Routing to the Expert: Efficient Reward-guided Ensemble of Large Language Models

Keming Lu, Hongyi Yuan*, Runji Lin*
Junyang Lin, Zheng Yuan, Chang Zhou, Jingren Zhou
Alibaba Inc.
{lukeming.lkm,yuanhongyi.yhy,linrunji.lrj}@alibaba-inc.com
{junyang.ljy,yuanzheng.yuanzhen}@alibaba-inc.com
{ericzhou.zc,jingren.zhou}@alibaba-inc.com

Abstract

The complementary potential of Large Language Models (LLM) assumes off-the-shelf LLMs have heterogeneous expertise in a wide range of domains and tasks so that an ensemble of LLMs can achieve consistently better performance. Existing ensemble methods for LLMs mainly focus on reward model ranking of outputs, leading to significant computation overhead. To combat this issue, we revisit the complementary potential of LLMs and further elaborate it by mining latent expertise with off-the-shelf reward models. We propose ZOOTER, a reward-guided routing method distilling rewards on training queries to train a routing function, which can precisely distribute each query to the LLM with expertise about it. We also integrate a tag-based label enhancement to mitigate noise from uncertainty when using rewards as silver supervision. ZOOTER shows computation efficiency in inference as it only introduces minor computation overhead of a routing function compared with reward model ranking methods. We evaluate ZOOTER on a comprehensive benchmark collection with 26 subsets on different domains and tasks. ZOOTER outperforms the best single model on average and ranks first on 44% of tasks, even surpassing multiple reward model ranking methods.

1 Introduction

Large Language Models (LLMs) aligned with human preference rapidly emerge and are almost daily released (Touvron et al., 2023a,b; Anil et al., 2023; Bai et al., 2023). These off-the-shelf LLMs are further finetuned or aligned with human preference to be generalists (Xu et al., 2023; Touvron et al., 2023b,a) or specialists (Yuan et al., 2023a; Luo et al., 2023a,b; Roziere et al., 2023) for solving versatile tasks. It is worth noticing that LLMs are pretrained and aligned with various data, leading to diverse strengths and weaknesses in versatile downstream tasks (Jiang et al., 2023). Therefore, the ensemble of LLMs harnesses the complementary potential among them and may achieve better performance than a single best-on-average model across diverse tasks.

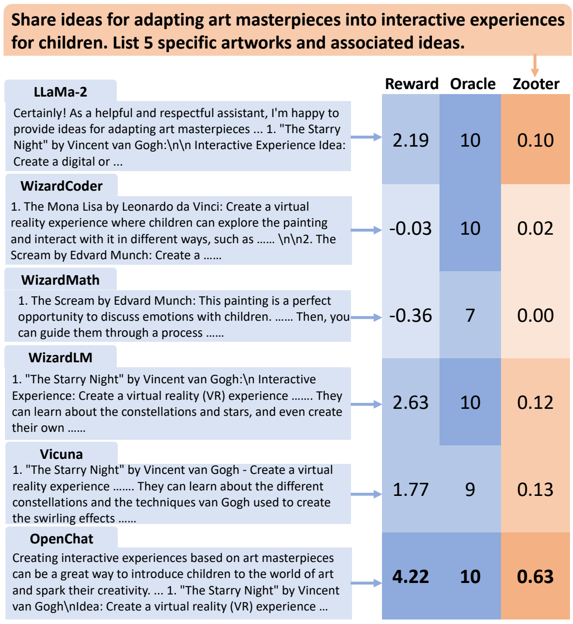

Figure 1: An example of the large language model ensemble. Reward model ranking marked in blue needs to generate responses from all models while ZOOTER routers the given query to the best model and only infers one model. This case is collected from the MT-Bench benchmark and we also present oracle judgements of each response.

One of the key challenges in the LLM ensemble is computation efficiency due to the large parameter size of existing LLMs. Previous research (Jiang et al., 2023; Shnitzer et al., 2023) provides solid methods to merge generation outputs of LLMs as an ensemble. Such methods require tremendous inference cost that makes it unscalable and thus not competitive to the best-on-average model under low-resource scenarios. To efficiently assemble off-the-shelf LLMs, we first dive deeper into the considerably straightforward but still understudied

*Work done during internship at Alibaba Inc.

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 1964–1974
June 16-21, 2024

assumption: *Off-the-shelf aligned LLMs, even for those aligned as "generalists", have heterogeneous expertise in a wide range of domains and topics*. However, analyzing the expertise of an LLM is also challenged as the latent expertise of LLMs is highly related to the pretrained and alignment data, which is very vague and inaccessible even for popular open-source LLMs such as LLAMA-2-CHAT (Touvron et al., 2023b) and WIZARDLM (Xu et al., 2023).

If this assumption strongly holds, off-the-shelf LLMs can be assembled efficiently by assigning queries to the model that is proficient in the query without additional inference costs on each model. Such an efficient routing strategy only requires inference cost for a single model for each query and the overhead cost of a much smaller query router. However, probing the detailed expertise of off-the-shelf LLMs and generating supervision for training routers also require annotations. Developing a data-efficient training method for routing queries is significantly understudied.

To combat these issues, we propose ZOOTER, a reward-guided query routing method for efficiently assembling off-the-shelf LLMs. ZOOTER obtains and enhances silver supervision from existing reward models (RM) for query router training and distributes queries in advance to "experts". As shown in Fig. 1, the reward distribution implies the oracle judgments and reveals a latent expertise between LLMs. And ZOOTER captures the expertise from reward distributions and provides query distribution during inference. Specifically, we first conduct a comprehensive study involving four groups of benchmarks across more than 26 subsets in various domains and tasks. We investigate six widely used open-source LLMs and show the complementary potential of such wide-range downstream tasks by aggregating them via reward model ranking. We then collect a diverse training query set and distill rewards of model expertise as indirect supervision for training an LLM router and develop tag-based label enhancement to overcome the shortage of such silver labels from reward models further. With comprehensive experiments, we show ZOOTER can benefit from RM silver supervision to learn the latent expertise among LLMs and conduct efficient routing for the model ensemble. Our contributions are mainly three-fold:

- We revisit the complementary potential of open-source LLMs, which proves the effectiveness of LLM ensemble, and show rewards from off-the-shelf RMs can be silver supervision for model expertise.
- We propose ZOOTER, an efficient reward-guided routing method, distilling rewards from off-the-shelf reward models for probing model expertise. Then, we develop a tag-based label enhancement to mitigate noise from the uncertainty of reward models.
- We comprehensively evaluate ensemble methods, including reward model ranking and ZOOTER on four groups of benchmarks with 26 subsets on different tasks and domains. Our evaluation shows ZOOTER can effectively assemble LLMs and even outperforms reward model ranking methods with significantly less computation overhead.

2 Related Works

Instruction Tuning and Alignment. Instruction tuning (Longpre et al., 2023) helps LLMs to follow versatile instructions, which is widely adopted to align LLMs with human preference (Chiang et al., 2023; Xu et al., 2023; Bai et al., 2023). The quick emergence of aligned LLMs motivates us to develop routing techniques to maximize the strengths of ensemble LLMs. In this work, we focus on assembling aligned LLMs, such as Llama-2-Chat (Touvron et al., 2023b), WizardLM (Xu et al., 2023), Vicuna (Chiang et al., 2023), and so on. And we evaluate them on a wide range of alignment evaluation tasks.

Large Language Model Ensemble. The ensemble of LLMs is an emerging topic due to the explosion of open-source LLMs. LLM ensemble aims to merge off-the-shelf LLMs to perform better consistently across diverse downstream tasks. Few works explore the complementary potential assumption of LLMs and how to assemble LLMs with it. Jiang et al. (2023) presents an ensembling framework consisting of a pair ranker and a generation fuser. Chen et al. (2023) sequentially infers off-the-shelf LLMs and stops until the response meets a sufficient quality. Wang et al. (2023b) proposes a fusing-of-experts problem that fuses outputs of expert models with complementary knowledge of the data distribution and formulates it as supervised learning. Shnitzer et al. (2023) show the utility and limitations of learning model routers from various benchmark datasets. Although these works all focus on reward ranking or routing strategies to assemble LLMs, ZOOTER distinguishes from these concurrent works in two aspects. First, our concurrent works require output generations or the forward process to get prompt representations of all candidates, leading to significant computation overhead. ZOOTER infers model expertise by distilling rewards on a predefined training query set to avoid such inference overhead. Then, all these works are developed and evaluated on a set of benchmarks. At the same time, ZOOTER can be developed with only queries without golden responses, and ZOOTER aims for more diverse alignment tasks. Therefore, ZOOTER stands out for its efficiency in data and computation. We also evaluate ZOOTER on more diverse alignment tasks to comprehensively examine the complementary potential of LLMs.

Reward Model Guided Generation. Reward models in the context of large language models are commonly used to improve alignment performance by reinforcement learning (Schulman et al., 2017; Ouyang et al., 2022) or preference learning (Yuan et al., 2023b; Rafailov et al., 2023; Song et al., 2023). Reward models can also improve the performance during the generation phase. The math reasoning ability of language models can be improved by using reward models ranking multiple generated reasoning paths (Cobbe et al., 2021; Uesato et al., 2022; Lightman et al., 2023). Liu et al. (2023) uses reward models to formulate reward-guided decoding. Inspired by these successful applications of reward models in alignment, ZOOTER also takes advantage of off-the-shelf reward models to investigate the latent expertise of LLMs.

Mixture-of-Experts. Traditionally, Mixture-of-Experts (MoE) models encompass neural networks that include gating and expert modules (Xu et al., 1994). Yet, in large language models (LLMs), MoE LLMs are primarily developed to tackle computational scalability challenges by adopting parameter-efficient routing and expert structures (Rajbhandari et al., 2022; He et al., 2021a; Jiang et al., 2024). With this understanding, we assert that Zooter and MoE models differ markedly. Though we share the terminology "router" and "expert.", we tried to deal with significantly different problems on different motivations. Zooter and other LLM Ensemble methods, as well as recent developments (Yu et al., 2024; Wan et al., 2024), focus on integrating off-the-shelf LLMs. By off-the-shelf, we mean these are pre-trained (or aligned) models, which we do not further train during the ensemble process. This approach stems naturally from the rapid proliferation of LLMs, which we discuss at the outset of our paper. Conversely, MoE models, which combine routing and expert neural network modules, typically require the training of the experts and are driven by the pursuit of parameter efficiency. Furthermore, Zooter, along with most other LLM ensemble methods mentioned in the related works, is a sequence-level ensemble of frozen LLM models as a whole. In other words, we consider the aggregation of the sequence outputs of each candidate LLM. However, MoEs mainly conduct token-level routing and aggregating with expert modules.

3 Methods

We first revisit the complementary potential of LLMs (§3.1) and then introduce ZOOTER as an efficient LLM ensemble method (§3.2).

3.1 Complementary Potential of LLMs

In this section, we present the preliminaries about the assumption: *Off-the-shelf aligned LLMs have heterogeneous expertise across a wide range of domains and topics*. We also briefly introduce two LLM ensemble strategies, reward model ranking, and query routing.

Complementary Potential Assumption. Considering a set of LLMs denoted as $\mathcal{M} = \{m_i | i \in \mathcal{Z}^+\}$ and a set of downstream queries denoted as $\mathcal{Q} = \{q_i | i \in \mathcal{Z}^+\}$, we assume that for each LLM m_i in $\mathcal{M}$, there exists a non-empty query subset $\mathcal{Q}_{m_i}$ such that the LLM can achieve uniformly better performance than other LLMs in $\mathcal{M}$ for any query $q_j \in \mathcal{Q}_{m_i}$, which is $m_i = \mathrm{argmax}_{m \in \mathcal{M}} P\left(q_j, m(q_j)\right)$. P can be any preference or metric for performance assessment. Under this assumption, LLMs have their own expertise across different domains and tasks, further indicating the potential complementary among LLMs through model ensemble.

Reward Model Ranking. Integrating LLMs with such potential complementary can be achieved with the reward model. Reward model ranking (RMR) uses a reward function $\hat{P}$ estimating the oracle preference P to find the best LLM response for each query through ranking (Jiang et al., 2023). However, RMR, as a post-generation ranking strategy, requires inference of all candidate LLMs to score

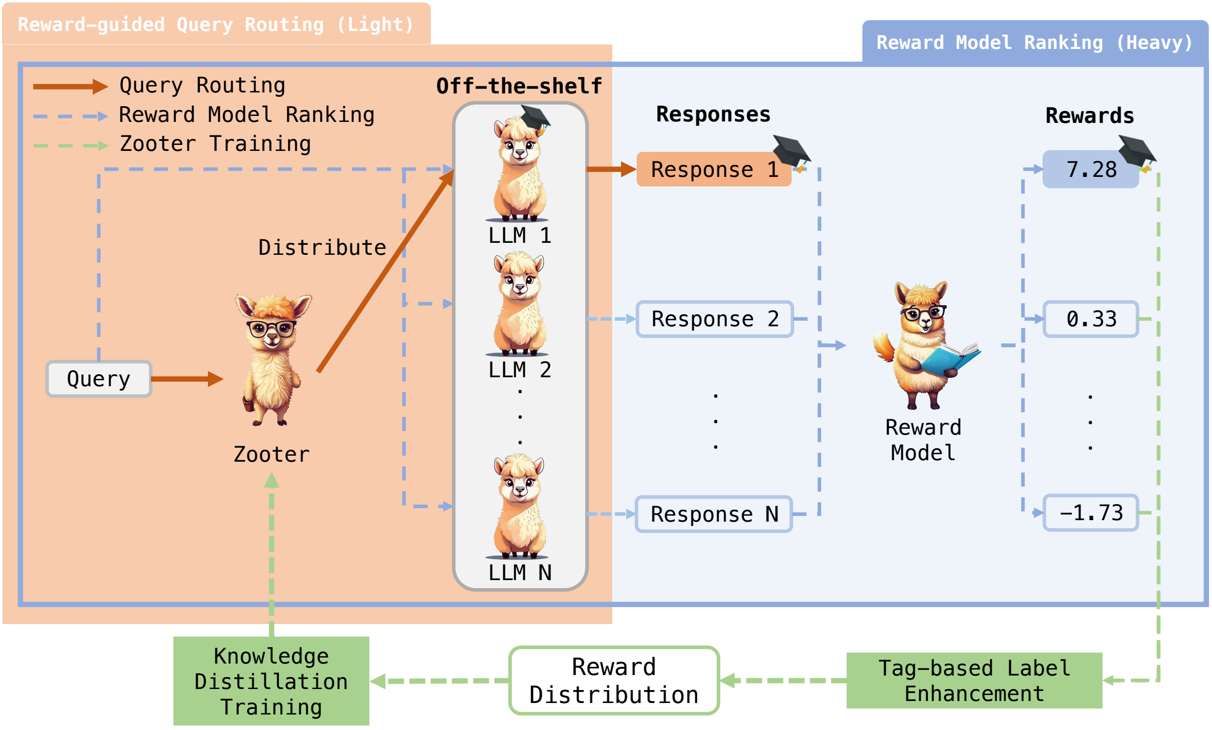

Figure 2: Overview of ZOOTER. ZOOTER aims to assemble a set of off-the-shelf LLMs by first conducting a reward model ranking on a diverse training set to obtain supervision of model expertise, highlighted in blue in the figure. Instruction tags are then used to mitigate the uncertainty in reward estimation. ZOOTER uses the normalized rewards as supervision to train a routing function by knowledge distillation. The training circle is marked in green, and the inference is marked in orange. ZOOTER is much lighter in computation as it routes the query to the corresponding expert LLM during inference time, while reward model ranking has to generate outputs for all candidates.

the responses. Therefore, RMR is hindered by inefficient large computation overhead.

Query Routing. Efficiency defect can be mitigated by pre-generation query routing. In general, query routing tries to find a routing function $\mathcal{Z}(q, m_i)$ with respect to $q_j \in \mathcal{Q}$ exists, so that $m_i = \text{argmax}_{m \in \mathcal{M}} \mathcal{Z}(q_j, m)$. The routing function distributes queries to the expert LLM without generating outputs in advance. If the complementary potential of LLMs holds, the routing function predicts the probability that a query q belongs to the expertise of an LLM Q_m. In this work, we bridge reward model ranking and query routing to achieve an efficient and effective LLM ensemble method.

3.2 Zooter

In this section, we propose ZOOTER, a reward-guided query routing method for efficiently assembling large language models. ZOOTER learns from the reward model ranking to interpret the latent expertise of each model. So, as shown in Fig. 2, ZOOTER first infers all candidate LLMs on a training set containing diverse queries to generate responses. Then, all responses will be rewarded by an off-the-shelf reward model providing scalar rewards, marked in blue dash lines in Fig. 2. The rewards are first enhanced by a tag-based prior for smoothing and denoising. The normalized reward distribution is then used as supervision in the knowledge distillation training of the routing function, shown in the green dash lines in Fig. 2. During inference, the routing function categorizes the input query to an LLM with the strongest expertise potential in this query, and the LLM will generate an expert response. By training such a routing function, ZOOTER achieves a much more efficient ensemble as it only needs to infer one expert LLM, plus a small computation overhead of the routing function. In this section, we introduce the two key components, reward distillation and tag-based label enhancement, along with the design motivations.

Reward Distillation. According to the idea of query routing in §3.1, we define our routing function $\mathcal{Z}_\theta(q)$ which represents how likely a query q falls into the expertise of each LLM by producing categorical distribution over $|\mathcal{M}|$ LLMs. We learn such a routing function by distilling the RMR results on a set of training queries $\mathcal{Q}_{\text{train}}$. Concretely, in RMR, we can acquire the reward model score for each LLM response $\hat{P}(q, m_i(q))$, where $q \in \mathcal{Q}_{\text{train}}$ and $m_i \in \mathcal{M}$. The score can be interpreted as the relative advantage of an LLM m_i over all other candidates on the query q. Higher advantages inherently present the expertise of the LLM. By normalizing the reward scores to categorical distributions, there present silver labels $\mathbf{s}(q)$ of estimating how likely the LLMs expertise the query q. We can then train the router function $\mathcal{Z}_\theta(q)$ by distillation using a Kullback-Leibler divergence (KLD) according to the solver labels. With a straightforward normalization approach of softmax, our loss function is

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{Q}_{\text{train}}|} \sum_{q \in \mathcal{Q}_{\text{train}}} \text{KLD}(\mathcal{Z}_\theta(q), \mathbf{s}(q)),$$

where

$$\mathbf{s}(q)_i = \frac{\exp(\hat{P}(q, m_i(q)))}{\sum_{m_i \in \mathcal{M}} \exp(\hat{P}(q, m_i(q)))}.$$

However, queries in the training set are expected to be as diverse as possible to maximize the generalization abilities of the routing function. The distillation process helps ZOOTER to learn the latent expertise of each model. So, we can mitigate the computation cost by only judging whether a query belongs to the expertise set with our routing function during inference.

Tag-based Label Enhancement. Although reward distillation provides a feasible way for routing functions to leverage silver supervision from reward model ranking, the language reward model provides rewards with uncertainty, introducing certain noises (Gleave and Irving, 2022). Through our preliminary experiment, the naive softmax-normalized silver labels are prone to noise from the reward model score, because the off-the-shelf reward models are not necessarily accurate for preference assessment as shown in Fig. 3.. The analysis of this uncertainty can be found in §4.3. Therefore, we leverage instruction tagging (Lu et al., 2023) to enhance rewards on the training queries further. The tag-based label enhancement we proposed is similar to the widely used label smoothing techniques and proven effective in knowledge distillation (Yuan et al., 2020). Specifically, we first tag each query $\hat{q}_i \in \hat{Q}$ with a local tagger $\mathcal{T}(\cdot)$ to obtain a set of tags $\mathcal{T}(q_i)$. Then, we aggregate all rewards on queries with the same tags for the tag-wise rewards as follows:

$$Q_t = \{q_i | t \in \mathcal{T}(q_i), i = 1, \ldots, |\hat{Q}|\}$$
$$\mathbf{s}_t(q) = \frac{1}{|Q_t|} \sum_{i \in Q_t} \mathbf{s}(q_i)$$

Then, we enhance rewards for each query with tag-wise rewards by a linear combination:

$$\mathbf{s}(q)^* = \beta \mathbf{s}(q) + (1 - \beta) \frac{1}{|\mathcal{T}(q)|} \sum_{t \in \mathcal{T}(q)} \mathbf{s}_t(q),$$

where β is a hyper-parameter for the trade-off between coarse-grained tag-wise rewards and fine-grained sample-level rewards. Then, we replace original rewards in the KL divergence loss training with tag-based enhanced rewards $\mathbf{s}(q)^*$ during routing function training.

4 Experiments

In this section, we report experimental setup (§4.1), main results (§4.2), and analysis about ZOOTER (§4.3).

4.1 Experimental Setup

We verify the effectiveness of ZOOTER by routing queries in multiple benchmarks to a group of candidate LLMs. We compare the performance of ZOOTER with the single candidate model and various reward model ranking baselines.

Candidate LLMs. We select six LLAMA-based LLMs of the same 13B size as the candidate LLMs for query routing. (a) **WizardLM** (Xu et al., 2023) is aligned with queries and responses augmented by EVOLINSTRUCT, (b) **WizardCoder** (Luo et al., 2023b) is a coding expert LLM using the same techniques as WizardLM, (c) **WizardMath** (Luo et al., 2023a) is a math expert LLM aligned with query augmentation, ChatGPT rewards and PPO optimization, (d) **Vicuna** (Chiang et al., 2023) is aligned on tremendous conversations between users and proprietary chatbots, (e) **OpenChat** (Wang et al., 2023a) is aligned with a selected set of ShareGPT with additional training strategies, (f) **Llama-2-Chat** (Touvron et al., 2023b) is first aligned by supervised fine-tuning and then multi-turn rejection sampling. Both baselines and ZOOTER are experimented with and evaluated based on these six candidates.

Model	#Param		AlpacaEval (5)		FLASK (10)		MT-Bench (8)		Benchmarks (3)		All (26)	
	Ranker	Infer	Avg.	MTR	Avg.	MTR	Avg.	MTR	Avg.	MTR	MTR	% Uplift
Routing Candidates												
WizardCoder	−−	13B	0.42	5.6	3.12	5.2	4.44	5.38	30.9	4.33	5.3	0.06
WizardLM	−−	13B	0.89	2.0	3.89	1.8	7.15	2.0	44.2	2.0	1.83	0.25
WizardMath	−−	13B	0.47	5.0	3.28	5.0	5.73	4.38	34.8	4.0	4.6	0.03
Llama-2-chat	−−	13B	0.91	1.6	3.88	1.5	6.72	2.88	32.3	3.67	2.23	0.31
OpenChat	−−	13B	0.89	2.2	3.79	3.1	7.12	2.0	31.2	3.33	2.67	0.19
Vicuna	−−	13B	0.8	3.8	3.7	3.5	6.58	3.25	33.6	2.67	3.4	0.06
BMA	−−	13B	0.91	1.6	3.88	1.5	6.72	2.88	32.3	3.67	2.23	0.31
Zooter												
Ours	86M	13B	0.93	1.17	3.89	1.82	7.11	2.33	34.2	3.0	1.94	0.44
Reward Model Ranking (RMR)												
w/ OAssistRM	300M	6×13B	0.79	4.0	3.75	3.73	6.59	3.22	35.1	3.25	3.42	0.19
w/ LLM-Blender	300M	6×13B	0.83	3.67	3.77	3.36	6.21	4.0	36.4	2.75	3.39	0.17
w/ Auto-J	13B	6×13B	0.89	2.67	3.92	1.64	7.03	2.22	32.2	3.5	2.25	0.42
w/ UltraRM	13B	6×13B	0.92	1.17	4.06	1.0	7.18	1.89	40.1	3.25	1.53	0.72
w/ QwenRM	7B	6×13B	0.92	1.33	4.04	1.0	7.26	2.11	38.6	3.0	1.58	0.67
w/ Oracle	−−	6×13B	0.98	1.0	4.56	1.0	8.25	1.0	75.3	1.0	1.0	1.0
Proprietary Models												
GPT-3.5-turbo	−−	−−	0.89	2.67	4.06	1.91	7.94	1.78	73.0	1.0	1.78	0.61
GPT-4	−−	−−	0.94	1.0	4.37	1.0	8.99	1.0	88.3	1.0	1.0	1.0

Table 1: Main results of both Zooter and reward model ranking. We report performance across four groups of benchmarks and report the number of subsets beside the name of benchmarks. We also report the parameters of ranker and total inference models for both candidates and ensemble methods. MTR denotes the mean task rate, and %Uplift denotes the rate of uplift. The average scores and uplift rate are as higher as better while MTR is as lower as better. We mark better scores in darker blue for better visualization and easier interpretation.

Training Datasets. We create a diverse mix instruction dataset from the open-source data to maximize the generalization abilities of Zooter. We first collect and tag open-source data from 13 datasets with a local tagger developed by Lu et al. (2023). Each query is assigned a group of tags that describe its intentions and semantics. For trustworthy evaluation results, we decontaminate all samples containing queries with a 6-gram overlap with any samples in our benchmarks described below to avoid data leakage. Then, we randomly select ten samples for each unique tag to form a diverse mix instruction dataset DivInstruct with 47,986 instructions and samples across 6,270 different tags. Detailed statistics of DivInstruct is in Appx. §A.

Evaluations. We involve four sets of benchmarks to evaluate Zooter on various downstream tasks comprehensively. We first include three widely-used alignment benchmarks with GPT-4 judge:

- **AlpacaEval** (Li et al., 2023b) consists of 5 subsets from the koala, vicuna, and others evaluation sets. It contains 805 samples in total. AlpacaEval reports the win rate between evaluated models and *text-davinci-003*, measured by GPT-4.
- **FLASK** (Ye et al., 2023) is a fine-grained evaluation for alignment. We evaluate methods on all ten domains in FLASK and report the average score across all domains as a final score. Each score is provided by GPT-4, ranging from 0 to 5.
- **MT-Bench** (Chiang et al., 2023) is a multi-turn evaluation across eight aspects, including mathematics and coding. We only train and route with the first-turn query but evaluate in the multi-turn manner as the original recipe. We report the average score over turns, aspects, and samples ranging from 0 to 10.
- **Benchmarks** includes a group of benchmarks consisting of MMLU (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), and HumanEval (Chen et al., 2021). We all conduct zero-shot inference on each benchmark. We report accuracy on MMLU and GSM8K, and Pass@1 on the HumanEval.

As reported by Wang et al. (2023c), GPT-4 judgments may have bias and significant disagreement with humans. Therefore, we also include a group of benchmarks in our evaluation to provide a multi-faceted conclusion.

Ranking Metrics. Comparing ensemble models on various benchmarks is challenging as the scale of scores is different on each benchmark. To combat this issue, we do not only report the scores on each benchmark but also the mean task rank (MTR). All benchmarks we evaluate have multiple subsets. We define MTR as the rank of the evaluated model among all baselines' averages on all subsets. MTR is only about the rank among baselines so it can be easily adopted across benchmarks with different score scales. Similarly, we also propose an uplift rate, denoting the rate of subsets that the evaluated model achieves the best performance of benchmarks. We report these two metrics on 26 evaluation subsets in all benchmarks. Lower MTR and higher uplift rates show the evaluated model has consistently higher performance among versatile downstream tasks.

Baselines. A natural baseline for ZOOTER is each candidate model itself. Furthermore, we report the best model on average (BMA), which denotes the best candidate model across all benchmarks. We also compare ZOOTER with reward model ranking (RMR) based on different reward models. We set up RMR baselines with the latest rewards models: (1) OASSISTRM is a DEBERTA-based reward model trained on OAssist preference annotations, (2) AUTO-J (Li et al., 2023a) is a text-based large language model for critique and judgment, (3) ULTRARM (Cui et al., 2023) is a scalar reward model trained on the UltraFeedback dataset, (4) QWENRM (Bai et al., 2023) is a pretrained and fine-tuned reward model developed for multilingual preference learning and an Oracle ranking for reference. We also consider the pair ranking in LLM-Blender (Jiang et al., 2023) as one of the RMR methods. Besides, we also report the performance of proprietary models across our benchmark collections for reference, including GPT-3.5-turbo and GPT-4.

Configurations. We train our routing function from *mdeberta-v3-base* (He et al., 2021b). We use QwenRM to generate rewards on training queries as supervision for our routing function, as it achieves the best performance in reward model ranking with considerably smaller model parameters described in §4.2. And we run all training and inference on 8 A100 GPUs. We infer and evaluate all benchmarks with corresponding configurations and GPT-4 settings. We use greedy decoding for MMLU, GSM8K, and HumanEval. More details are described in Appx. §B.

4.2 Results

We present the main results in Tab. 1. We report the performance of six routing candidates across our benchmarks, and the best model on average (BMA) is LLAMA-2-CHAT. And we report ZOOTER with $\beta = 0.3$ in tag-based label enhancement. We further analyze the results in the following two aspects:

Complementary Potential. We evaluate the ensemble with reward model ranking (RMR) on five different off-the-shelf reward models. RMR with UltraRM achieves the best performance in MTR and uplift rate on the aggregation of all benchmarks, which ranks at 1.53 and achieves the best model across 72% subtasks. RMR with QwenRM achieves the second best and performs similarly to UltraRM with smaller parameter sizes, followed by RMR with Auto-J, LLM-Blender, and OAssistRM. RMR with QwenRM, UltraRM, and Auto-J outperform that of BMA, showing the effectiveness of RMR. Furthermore, we also calculate the score of RMR with an Oracle ranker, which consistently outperforms all candidates and even outperforms GPT-4 on AlpacaEval and FLASK. Such results provide solid evidence for the complementary potential of off-the-shelf LLMs and also support the key motivation behind ZOOTER, i.e., using rewards from off-the-shelf reward models as silver supervision for the routing function training. However, we notice RMR fails on benchmarks, such as MMLU, GSM8K, and HumanEval, showing that precisely judging knowledge, mathematics, and coding problems are still challenging for existing RMs.

Zooter Performance. We then compare the performance of ZOOTER with that of BMA and RMR. ZOOTER outperforms BMA on AlpacaEval, MT-Bench, and Benchmarks, and achieves similar performance on FLASK. The most significant improvement is witnessed on MT-Bench, where the performance of ZOOTER is higher than that of BMA by 0.39. In general, ZOOTER achieves top-1 on 44% subtasks while BMA is only on 31%. With the evidence above, ZOOTER successfully utilizes the complementary potential between LLMs to achieve the best performance more consistently over our benchmarks, with computation overhead from only 86M ranker. At the same time, ZOOTER

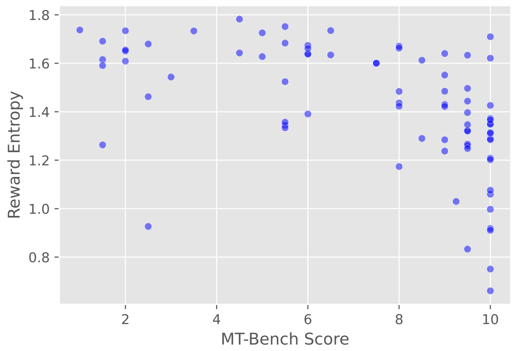

Figure 3: Analysis between reward entropy and scores of reward preference ranking on MT-bench.

β	**AlpacaEval**	**FLASK**	**MT-Bench**	**Benchmarks**	**All**
		Hard Label Training			
0	1.4	2.3	3.12	4.00	2.31
		Reward Distillation			
0	1.4	2.2	2.25	3.67	2.06
0.1	1.2	2.1	2.38	3.67	2.00
0.3	1.2	1.9	2.50	3.67	1.97
0.5	1.2	2.2	3.12	3.67	2.23
0.7	1.2	2.2	3.38	4.00	2.31
0.9	1.2	2.3	3.12	4.00	2.31
1.0	1.2	2.3	3.25	4.00	2.34

Table 2: Mean task rank (MTR) of hard label training and reward distillation with different β in tag-based label enhancement across all benchmarks. The best value of β is marked in blue.

outperforms RMR with OAssistRM, LLM-Blender, and Auto-J, by significantly less computation overhead. However, though ZOOTER outperforms RMR with QwenRM on AlpacaEval, there are still obvious gaps between ZOOTER and RMR with QwenRM in general.

4.3 Analysis

We provide further analysis on how RM uncertainty may influence the training of ZOOTER.

RM Uncertainty. As presented in the previous research, RM may have uncertainty on its scalar rewards, which may introduce noise in the routing training since we use RM scores as silver supervision. In this subsection, we first present the existence of this uncertainty to explain the motivation behind tag-based label enhancement, the method we propose to mitigate such uncertainty in the routing function training. We calculate the entropy of rewards from QwenRM among all candidate LLMs for each query in MT-Bench and draw it with the MT-Bench scores of each sample by reward preference ranking with QwenRM. As shown in Fig. 3, samples with lower reward entropy tend to have high MT-bench scores. We interpret this observation as higher reward entropy reveals more uncertainty in the reward. Therefore, we propose tag-based label enhancement to mitigate the noise when using rewards as silver labels.

Label Enhancement. The tag-based label enhancement proposed in §3.2 contains a hyper-parameter β, representing the trade-off between fine-grained sample-level rewards and coarse-grained tag-level rewards. We conduct experiments to tune this hyperparameter and analyze how rewards in different granularities may influence the training of our routing function. As shown in Tab. 2, ZOOTER achieves the best performance when β equals 0.3, proving a combination of sample-level and tag-level rewards will benefit the reward distillation. The ablation also shows the necessity of tag-based label enhancement. Furthermore, distilling tag-level rewards ($\beta = 0$) shows significantly better performance than distilling sample-level rewards ($\beta = 1$), supporting the analysis that noises from the uncertainty of RMs in sample-level rewards damage reward distillation.

Hard Label Training. We further ablate the reward distillation by proposing a straightforward baseline named hard label training. In this setting, we use the model with the highest reward as the categorized label to formulate a classification task. And then, we train our routers with the cross entropy loss on these hard labels. As presented in Tab. 2, reward distillation under $beta = 0$ achieves 2.06 MTR, which is significantly lower than the MTR of hard label training (2.31). This result provides further evidence of the necessitate of reward distillation.

5 Conclusion

In this work, we revisit the complementary potential of open-source LLMs and reward model ranking of multiple off-the-shelf reward models, providing evidence of LLM ensemble's effectiveness. We propose ZOOTER, an efficient reward-guided routing method for ensemble off-the-shelf LLMs. Comprehensive evaluation shows ZOOTER can outperform the best single model on average and even ensemble models by reward model ranking with significantly fewer computation overhead. Valuable future works include diving deep into interpreting latent expertise in each LLM.

Limitations

ZOOTER requires an off-the-shelf reward model (RM) and its performance is highly related to the abilities of the RM. Though there are open-source RMs to re-implement ZOOTER, this requirement still restricts the further application of this method.

Ethics statement

ZOOTER assembles off-the-shelf open-source large language models, which may potentially generate harmful and toxic contents. We use GPT-4 to provide evaluations on three benchmarks, AplacaEval, FLASK, and MT-Bench.

References

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback.

Adam Gleave and Geoffrey Irving. 2022. Uncertainty estimation for language reward models. *arXiv preprint arXiv:2203.07472*.

Jiaao He, Jiezhong Qiu, Aohan Zeng, Zhilin Yang, Jidong Zhai, and Jie Tang. 2021a. Fastmoe: A fast mixture-of-expert training system.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021b. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. *arXiv preprint arXiv:2306.02561*.

Junlong Li, Shichao Sun, Weizhe Yuan, Run-Ze Fan, Hai Zhao, and Pengfei Liu. 2023a. Generative judge for evaluating alignment. *arXiv preprint arXiv:2310.05470*.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe.

2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. 2023. Don't throw away your value model! making ppo even better via value-guided monte-carlo tree search decoding.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, and Chang Zhou. 2023. # instag: Instruction tagging for diversity and complexity analysis. *arXiv preprint arXiv:2308.07074*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023a. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023b. Wizardcoder: Empowering code large language models with evol-instruct.

Long Ouyang, Jeff 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 Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model.

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.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. 2023. Large language model routing with benchmark datasets.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. 2023. Preference ranking optimization for human alignment.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. 2024. Knowledge fusion of large language models.

Guan Wang, Sijie Cheng, Qiying Yu, and Changling Liu. 2023a. OpenChat: Advancing Open-source Language Models with Imperfect Data.

Hongyi Wang, Felipe Maia Polo, Yuekai Sun, Souvik Kundu, Eric Xing, and Mikhail Yurochkin. 2023b. Fusing models with complementary expertise.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023c. How far can camels go? exploring the state of instruction tuning on open resources.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions.

Lei Xu, Michael I. Jordan, and Geoffrey E. Hinton. 1994. An alternative model for mixtures of experts. In *Neural Information Processing Systems*.

Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. 2023. Flask: Fine-grained language model evaluation based on alignment skill sets.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch.

Li Yuan, Francis EH Tay, Guilin Li, Tao Wang, and Jiashi Feng. 2020. Revisiting knowledge distillation via label smoothing regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3903–3911.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023a. Scaling relationship on learning mathematical reasoning with large language models.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023b. Rrhf: Rank responses to align language models with human feedback without tears.

Dataset	**Amount**
ultrachat	18,588
sharedgpt	10432
wizardlm(sharedgpt)	5325
wizardlm(alpaca)	5145
alpaca	2186
repair	1034
openchat	1033
flan	862
math	849
unnatural	582
dmcc	573
dolly	560
oasst	183
lima	70
mbpp	43

Table 3: Composition of DIVINSTRUCT

A Datasets

DIVINSTRUCT is a diverse mix instruction set from multiple open-source datasets with careful decontamination on all benchmarks evaluated in this work. The detailed composition of DIVINSTRUCT is report in Tab. 3.

B Hyper-parameters

The QwenRM we used for supervision contains about 7 billion parameters. We infer QwenRM on 8 A100 GPUs for 24 hours to generate annotations for all samples and all candidate models. And the routing function *mdeberta-v3-base* has about 100 million parameters. We train it with 8 A100 GPUs with 5 epochs for 2 hours. We search the training hyper-parameters by grid search, and the best learning rate is $1e-5$ and the weight decay is $5e-7$. Responses for all candidate models are sampled with temperature 0.7 and a maximum token number 2048.