z_i = (W^{(base)} + \Delta W^{(i)})x \quad (3)$$

$$r_i = \sigma(w_i z_i + b_i) \quad (4)$$

To obtain a single score as the final reward from multiple experts, we concatenate all low-dimensional representations of experts and use a two-layer MLP to aggregate them. Then the final reward score r is:

$$z = \bigoplus_{i=0}^{k-1} z_i \quad (5)$$

$$r = \sigma(W_1 \text{PreLU}(W_0 z + B_0) + B_1) \quad (6)$$

Note that our MLP does not act on the final scalar outputs, but rather on the multiple low-dimensional decomposition without being fed into FCNs, as we believe there may be underlying correlations between different capability points, which can be learned using MLP in their low-dimensional embedding space.

We use logsigmoid as the loss function, which is also the most commonly used loss function for training RMs, where k represents the number of responses in a piece of data:

$$\mathcal{L} = -\frac{1}{\binom{k}{2}} E_{(x, y_w, y_l) \sim D} [\log(\sigma(r(y_w) - r(y_l)))] \quad (7)$$

4.2.2 Training

Figure 3 illustrates the training framework of each inner layer MoE. We use a pre-trained $RM_{t_{base}}$ as the base general RM, and then perform the following three phases of training in sequence:

- **Phase 1: Task-Specific Training.** Use 60% of the task-specific preference data to full-parameter fine-tune on $RM_{t_{base}}$ to get base task-specific model $RM_{t_{base}}$.
- **Phase 2: Capabilities Training.** Use the data with capability point labels (the method of obtaining these labels is introduced in Section 4.3) to train RM_t using LoRA fine-tuning to obtain $RM_{t_0}, \dots, RM_{t_{k-1}}$. Each time, a new linear head is learned from the original linear head of $RM_{t_{base}}$.
- **Phase 3: Ensemble Training.** Remove the original FCN on each expert RM and use an MLP to aggregate $RM_{t_0}, \dots, RM_{t_k}$ to obtain the final model RM_t , and use the remaining 40% task-specific preference data to train it. During this phase, $RM_{t_{base}}$ and the LoRA layers are frozen, and only the newly added MLP layer is trained.

4.3 Capability Point Labels Obtaining

Since it is costly to obtain all preference labels on each capability point, instead of manually sorting or scoring, we use the method of calling the public Ernie Bot API¹ to obtain the comparative preference of the single capability point in each response pair, which significantly reduces the labeling cost.

To avoid the positional bias inherent in LLMs, we employ a strategy that involves swapping positions and requiring each pair to be processed twice. We then selectively retain only those data pairs that exhibit consistency in twice-calling. This method also doubles as a data cleansing technique, as it effectively filters out pairs with minimal discrepancies, which may introduce potential noise into training data. The prompt template is shown in both Chinese (Table 13) and English (Table 14) in the Appendix.

Our approach does not require additional data since the task-specific data from training Phases 1 and 3 can be directly utilized as raw response

¹The ERNIE Bot API has similar functions to ChatGPT, but it is much cheaper and can achieve the same level of proficiency in Chinese: <https://cloud.baidu.com/doc/WENXINWORKSHOP/s/flfmc9do2>

pairs, which enables us to acquire capability point preferences based on them. In our experiments, we reuse all the task-specific data as raw response pairs during Phase 2.

5 Experiment Setup

5.1 Model

We use Qwen-1.8B-Chat² as both the policy model and the base reward model, which is an open source Chinese-English bilingual Transformer-based large language model proposed by Alibaba Cloud. To make a fair comparison, we use the same model as the base model of our baseline ensemble methods.

5.2 Baseline

We use a single RM baseline and collect three state-of-the-art ensembling methods for reward models from a range of papers (Coste et al., 2024; Eisenstein et al., 2023; Zhai et al., 2024). All baseline methods and our model are trained and evaluated with the same dataset.

- **Single RM**

We use the training of a single reward model as the most basic benchmark.

- **Mean Optimization**

Mean optimization simply takes the mean of the outputs of the different ensemble members:

$$R_{\mu}(x) = \frac{1}{k} \sum_{i=0}^{k-1} r_i(x) \quad (8)$$

- **Worst-Case Optimization**

Worst-case optimization (WCO) creates a conservative estimate by choosing the lowest reward from the ensemble at every step:

$$R_{WCO}(x) = \min_{i=0}^{k-1} r_i(x) \quad (9)$$

- **Uncertainty-Weighted Optimization**

Uncertainty-weighted optimization (UWO) calculates reward by combining the average reward across all models in an ensemble with the intra-ensemble variance, weighted by a

²<https://huggingface.co/Qwen/Qwen-1.8B-Chat>

coefficient λ . Mathematically, this objective is given by:

$$R_{\text{UWO}}(x) = \underbrace{R_{\mu}(x)}_{\text{mean}} - \lambda \underbrace{\frac{1}{k} \sum_{i=0}^{k-1} (r_i(x) - R_{\mu}(x))^2}_{\text{variance}} \quad (10)$$

5.3 Dataset

Our prompt dataset are very diverse and can be mainly classified into five categories: roleplay, chitchat, subjective knowledge QA, objective knowledge QA, and text creation, with some others (including logical reasoning, mathematical calculations, code understanding and generation, translation, etc). Our dataset is over 98% Chinese. We ensure that the training and test sets contain no intersection and heuristically eliminate duplicate prompts. We also filter prompts containing personally identifiable information (PII).

Our data has a turn range of 1 to 27, with an average of 3.72 (a turn contains a user query and an LLM response). Each sample’s final query has multiple responses generated through either automated or manual processes. These responses are then assigned preference rankings through manual labeling. A detailed statistic of our data is shown in Table 2.

Table 2: The statistics of our dataset.

Categories	Count	Avg. Turn
text creation	6,498	2.96
roleplay	4,150	5.32
objective knowledge QA	3,595	3.77
subjective knowledge QA	3,413	3.50
chitchat	3,315	3.45
others	709	3.40
all	21,680	3.72

We employ a rigorous annotation process with clear documentation guides to direct our annotators (we have shown an example in our GitHub repository). Each piece of data is evaluated by two annotators to ensure quality, and the final ranking is established through discussion to reach a consensus. We have observed that the consistency rate between each pair of annotators reached 74% on average.

6 Result

6.1 Training Phases

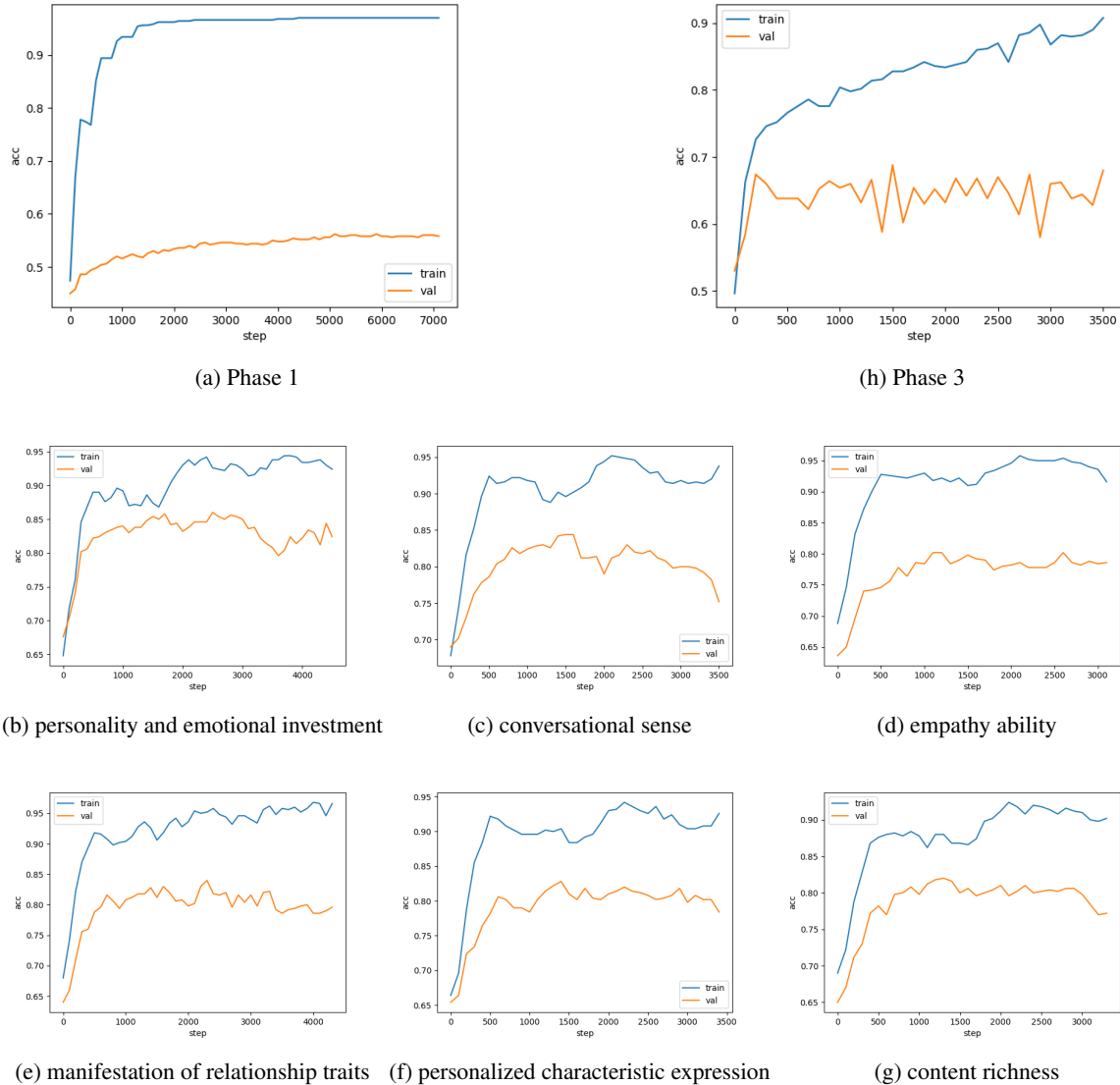
In this section, we use roleplay as an example to demonstrate the progress of the model at different training phases. By recording the accuracy of the reward model’s ranking on the pairs of responses in training and testing sets, we can intuitively display the training results. The results are shown in Figure 4. We find:

- **In Phase 1** (Figure 4a): the accuracy of the training set improves rapidly, while the improvement of the test set dataset is slow, eventually stabilizing at 56%.
- **In Phase 2** (Figure 4b to 4g): we divide the roleplay task into six capability points, namely 1) personality and emotional investment, 2) conversational sense, 3) empathy ability, 4) manifestation of relationship traits, 5) personalized characteristic expression, and 6) content richness. We find that depending on different single capability points, an accuracy of 80-86% can be achieved on the test set. Note that the training and testing set labels here are only one capability point and not an overall preference.
- **In Phase 3** (Figure 4h): the improvement in the test set was rapid, reaching a peak of 68%. Compared with the 56% accuracy in Phase 1, it can be proven that our method of training multiple experts based on different capability points and aggregating those experts can significantly improve the model’s performance.

6.2 Preference Consistency Evaluation

Since RM is essentially an imperfect proxy for human preferences, testing the consistency rate of trained RM using human-labeled preference data is a direct and effective evaluation method. Given a pair of preference data, we use the trained RMs to assign scores to each response and record the consistency between the sorting of scores and the sorting of manual labels. Higher consistency rates (or accuracy) mean better performance of RMs as the proxies for human preferences. Compared with the methods introduced in Section 5.2, we add the GPT-4 generative evaluation benchmarks for a more comprehensive evaluation. Their implementation details are introduced in Appendix B.3.3.

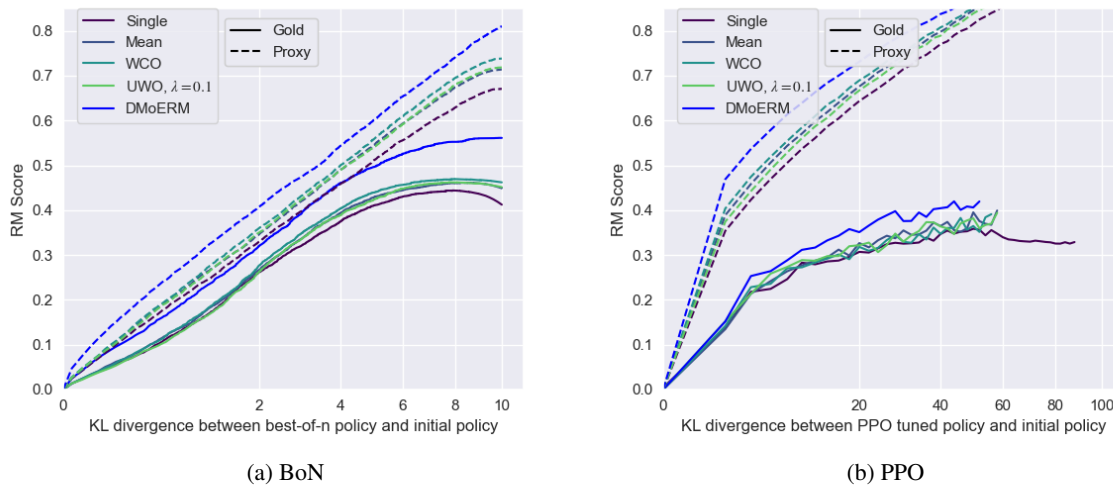
Figure 4: The progress of the model at different training stages. The horizontal axis of each image represents the number of training steps, and the vertical axis represents the accuracy of sorting pairs of responses on the training and testing set. Figure 4a shows the results of the training Phase 1. Figures from 4b to 4g show the results of the training Phase 2. Figure 4h (top-right) shows the results of the training Phase 3.



Categories	text creation	roleplay	okqa	skqa	chitchat	others	overall
Zero-shot GPT-4	58.9	56.1	64.7	63.5	54.8	60.1	59.5
One-shot GPT-4	61.4	59.9	66.5	66.2	58.4	<u>61.7</u>	62.3
Single	58.1	57.6	60.4	62.0	54.3	54.4	58.2
Mean	63.3	62.0	65.2	66.3	58.5	56.6	62.4
WCO(min)	63.6	62.3	65.5	66.1	59.1	56.7	62.6
UWO(mean minus std)	63.4	61.8	65.4	66.5	59.0	56.6	62.6
DMoERM-w/o-Outer(ours)	<u>68.3</u>	<u>66.5</u>	<u>70.1</u>	<u>71.3</u>	<u>63.2</u>	61.3	<u>67.0</u>
DMoERM(ours)	71.9	68.2	73.8	74.2	66.4	63.0	70.7

Table 3: The consistency with human preferences. Note that for the methods except GPT-4, the overall results is not simply adding each up but train with all data and test. The best performance is in bold and the second best is underlined.

Figure 5: The optimization results for BoN and PPO for the roleplay task. The x-axes have a square-root scale, and the KL divergence scale differs between BoN and PPO due to differences in the algorithm and the KL calculation. All RMs will be normalized to have a zero mean after training.



We present the results in Table 3. There are two noteworthy findings: firstly, our DMoERM achieves the best results in all categories and overall experiments, with a 6 to 8 percentage point improvement compared to other methods. This indicates that our training method can better learn human preferences without increasing the amount of training data or model parameters. Secondly, the DMoERM-w/o-Outer model removes the outer layer and instead uses the same capability point partition to train on various categories. It achieves the second effect and still has a significant improvement compared to other methods, making it a viable alternative when memory or task-specific data is limited.

6.3 Optimization Evaluation

In the optimization evaluation, we use BoN and PPO as optimization strategies to optimize the same policy model. Since we have verified in section 3.1 that the outer MoE will improve performance by dividing different categories to train different models, for the sake of fair comparison, we restrict the task within roleplay to comparing the effects of the corresponding inner MoE with other aggregation methods. Under the same degree of optimization measured by KL divergence, we use another pre-trained reward model³ fine-tuned on our task-specific data with a larger number of parameters as

³<https://huggingface.co/IDEA-CCNL/Ziya-LLaMA-7B-Reward>

the referee to judge the outputs of results, which we called the *gold* reward model. Model strategies that score higher under the gold reward model are considered better. We evaluate BoN for a maximum of $n_{max} = 60,000$ samples, which roughly equals 10 nats of KL. The reported results represent the mean outcomes derived from a set of 500 distinct prompts. For PPO, we train for 3,000 PPO steps and compute the average results derived from three distinct random seeds. We give further details on implementation and other hyperparameters in Appendix B.3.

We showcase the results in Figure 5. Both BoN and PPO optimization results demonstrate that our model consistently outperforms alternative integration approaches when policy models are optimized to the same degree. Moreover, after $n = 8,000$ (KL ≈ 8 nats) in the BoN optimization experiment, our model maintains stability without signs of overoptimization, unlike other ensemble methods that exhibit varying degrees of overoptimization. These findings suggest that our model outperforms baselines in optimizing LLMs and is capable of addressing the issue of overoptimization.

7 Conclusion

In this work, we propose DMoERM to enhance RMs' performance. The outer layer MoE divides inputs into different tasks to avoid multi-task disturbance, while the inner layer MoE reduces the impact of data noise by learning LoRA experts on

different capability points. Preference consistency experiments demonstrate our model is more representative of human preferences. Optimization evaluations indicate our model is more effective in optimizing LLMs and can mitigate the overoptimization problem.

Limitations

Although we attempt to reduce annotation costs by calling public LLM API instead of manual labeling, it is still costly as using the ERNIE Bot API will cost approximately \$3,000 in total, and using the ChatGPT API will cost about ten times more. However, if the annotation standards for human are predetermined in advance, it will not significantly bring more annotation costs. In our annotation process, we found that due to the annotators spending most of their time understanding the queries and various responses, the proposed annotation method (first annotates the preferences of pre-determined capability points and then annotates the overall preferences) only reduces the annotator's speed by about 10%, but can increase the annotation consistency by 5 percentage points.

Another problem is the training time, with about 80 NVIDIA A100 GPU hours to train one inner MoE, which is about eight times longer than training a traditional single RM with the same amount of parameters. While many works focus on exploring efficient training methods, we reserve this problem as a future work.

References

- Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. [A general theoretical paradigm to understand learning from human preferences](#). *CoRR*, abs/2310.12036.
- Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, Benjamin Mann, and Jared Kaplan. 2022. [Training a helpful and harmless assistant with reinforcement learning from human feedback](#). *CoRR*, abs/2204.05862.
- Ahmad Beirami, Alekh Agarwal, Jonathan Berant, Alexander D'Amour, Jacob Eisenstein, Chirag Nagpal, and Ananda Theertha Suresh. 2024. [Theoretical guarantees on the best-of-n alignment policy](#). *CoRR*, abs/2401.01879.
- Aidan Clark, Diego de Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake A. Hechtman, Trevor Cai, Sebastian Borgeaud, George van den Driessche, Eliza Rutherford, Tom Hennigan, Matthew J. Johnson, Albin Cassirer, Chris Jones, Elena Buchatskaya, David Budden, Laurent Sifre, Simon Osindero, Oriol Vinyals, Marc Aurelio Ranzato, Jack W. Rae, Erich Elsen, Koray Kavukcuoglu, and Karen Simonyan. 2022. [Unified scaling laws for routed language models](#). In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 4057–4086. PMLR.
- Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. 2024. [Reward model ensembles help mitigate overoptimization](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. [Stable-moe: Stable routing strategy for mixture of experts](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 7085–7095. Association for Computational Linguistics.
- Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. 2023. [Safe RLHF: safe reinforcement learning from human feedback](#). *CoRR*, abs/2310.12773.
- Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. [How abilities in large language models are affected by supervised fine-tuning data composition](#). *CoRR*, abs/2310.05492.
- Nan Du, Yanping Huang, Andrew M. Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P. Bosma, Zongwei Zhou, Tao Wang, Yu Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen S. Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V. Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. 2022. [Glam: Efficient scaling of language models with mixture-of-experts](#). In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 5547–5569. PMLR.
- Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. [Alpaca-farm: A simulation framework for methods that learn from human feedback](#). *CoRR*, abs/2305.14387.

- Jacob Eisenstein, Chirag Nagpal, Alekh Agarwal, Ahmad Beirami, Alex D’Amour, Dj Dvijotham, Adam Fisch, Katherine A. Heller, Stephen Pfohl, Deepak Ramachandran, Peter Shaw, and Jonathan Berant. 2023. [Helping or herding? reward model ensembles mitigate but do not eliminate reward hacking](#). *CoRR*, abs/2312.09244.
- William Fedus, Barret Zoph, and Noam Shazeer. 2022. [Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity](#). *J. Mach. Learn. Res.*, 23:120:1–120:39.
- Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, Andy Jones, Sam Bowman, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Nelson Elhage, Sheer El Showk, Stanislaw Fort, Zac Hatfield-Dodds, Tom Henighan, Danny Hernandez, Tristan Hume, Josh Jacobson, Scott Johnston, Shauna Kravec, Catherine Olsson, Sam Ringer, Eli Tran-Johnson, Dario Amodei, Tom Brown, Nicholas Joseph, Sam McCandlish, Chris Olah, Jared Kaplan, and Jack Clark. 2022. [Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned](#). *CoRR*, abs/2209.07858.
- Leo Gao, John Schulman, and Jacob Hilton. 2023. [Scaling laws for reward model overoptimization](#). In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 10835–10866. PMLR.
- Çağlar Gülçehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. [Reinforced self-training \(rest\) for language modeling](#). *CoRR*, abs/2308.08998.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [Lora: Low-rank adaptation of large language models](#). In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.
- Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. [Adaptive mixtures of local experts](#). *Neural Comput.*, 3(1):79–87.
- Michael I. Jordan and Robert A. Jacobs. 1994. [Hierarchical mixtures of experts and the EM algorithm](#). *Neural Comput.*, 6(2):181–214.
- Pei Ke, Bosi Wen, Zhuoer Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, Jie Tang, and Minlie Huang. 2023. [Critiquellm: Scaling llm-as-critic for effective and explainable evaluation of large language model generation](#). *CoRR*, abs/2311.18702.
- Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. [Gshard: Scaling giant models with conditional computation and automatic sharding](#). In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.
- Ziniu Li, Tian Xu, Yushun Zhang, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. 2023. [Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models](#). *CoRR*, abs/2310.10505.
- Xiao Liu, Xuanyu Lei, Shengyuan Wang, Yue Huang, Zhuoer Feng, Bosi Wen, Jiale Cheng, Pei Ke, Yifan Xu, Weng Lam Tam, Xiaohan Zhang, Lichao Sun, Hongning Wang, Jing Zhang, Minlie Huang, Yuxiao Dong, and Jie Tang. 2023. [Alignbench: Benchmarking chinese alignment of large language models](#). *CoRR*, abs/2311.18743.
- Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. 2018. [Modeling task relationships in multi-task learning with multi-gate mixture-of-experts](#). In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 1930–1939. ACM.
- Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. [Webgpt: Browser-assisted question-answering with human feedback](#). *CoRR*, abs/2112.09332.
- OpenAI. 2023. [GPT-4 technical report](#). *CoRR*, abs/2303.08774.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. [Training language models to follow instructions with human feedback](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.
- Shanghaoran Quan. 2024. [Automatically generating numerous context-driven sft data for llms across diverse granularity](#). *CoRR*, abs/2405.16579.
- Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. 2022. [DeepSpeed-moe: Advancing mixture-of-experts inference and training to power next-generation AI](#)

- scale. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 18332–18346. PMLR.
- Prajit Ramachandran and Quoc V. Le. 2019. [Diversity and depth in per-example routing models](#). In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.
- Alexandre Ramé, Kartik Ahuja, Jianyu Zhang, Matthieu Cord, Léon Bottou, and David Lopez-Paz. 2023a. [Model ratatouille: Recycling diverse models for out-of-distribution generalization](#). In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 28656–28679. PMLR.
- Alexandre Ramé, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. 2023b. [Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- Alexandre Ramé, Nino Vieillard, Léonard Hussenot, Robert Dadashi, Geoffrey Cideron, Olivier Bachem, and Johan Ferret. 2024. [WARM: on the benefits of weight averaged reward models](#). *CoRR*, abs/2401.12187.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. [Proximal policy optimization algorithms](#). *CoRR*, abs/1707.06347.
- Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. [Outrageously large neural networks: The sparsely-gated mixture-of-experts layer](#). In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.
- Sheng Shen, Le Hou, Yan-Quan Zhou, Nan Du, S. Longpre, Jason Wei, Hyung Won Chung, Barret Zoph, William Fedus, Xinyun Chen, Tu Vu, Yuexin Wu, Wuyang Chen, Albert Webson, Yunxuan Li, Vincent Zhao, Hongkun Yu, Kurt Keutzer, Trevor Darrell, and Denny Zhou. 2023. [Mixture-of-experts meets instruction tuning: a winning combination for large language models](#). *CoRR*, abs/2305.14705.
- Charlie Snell, Ilya Kostrikov, Yi Su, Sherry Yang, and Sergey Levine. 2023. [Offline RL for natural language generation with implicit language Q learning](#). In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.
- Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. 2023. [Preference ranking optimization for human alignment](#). *CoRR*, abs/2306.17492.
- Trevor Standley, Amir Zamir, Dawn Chen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. 2020. [Which tasks should be learned together in multi-task learning?](#) In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9120–9132. PMLR.
- Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. [Learning to summarize with human feedback](#). In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.
- Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. 2023. [Pandalm: An automatic evaluation benchmark for LLM instruction tuning optimization](#). *CoRR*, abs/2306.05087.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.
- Cheng Wen, Xianghui Sun, Shuaijiang Zhao, Xiaoquan Fang, Liangyu Chen, and Wei Zou. 2023. [Chathome: Development and evaluation of a domain-specific language model for home renovation](#). *CoRR*, abs/2307.15290.
- Fuzhao Xue, Ziji Shi, Futao Wei, Yuxuan Lou, Yong Liu, and Yang You. 2022. [Go wider instead of deeper](#). In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 8779–8787. AAAI Press.
- Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023. [RRHF: rank responses to align language models with human feedback without tears](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, December 10-16, 2023, New Orleans*.
- Yuanzhao Zhai, Han Zhang, Yu Lei, Yue Yu, Kele Xu, Dawei Feng, Bo Ding, and Huaimin Wang. 2024.

Uncertainty-penalized reinforcement learning from human feedback with diverse reward lora ensembles. *CoRR*, abs/2401.00243.

Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Qin Liu, Yuhao Zhou, Limao Xiong, Lu Chen, Zhiheng Xi, Nuo Xu, Wenbin Lai, Minghao Zhu, Cheng Chang, Zhangyue Yin, Rongxiang Weng, Wensen Cheng, Haoran Huang, Tianxiang Sun, Hang Yan, Tao Gui, Qi Zhang, Xipeng Qiu, and Xuanjing Huang. 2023. [Secrets of RLHF in large language models part I: PPO](#). *CoRR*, abs/2307.04964.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. 2023. [Least-to-most prompting enables complex reasoning in large language models](#). In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. 2019. [Fine-tuning language models from human preferences](#). *CoRR*, abs/1909.08593.

A Additional Related Work

A.1 Reinforcement Learning with Human Feedback

Reinforcement Learning with Human Feedback (RLHF) is a foundational method for fine-tuning language models to align with human preferences. RLHF has been applied to a variety of tasks, including text summarization (Stiennon et al., 2020) and improving the helpfulness and harmlessness of language models (Bai et al., 2022). In particular, InstructGPT (Ouyang et al., 2022) employs a three-step RLHF process that includes a supervised learning technique and the PPO algorithm (Schulman et al., 2017), which has proven to be effective for ChatGPT. Despite its success, RLHF encounters several challenges, such as low sample efficiency (Snell et al., 2023; Gülçehre et al., 2023) and overoptimization (Gao et al., 2023). Since we require no additional data to improve performance and mitigate overoptimization, our method works in both two aspects.

RLHF heavily depends on reward modeling as a proxy for human preferences. Recent research has attempted to bypass the reward modeling step (Yuan et al., 2023; Song et al., 2023). Specifically, Direct Policy Optimization (DPO) aims to refine policies by classifying human preference data without reward models. Although this

method is simpler to implement and offers training stability, more recent studies reveal several advantages of using reward models. Investigations into the robustness of reward-model-based strategies suggest they are more resistant to overfitting due to the limitations of KL regularization (Azar et al., 2023). Moreover, in comparison to DPO, reward-model-based RLHF shows great advantages on out-of-preference samples (Li et al., 2023).

A.1.1 the Overoptimization Problem of RMs

As the learned reward model is only a proxy for the true reward function, optimizing it may not always result in an improvement according to true human preferences. In practice, optimizing a (fixed) learned reward model almost always leads to improvement according to this learned reward model but only improves according to the *true* reward model (i.e., humans) for some initial period, after which performance often begins to regress. This phenomenon is referred to as *overoptimization*.

B Additional Experiment Setup

B.1 Hyperparameters

Since in different training phases there are different sets of model parameters that need to be trained, we use different learning rates to better adapt to these three phases and allocate different proportions of training set for Phase 1 and Phase 3. The settings are shown in Table 4.

	learning rate	proportion
Phase 1	1×10^{-7}	60%
Phase 2	5×10^{-5}	—
Phase 3	1×10^{-6}	40%

Table 4: The hyperparameters used in different training phases.

We set the minibatch size to 1 and conduct an evaluation on the validation set every 100 training steps. If the best result on the validation set does not improve for 20 consecutive evaluations, we implement early stopping and use the best-performing model for the next training phase or the final testing if it is already Phase 3.

The settings for the LoRA components are presented in Table 5, applicable to both our LoRA experts and the LoRA ensembles used in the baseline comparisons.

rank	32
lora alpha	32
target modules	q_proj & v_proj
lora dropout	0.05

Table 5: LoRA configurations.

The PPO hyperparameters and generation hyperparameters are shown in Tables 6 and 7, respectively.

learning rate	1.41×10^{-5}
cosine annealing scheduler	1×10^{-7}
PPO steps	3,000
batch size	32
KL penalty	0.1
clipping range	0.2
clipping value	0.2
gamma	1.0
lambda	0.95

Table 6: PPO hyperparameters.

max instruction length	1,024
max new tokens (answer length)	1,024
top-p	0.8
temperature	1.0

Table 7: Generation hyperparameters.

All experiments are run on a single machine with eight NVIDIA A100 80G GPUs, and we use Adam optimizer for the optimization process.

B.2 Ensemble Creation for Baseline Methods

To create an ensemble for the Mean, WCO, and UWO baselines, under Coste et al. (2024); Eisenstein et al. (2023) guidance, we train a fixed number of proxy reward models using identical data and hyperparameters. Each model, however, is initialized with different random seeds. This leads to variations in the random initialization of the scalar reward head that is added on top of the pre-trained language model, as well as in the data shuffling order. Table 5 presents the LoRA parameters for

these models. We have trained an ensemble of five reward models, aligning with the configurations used in previous works. This number is also comparable to the number of LoRA experts in our model, which ranges from five to six, depending on the numbers of capability points allocated under a category.

B.3 Optimization Method

B.3.1 Best-of- n Sampling

Best-of- n (BoN) sampling, also called rejection sampling, is a simple inference-time optimization method (Nakano et al., 2021; Ouyang et al., 2022). For a given prompt, n responses are generated from the policy model, and the answer with the highest proxy reward model score is returned. To evaluate the degree of optimization, the KL divergence⁴ is defined analytically as a function of n :

$$\text{KL}_{\text{bon}} = \log n - \frac{n-1}{n} \quad (11)$$

In our experiments, we evaluate BoN for a maximum of $n_{\text{max}} = 60,000$ samples, which roughly equals 10 nats of KL.

B.3.2 Proximal Policy Optimization

Proximal Policy Optimization (PPO) (Schulman et al., 2017) is a policy-gradient-based online reinforcement learning method that maximizes a given reward function by repeatedly performing small incremental updates to the policy. PPO is the standard algorithm used in fine-tuning language models based on human feedback (Ouyang et al., 2022; Bai et al., 2022; Stiennon et al., 2020; Zheng et al., 2023). When using PPO to fine-tune a language model, a KL penalty term is added during the reward calculation to regularize the policy by preventing it from deviating far from the initial policy:

$$R^{\text{PPO}}(q, r) = R(q, r) - \beta \text{KL}_{\text{PPO}} \quad (12)$$

where π^{PPO} is the policy being optimized and π^{init} is the initial (pre-trained) language model.

The naive way to calculate KL divergence between the PPO-optimized policy π^{PPO} and the initial model is as follows:

⁴A recent work (Beirami et al., 2024) claims to have proved that this boundary is not so accurate and provided a tighter boundary defined by the binary entropy function. At present, they have not been widely recognized, and even if they are correct, our experimental conclusions are still valid in terms of trends.

$$\text{KL}_{\text{PPO}}(\pi^{\text{PPO}}, \pi^{\text{init}}) = E_{(q,r) \sim \pi^{\text{PPO}}} \left[\log \frac{\pi^{\text{PPO}}(r|q)}{\pi^{\text{init}}(r|q)} \right] \quad (13)$$

However, this estimator suffers from high variance and may yield negative values. Consequently, we employ the following estimator (Coste et al., 2024):

$$\text{KL}_{\text{PPO}}(\pi^{\text{PPO}}, \pi^{\text{init}}) = E_{(q,r) \sim \pi^{\text{PPO}}} \left[\frac{1}{2} \left(\log \frac{\pi^{\text{PPO}}(r|q)}{\pi^{\text{init}}(r|q)} \right)^2 \right] \quad (14)$$

We train a total of 3,000 PPO steps, and the PPO parameters are shown in Table 6.

B.3.3 Generative Baseline

Apart from RM-based methods, if only the preference order of response pairs needs to be given, a simple way is to call the public language model API through zero-shot or few-shot. It can also be combined with BoN or DPO to optimize LLM. For a more comprehensive evaluation, in preference consistency experiments, we have added the GPT-4 baseline.

- **GPT-4 (OpenAI, 2023)** We use the most advanced gpt-4-1106-preview⁵ as the evaluator to evaluate a pair of preference data. Each time, we swap the positions of the responses and make two requests, and we will re-request until we get identical results from two requests as the final results.

We try to find well-prompted instruction and use both zero-shot and one-shot to evaluate and do not use few-shot for comparison, for we find it will make the context much longer and the result is not better than one-shot. There are also some other generative evaluation methods. Since they have reported in their papers that their performance is just similar with (Ke et al., 2023) GPT-4 or even worse than (Wang et al., 2023) GPT-4, we just use GPT-4 to represent these methods.

C Discussion on Model Interpretability

The single traditional RM only outputs the final reward score and we have no clues about why a

⁵<https://platform.openai.com/docs/models/gpt-4-and-gpt-4-turbo> We called the API in January 2024.

response gets its score, which makes it have poor interpretability. However, our DMoERM first learns latent embedding at different capability points under specific tasks, and the final score is learned from and based on an ensemble of embeddings of different capability points. By using the FCN trained in Phase 2 to act on the corresponding hidden embedding, we can know the score on each capability point of a response. In this way, we can identify which aspects of a good response are effective and which aspects of a poor response are not. Similarly, if the overall consistency of the model is not satisfactory, we can also identify the problem and prescribe targeted solutions by analyzing the performance of each single capability point.

The method that our DMoERM checks the task of inputs, works on different capability points, and aggregates latent embeddings of each expert to obtain the final reward is just like the *Chain-of-Thought* of the reward models, which can enhance the reasoning (give the reward under the context) ability of reward models as well as the interpretability of the reward models.

D Additional Experimental Result

D.1 Larger Model Performance

We further apply the preference consistency evaluation to the Qwen7B-Chat and Qwen14B-Chat models. Except for adjusting the learning rate to adapt to different parameter sizes ($[5e-8, 2.5e-5, 5e-7]$ for the 7B model and $[3e-8, 1.5e-5, 3e-7]$ for the 14B model, during the three training phases), the other parameter settings and dataset partitioning are consistent with the paper, and the specific experimental results are shown in the Table 10.

We find that our DMoERM achieved the best results among all categories and all model sizes (in bold), while DMoERM-7B achieved the second best results (is underlined). Meanwhile, in the comparison between our DMoERM in different parameter sizes, the final results for different categories improve by about 1.9% for 7B and 3.6% for 14B against the smallest 1.8B model, indicating that our model structure can benefit from scaling up the size of parameters. The increase in consistency rate is not too significant when scaling up due to the presence of certain noise in the labeled data (with observed 74% consistency among human annotators), which leads to it being challenging to further improve when consistency rates near the 70% mark. Meanwhile, due to the amount of train-

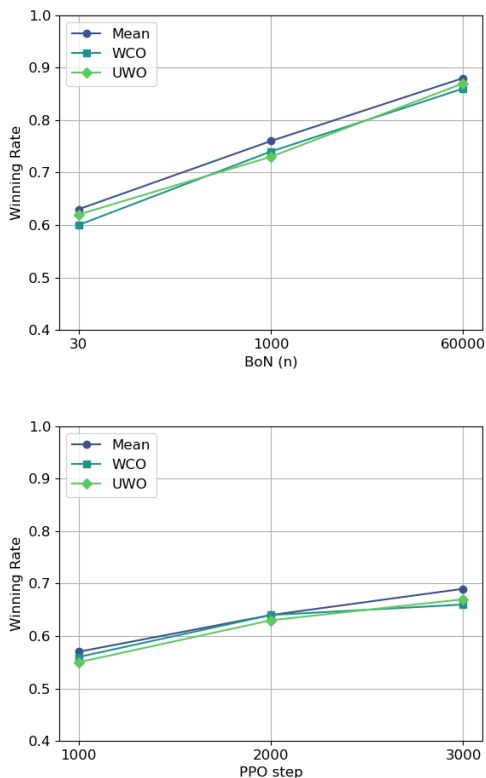
ing data is fixed, the larger models may not have been sufficiently trained, which may also be a factor contributing to the less significant improvement.

D.2 Human Evaluation

In the process of BoN and PPO optimization, we both set three checkpoints ($n = 30, 1000, 6000$ for BoN and 1000, 2000, 3000 PPO steps for PPO) and use 100 prompts at each checkpoint for human evaluation. Specifically, for the different outputs obtained from these prompts under optimizations from different RMs, we require annotators to sort them according to their preferences, and we record the winning rates of our model against other ensemble methods in Figure 6.

We have two findings: Firstly, compared to BoN, the comparison improvement under PPO optimization is slower. Secondly, in both optimization experiments, the winning rates of our model increase steadily as optimization progresses, ultimately reaching approximately 87% for BoN and 68% for PPO.

Figure 6: The winning rates of DMoERM against other ensemble methods in human evaluation.



D.3 OOD Optimization Evaluation

In contrast to several previous works (Gao et al., 2023; Coste et al., 2024; Eisenstein et al., 2023; Zhai et al., 2024) where the RM training set, optimization set, and evaluation set are typically independently and identically distributed (IID), our following experiments involve using out-of-distribution (OOD) RM training sets in PPO I and OOD evaluation sets in PPO II (since BoN is a training-free optimization method used in the inference stage, the optimization prompt set is directly the evaluation prompt set, so we only conduct OOD optimization experiments on PPO). We use AlignBench (Liu et al., 2023), a publicly and popularly used alignment benchmark in Chinese, to evaluate the effectiveness of different models. We use the DMoERM-w/o-Outer for comparison and set the reward models and policy models in the same model sizes of 7B. Due to the models being larger, we compare the optimization results of different RMs using GPT-4 as the referee, which we believe has the most sufficient capability to evaluate the generation results and is naturally fair. The RM training set, the optimization set, and the evaluation set used in PPO I and PPO II are represented in Table 8 and the experimental results are shown in Figure 7.

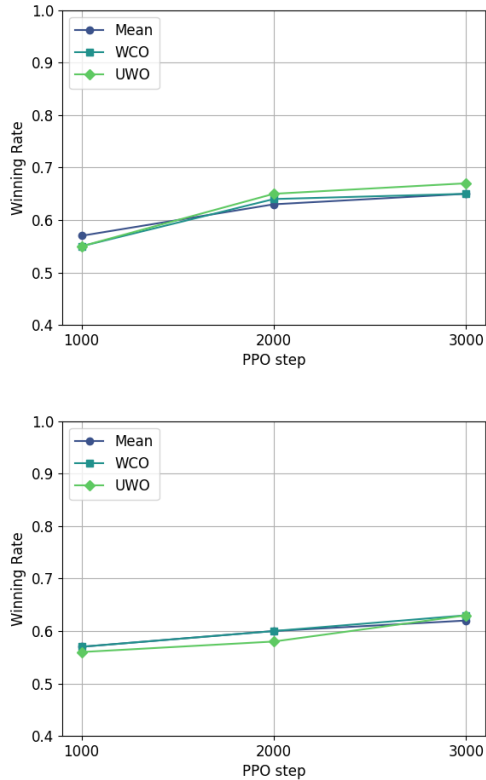
	RM training	optimization	evaluation
I	our dataset	AlignBench	AlignBench
II	our dataset	our dataset	AlignBench

Table 8: The datasets used in different periods of PPO I and PPO II. Note that the datasets are randomly sampled from overall datasets and have no intersection in different periods.

We have two significant findings: First, PPO I and PPO II show consistent trends, both steadily improving the winning rate against the baseline method during the optimization process. Second, In PPO II, the winning rate initially increased more rapidly (before 1,000 steps), while the growth in PPO I was more stable and persistent in all optimization periods.

Intuitive analysis: Due to the fact that in PPO II, the optimization set and training set are IID, the reward model can provide a clear reward signal during optimization, enabling training to quickly achieve positive optimization; conversely, In PPO I, while the optimization and evaluation sets are IID, they are OOD with the training set, making

Figure 7: The winning rates of DMoERM against other ensemble methods in OOD optimization.



the reward signal is not such clear and strong, but advantages can gradually be reflected in the continuous optimization process.

D.4 Annotation Quantitative Experiment

Since our method requires more annotation costs, to provide a more comprehensive evaluation of our method, we quantified the annotation consumption and maintained consistency between our method and the baseline method in terms of annotation consumption. In this experiment, we compared DMoERM-w/o-Outer with other baseline models in all three model sizes on the entire dataset. Due to the need that our model requires each piece of data 5 times additional annotations on intent conformity, logic, conversational sense, content richness, and readability, we only randomly sample one-sixth of the data to train our model, while other models use all the data to control the consistency of the number of annotations. The experimental results are shown in Table 9.

We also recorded the performance of our DMoERM-w/o-Outer at the end of the training Phase 1 as DMoERM-Phase1. At this point, it is equivalent to a Single RM trained with only one-

sixth of the data, so the effect is poor. But when the three training phases ended (denoted as DMoERM-Phase3), its performance improved by 9.3 percentage points on average and exceeded all baseline models in three model sizes.

Note that this experiment is not very fair because in our empirical experiments, we found that our annotation method only reduces the annotator’s speed by about 10%, but can increase the annotation consistency by 5 percentage points. In the experiment, we only used one-sixth of the data, but exceeded all the baseline methods by at least 1.4 percentage points, indicating the compelling advantages of our method.

D.5 Qualitative Sample

Throughout our experiments, quantitative metrics play a pivotal role as they enable the rapid and comprehensive evaluation of various methods. In this section, we offer a concise insight into the qualitative aspect of the approaches discussed in this study. Specifically, for a given prompt, responses from the ultimate policies ($n = 60,000$ for BoN and 3,000 steps for PPO) of each method in Figure 5 are provided.

The main findings are as follows. Firstly, the brevity of BoN’s response is due to the differing manner in which policy optimization occurs compared to PPO. Secondly, there are indications of failure and overoptimization in the other ensemble RMs. As shown in Table 11, for BoN, this manifests in inadequate answers. For PPO, this means bad answers that are incorrectly long and repetitive. These are obvious signs of overoptimization. Lastly, we observe that even in scenarios where other ensemble RMs struggle, our DMoERM is capable of yielding robust qualitative outcomes.

Model Size	Single	Mean	WCO	UWO	DMoERM-Phase1	DMoERM-Phase3
1.8B	58.2	62.4	62.6	62.6	54.8	64.1
7B	59.2	63.6	63.5	63.6	55.9	65.3
14B	60.5	64.9	65.1	65.1	57.2	66.5

Table 9: The results of labeling quantitative experiment. We record the performance of our model at the end of the training Phase 1 as DMoERM-Phase1 and at the end of three training phases as DMoERM-Phase3.

Model Size	Method	text	role	okqa	skqa	chat	others	overall
	Zero-shot GPT-4	58.9	56.1	64.7	63.5	54.8	60.1	59.5
	One-shot GPT-4	61.4	59.9	66.5	66.2	58.4	61.7	62.3
1.8B	Single	58.1	57.6	60.4	62.0	54.3	54.4	58.2
	Mean	63.3	62.0	65.2	66.3	58.5	56.6	62.4
	WCO(min)	63.6	62.3	65.5	66.1	59.1	56.7	62.6
	UWO(mean minus std)	63.4	61.8	65.4	66.5	59.0	56.6	62.6
	DMoERM-w/o-Outer DMoERM	<u>68.3</u> 71.9	<u>66.5</u> 68.2	<u>70.1</u> 73.8	<u>71.3</u> 74.2	<u>63.2</u> 66.4	<u>61.3</u> 63.0	<u>67.0</u> 70.7
7B	Single	59.1	58.7	61.7	63.2	55.3	55.3	59.2
	Mean	64.7	63.1	66.6	67.4	59.4	57.8	63.6
	WCO(min)	64.4	63.4	66.6	67.2	59.3	57.7	63.5
	UWO(mean minus std)	64.4	63.4	66.7	67.0	59.3	57.8	63.6
	DMoERM-w/o-Outer DMoERM	<u>70.0</u> 73.2	<u>67.9</u> 69.6	<u>71.6</u> 75.2	<u>72.9</u> 75.5	<u>64.5</u> 67.6	<u>62.4</u> 64.3	<u>68.4</u> 72.1
14B	Single	60.2	59.9	62.7	64.6	56.1	56.6	60.5
	Mean	66.0	64.7	68.1	68.7	60.6	58.7	64.9
	WCO(min)	66.3	64.6	68.4	68.5	60.6	58.8	65.1
	UWO(mean minus std)	66.0	64.7	68.1	68.6	60.6	58.8	65.1
	DMoERM-w/o-Outer DMoERM	<u>70.8</u> 74.7	<u>69.4</u> 71.2	<u>73.0</u> 76.7	<u>73.9</u> 77.0	<u>65.7</u> 69.1	<u>63.7</u> 65.5	<u>69.6</u> 73.2

Table 10: The consistency of different model sizes with human preferences. In each model size, the best performance is in bold and the second best is underlined. We can observe that our DMoERM and DMoERM-w/o-Outer consistently achieve the best and second-best results at different model sizes.

Table 11: A set of example answers to an evaluation query.

Query	如何简洁地描写点心的香味从盒子里飘出	How to briefly describe the aroma of the dim sum floating out of the box
Single (BoN)	当你打开点心盒时，那种香气会像风筝一样飘出来，让你的味蕾仿佛跳上了云朵，让你的心情变得愉快。	When you open the snack box, the aroma will float out like a kite, making your taste buds seem to jump on the clouds and making you feel happy.
Mean (BoN)	打开点心盒，那怡人的香气就像一束光照亮了整个房间，仿佛能够看到它们在空气中跳跃，让人不禁垂涎三尺。	Open the box of refreshments, the pleasant aroma lights up the whole room like a beam of light, as if you can see them jumping in the air, which makes people can't help salivating.
WCO (BoN)	一股香气从盒子里飘出，那是点心的香味，让人的口水不禁流了出来，欲望在空气中凝结。	A fragrance floats out of the box, which is the aroma of refreshments, making people's saliva flow out, and desire condenses in the air.
UWO (BoN)	打开盒子的一刹那，点心的香气就如同一缕春风，绕梁三日，飘散在空气中，令人垂涎欲滴。	The instant I opened the box, the aroma of the dessert spread in the air like a wisp of spring breeze lingering around for three days, which made people's mouths water.
DMoERM (BoN)	点心香气从盒中四溢，诱人味蕾，满室生香，令人陶醉。	The aroma of dim sum overflows from the box, tempts the taste buds, fills the room with fragrance, and intoxicates people.
Single (PPO)	点心盒子里隐藏着一种难以言表的诱惑，那是甜点所独有的甜美芳香。随着盒盖的轻启，这股香味仿佛被赋予了生命，它不再受限于那狭小的空间，开始缓缓地向四周扩散。它如同一个害羞的少女，初出茅庐，带着些许的迟疑和羞涩，一点一点地探出头来。先是一丝若有若无的甜腻，让人心生疑惑，随即又增强了几分，变得明确而肯定。它似乎在空气中跳跃、旋转，与每一个嗅觉细胞亲密接触，勾起了人们内心深处对甜蜜的向往。	The dim sum box hides an ineffable temptation, a unique sweet aroma that belongs exclusively to desserts. With the lightest lifting of the lid, this fragrance seems to be endowed with life, no longer confined to its narrow space and begins to slowly diffuse outwards. Like a shy maiden emerging from her shell, it tentatively and bashfully peeks out, little by little. At first, it's a mere hint of sweetness that leaves one puzzled, but it gradually strengthens and becomes clearer and more definite. It seems to dance and swirl in the air, intimately caressing every olfactory cell, evoking a deep-seated yearning for sweetness within people's hearts.

Continued on the next page

Query	如何简洁地描写点心的香味从盒子里飘出	How to briefly describe the aroma of the dim sum floating out of the box
Mean (PPO)	盒子轻轻打开，点心的香气瞬间弥漫。那是糯米的温润，豆沙的甜蜜，还有一丝丝清香的茶叶味。一阵阵，一股股，绵绵不绝，像是诗一般的甜美旋律，缭绕在空气中，点心的清香让人垂涎欲滴，心生欢喜。	The box opens gently, and the aroma of the dessert instantly spreads. It is the warmth of glutinous rice, the sweetness of red bean paste, and a trace of fresh tea flavor. Wave upon wave, continuous and unending, like a poetic sweet melody lingering in the air, the fragrance of the dessert makes people salivate and feel delighted.
WCO (PPO)	点心的香味从盒子里缓缓飘出，一次又一次地扑鼻而来，像是阵阵潮汐，持续不断地冲击着味蕾。每一波香气都带着诱人的酥脆和甜蜜，让人忍不住想要尝一口。这就是点心的魅力，它的香味像是魔法，一次又一次地吸引着人们。	The aroma of the dim sum slowly wafts out of the box, hitting the nostrils over and over again, like waves of the tide, constantly impacting the taste buds. Each wave of fragrance brings tempting crunch and sweetness, making it hard to resist taking a bite. This is the charm of dim sum, its fragrance is like magic, attracting people over and over again.
UWO (PPO)	点心的香气，如同精灵的歌唱，娓娓地从盒子的缝隙中溢出。一次，两次，每一次都像是在空气中划出一个甜美的弧线。轻轻一闻，那香气又再次被唤醒，从盒子里流淌出来，一次又一次，像是永无止境的旋律，让人沉醉。	The aroma of the dim sum, like the singing of elves, spills out of the cracks of the box in a continuous stream. Once, twice, each time it seems to draw a sweet arc in the air. With a gentle sniff, the aroma is awakened again, flowing out of the box, over and over again, like an endless melody that intoxicates the senses.
DMoERM (PPO)	点心的香气如轻烟，缓缓从精致的盒子中逸出，弥漫在空气中。那甜美的芬芳，仿佛有魔力般，无声地诱惑着每一个嗅觉细胞，让人不由自主地靠近，沉醉在这份难以抗拒的美味预告中。	The aroma of the dim sum, like light smoke, escapes slowly from the exquisite box and pervades the air. That sweet fragrance, as if magical, silently tempts every olfactory cell, drawing people irresistibly closer, intoxicated by this irresistible preview of deliciousness.

data_id

12822

类别

文本创作

对话历史

<用户>: 模仿“天街小雨润如酥，草色遥看近却无。最是一年春好处，绝胜烟柳满皇都。”为例子做一首诗。并以天街小雨润如酥开头，形容睡完二字，回家回笼觉喜悦心情的诗。

<模型>: 天街小雨润如酥，睡意渐浓似梦中。
草色青翠近在眼，微风拂面犹带香。

红尘嚣嚣心不动，归家回笼觉方宁。
梦里花开如绚烂，身心宁静任自然。

日出东方曙色明，回家的路途思绪平。
梦中花开如绚烂，身心宁静自然清。

红尘喧嚣难入眠，回家回笼觉中安。
梦里花开如绚烂，身心宁静自在闲。

日落西山红霞飞，回家的步伐心自闲。
梦中花开如绚烂，身心宁静自然清。

天街小雨润如酥，归家的路上自在行。
梦中花开如绚烂，身心宁静自然清。

<用户>: 模仿“天街小雨润如酥，草色遥看近却无。最是春好处，绝胜烟柳满皇都。”为例子做一首诗。并以天街小雨润如酥开头，形容睡完儿子上学，回家睡回笼觉喜悦心情的诗。

<模型>: 天街小雨润如酥，晨曦微凉意渐浓。
草色青翠近在眼，微风拂面犹带香。

详情上一步骤操作 回家睡回笼觉喜悦心情诗

当前轮询问

用睡回笼觉四个字，写一首藏头诗。

回复1

睡意朦胧似梦中，
回笼觉里身心宁。
静享安宁时光美，
恬淡岁月温柔情。

回复2

睡意频催夜色催，
回笼锦绣卧芳菲。
笼禽与我同清梦，
觉来红日已三竿。

意图符合程度

回复1更好 回复2更好

表达能力

回复1更好 回复2更好

可读性

回复1更好 回复2更好

内容丰富性

回复1更好 回复2更好

逻辑性

回复1更好 回复2更好

综合来看，你认为哪个更好?

回复1更好 回复2更好

提交

通过 API 使用 使用 Gradio 构建

Figure 8: The screenshot of the interface for the annotation consistency experiment. The content in the red box is the preference selection for the five capability points. We find that removing them would lead to a drop of 5 percentage points in the overall consistency of preferences.

Category	Capability Points	Chinese
roleplay	personality and emotional investment	人设和情感带入
	conversational sense	对话感
	empathy ability	共情能力
	manifestation of relationship traits	关系特点体现
	personalized characteristic expression	个性化特征体现
	content richness	内容丰富性
chitchat	conversational sense	对话感
	proactivity	主动性
	emotion expression	情感表达
	empathy ability	共情能力
	content richness	内容丰富性
subjective knowledge QA	convincing ability	说服力
	logic	逻辑性
	viewpoint richness	观点丰富度
	breadth of knowledge	知识面广度
	question specific	问题针对性
objective knowledge QA	correctness	正确性
	objectiveness	客观程度
	reasoning ability	推理能力
	logic	逻辑性
	depth of knowledge	知识面深度
	question specific	问题针对性
text creation	intent conformity	意图符合程度
	expressiveness	表达能力
	readability	可读性
	content richness	内容丰富性
	logic	逻辑性
overall (for DMoERM-w/o-Outer)	intent conformity	意图符合程度
	logic	逻辑性
	conversational sense	对话感
	content richness	内容丰富性
	readability	可读性

Table 12: The capability points partitions for each task in our experiments.

Template	<p>给定提问和两个回复，判断回复1和回复2在【{point}】方面哪个更好。</p> <p>【提问】：{query}</p> <p>【回复1】：{response1}</p> <p>【回复2】：{response2}</p>
Example	<p>给定提问和两个回复，判断回复1和回复2在【对话感】方面哪个更好。</p> <p>【提问】：你觉得我要不要裸辞</p> <p>【回复1】：裸辞和被公司优化后离职有很大的不同，具体取决于个人情况和职业规划。以下是一些需要考虑的因素：</p> <p>1. 职业规划：如果你的职业规划中包括了一份长期的工作，那么裸辞可能会影响到你未来的职业发展。在这种情况下，你需要认真思考自己的职业规划，并考虑如何在不影响自己长远发展的前提下处理眼前的问题。2. 经济压力：离职后可能需要面对经济上的压力，例如房租、信用卡账单等。如果你的经济状况不是很好，可以先找一份短期工作，缓解一下经济压力。但是需要注意的是，短期工作可能无法满足你的职业发展需求，也不利于建立长期的职业规划。3. 心理压力：离职后可能会带来一定的心理压力，例如失去工作、社交圈子等方面的变化等。如果你感到无法承受这种压力，可以考虑暂时休息或寻求心理咨询帮助。4. 个人成长：在一份工作中，可以不断学习和成长，积累经验和技能，这对于未来的职业发展也是非常有帮助的。因此，即使面临经济压力，也需要尽可能地保持积极的心态，寻求更好的发展机会。</p> <p>总之，裸辞和被公司优化后离职都需要慎重考虑，需要综合考虑自己的职业规划、经济压力、心理压力等多方面因素。如果决定离职，最好提前做好规划，尽可能减少离职对自己的影响。</p> <p>【回复2】：考虑到你的困扰，是否选择裸辞确实是一个需要深思熟虑的决定。在回答这个问题之前，我想先了解一下你目前的经济状况，是否有足够的储蓄来应对离职后可能的经济压力？另外，你的职业规划中，是否有更合适的工作机会在等待你？还有，你目前的心理状况能否承受裸辞带来的压力？</p> <p>裸辞的好处是，你可以暂时摆脱目前工作环境中的压力，给自己一些休息和反思的时间。然而，它的风险在于可能会给你带来经济上的压力，并且在寻找下一份工作的过程中，可能会出现职业空窗期，这对你未来的职业发展可能会有一些影响。</p> <p>所以，我建议你在做决定之前，尽量找一些职业咨询专家进行咨询，他们可以根据你的具体情况提供更具体的建议。同时，如果你感到压力很大，也可以考虑寻求心理咨询的帮助。</p> <hr/> <p>在对话感方面，回复2比回复1更好。</p> <p>回复1虽然详细列出了一系列关于是否应裸辞的因素让提问者考虑，但更像是一篇分析文章或建议清单，缺少对话的互动性和针对性。此外，回复1没有直接回应提问者的情感或具体状况，而是给出了一般性的建议，这在对话中可能显得有些疏远。</p> <p>相比之下，回复2首先从提问者的困扰出发，提出了一些具体问题来了解提问者的状况，这种方式更贴近对话的形式，有助于建立共鸣和理解。接着，回复2也分析了裸辞的利弊，并在最后给出了具体的建议，如咨询职业咨询专家或寻求心理咨询帮助。这些建议更具体，也更具操作性，显示出回复者对提问者问题的深度理解。</p> <p>所以，回复2在对话感方面表现更优秀。</p>

Table 13: The Chinese original version of the prompt for calling public LLM API to get the comparison on a single capability point. Note that we do not attach historical information, for it can greatly reduce the cost and is sufficient to achieve satisfactory results in our experiments. We determine the model's choice by identifying whether the number 1 or 2 appears first in its response.

Template	<p>Given a query and two responses, determine which of response 1 and response 2 is better in terms of [{point}].</p> <p>[Query]: {query}</p> <p>[Response 1]: {response1}</p> <p>[Response 2]: {response2}</p>
Example	<p>Given a query and two responses, determine which of Response 1 and Response 2 is better in terms of [conversational sense].</p> <p>[Query]: Do you think I should quit my job without having another one in hand</p> <p>[Response 1]: Quitting a job without a backup plan and being laid off as part of a company’s restructuring are quite different and depend on personal circumstances and career planning. Here are some factors to consider: 1. Career Planning: If your career planning includes a long-term job, quitting abruptly might affect your future career development. In this case, you should carefully consider your career plans and how to address current issues without jeopardizing your long-term prospects. 2. Financial Pressure: After leaving a job, you may face financial pressures such as rent or credit card bills. If you are not in a good financial situation, you might consider finding a short-term job to alleviate financial stress. However, keep in mind that short-term jobs may not meet your career development needs and may not be conducive to establishing a long-term career plan. 3. Psychological Pressure: Resigning might bring certain psychological pressures, such as the loss of a job and changes in your social circle. If you feel unable to cope with this pressure, consider taking a break or seeking psychological counseling for support. 4. Personal Growth: In a job, you can continuously learn and grow, accumulating experience and skills, which is very beneficial for future career development. Therefore, even in the face of financial pressure, it is necessary to maintain a positive attitude and seek better development opportunities.</p> <p>In summary, quitting a job without a plan and being laid off due to company restructuring both require careful consideration, taking into account factors such as your career planning, financial pressure, and psychological stress. If you decide to leave, it is best to plan ahead to minimize the impact on yourself.</p> <p>[Response 2]: Considering your concerns, deciding whether to resign without another job offer in hand is indeed a decision that requires careful deliberation. Before answering this question, I’d like to first understand your current financial situation. Are you equipped with enough savings to handle the potential financial pressures after resigning? Additionally, are there more suitable job opportunities waiting for you in your career plan? And, are you mentally prepared to handle the stress that may come with resigning without a backup plan?</p> <p>The advantage of resigning without a backup plan is that you can temporarily escape the pressures of your current work environment, allowing yourself some time to rest and reflect. However, the risk is that it can lead to financial stress, and during the process of finding your next job, there may be a gap in your career which could impact your future professional development.</p> <p>Therefore, before making a decision, I suggest that you consult with career counseling experts who can provide more specific advice based on your situation. Also, if you are feeling significant stress, consider seeking help from a mental health professional.</p> <hr/> <p>In terms of conversational sense, Response 2 is better than Response 1. Response 1, while detailing a list of factors for the questioner to consider regarding whether to quit without a backup plan, reads more like an analytical article or a list of suggestions, lacking interactive and targeted dialogue. Moreover, Response 1 did not directly address the questioner’s emotions or specific circumstances; instead, it offered general advice, which may seem somewhat distant in a conversation. By contrast, Response 2 begins by addressing the questioner’s concerns and poses specific questions to understand their situation, which is closer to the form of a dialogue and helps to establish empathy and understanding. Then, Response 2 also analyzes the pros and cons of quitting without a plan and concludes with concrete suggestions, such as consulting career counseling experts or seeking mental health counseling. These suggestions are more specific and actionable, showing a deeper understanding of the questioner’s issue. Therefore, Response 2 performs better in terms of conversational sense.</p>

Table 14: The English translated version of the prompt for calling public LLM API to get the comparison on a single capability point. Note that we do not attach historical information, for it can greatly reduce the cost and is sufficient to achieve satisfactory results in our experiments. We determine the model’s choice by identifying whether the number 1 or 2 appears first in its response.