Not Everything is All You Need: Toward Low-Redundant Optimization for Large Language Model Alignment

Anonymous EMNLP submission

Abstract

Large language models (LLMs) are still struggling in aligning with human preference in complex tasks and scenarios. They are prone to overfit into the unexpected patterns or superficial styles in the training data. We conduct an empirical study that only selects the top-10% most updated parameters in LLMs for alignment training, and see improvements in the convergence process and final performance. It indicates the existence of redundant neurons in LLMs for alignment training. To reduce its influence, we propose a low-redundant alignment method named **ALLO**, focusing on optimizing the most related neurons with the most useful supervised signals. Concretely, we first identify the neurons that are related to the human preference data by a gradient-based strategy, then identify the alignment-related key tokens by reward models for computing loss. Besides, we also decompose the alignment process into the forgetting and learning stages, where we first forget the tokens with unaligned knowledge and then learn aligned knowledge, by updating different ratios of neurons, respectively. Experimental results on 10 datasets have shown the effectiveness of ALLO. Our code and data will be publicly released.

1 Introduction

Alignment with human preferences has become a desired property of LLMs (Askell et al., 2021; Ouyang et al., 2022), *e.g.,* helpfulness, honesty, and harmlessness, and reinforcement learning from human feedback (RLHF) (Christiano et al., 2017; Zheng et al., 2023) is a crucial technique for achieving it. Typically, RLHF aims to fine-tune LLMs on human preference data, to maximize and minimize the likelihood of generating the positive and negative responses, respectively. After RLHF training on corresponding datasets, LLMs can better follow user instructions (Ouyang et al., 2022), solve complex problems (Wang et al., 2023), and generate unbiased responses (Bai et al., 2022a).

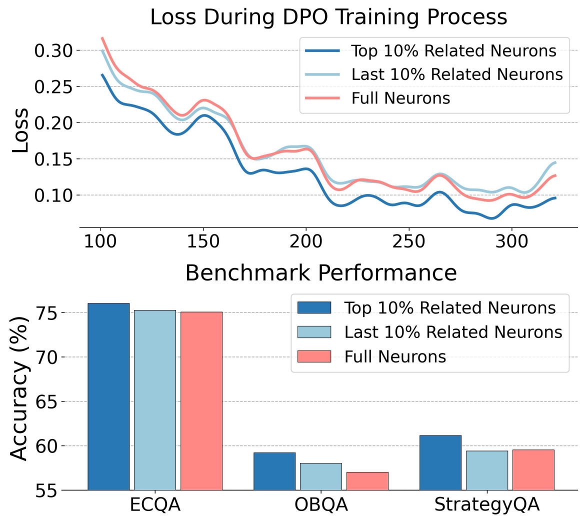

Figure 1: The training loss curve and benchmark performance of training different neurons in LLM on question-answering data using DPO (Rafailov et al., 2023).

However, it is hard to train a well-aligned LLM for complex tasks and scenarios (Feng et al., 2024; Gekhman et al., 2024). The key issue is that LLMs might overfit into the unexpected patterns or superficial styles in the human preference data (Du et al., 2024). It is the side effect of their powerful learning capability derived from the large-scale trainable parameters (Song et al., 2024; Meng et al., 2024a). Recently, a surge of work (Frankle and Carbin, 2019; Wang et al., 2024b) has found that each neuron is relevant with special knowledge, and the neurons in LLMs are generally sparsely activated. Inspired by it, we consider if the full-parameter trained LLMs might lead to ***redundant updates*** on alignment-irrelevant neurons. Thus, we conduct the empirical experiment using DPO algorithm (Rafailov et al., 2023), where we only update the top/last-10% neurons according to their accumulated gradient values. As shown in Figure 1, with the top-10% trainable neurons, LLMs can converge faster and achieve better performance than optimizing all the neurons. It indicates the existence of redundant updates in DPO training, affecting the convergence and final performance.

To reduce the influence of redundant updates, we focus on optimizing the most related neurons with the most useful supervised signals. Concretely, we first identify the neurons that are related to the human preference data, based on the accumulation values of gradients. Second, we identify the key tokens about human preference, and only compute loss on them for optimizing the alignment-related neurons. In this way, we perform a ***two-fold low-redundant optimization*** for aligning LLMs with humans, to reduce the redundancy of learning irrelevant tokens and training irrelevant neurons. Whereas, since the alignment training focuses on both removing the unaligned knowledge and learning the aligned one, the involved neurons and tokens are not always consistent for the two objectives. Therefore, we decompose the alignment process into the ***forgetting and learning stages***, and adapt the low-redundant optimization strategy on them. For the forgetting stage, relatively fewer neurons are trained by unlearning algorithm (Zhang et al., 2024) to forget the unaligned knowledge, and we leverage a token-level reward model (Chen et al., 2024b) to identify the unaligned tokens required to be focused. For the learning stage, we train more neurons using the DPO algorithm, and also utilize its reward score to select the key tokens.

In this work, we proposed an **AL**ignment method with **L**ow-Redundant **O**ptimization (ALLO) to fine-tune LLMs. In ALLO, we first identify the most important neurons based on the accumulation of gradients from a reference model. Then, we design the forgetting and learning stages, where we adopt the token-level reward model and the DPO reward function to select the key tokens, for computing loss to update different ratio of the important neurons (*e.g.,* top-5% and 10%), respectively. In this way, we only perform optimization on sparse tokens and neurons, greatly reducing the redundancy during LLM alignment training. To comprehensively assess the effectiveness of ALLO, we conduct extensive experiments on three downstream scenarios, *i.e.,* question answering, mathematical reasoning, and instruction following, totally 10 datasets. Experiment results have shown that ALLO mostly outperforms competitive human alignment methods (*e.g.,* SFT (Ouyang et al., 2022), DPO (Rafailov et al., 2023), PPO (Schulman et al., 2017)).

2 Related Work

Large Language Models. LLMs have shown remarkable performance on various tasks (qwe, 2024; Meta, 2024; Javaheripi et al., 2023). Generally, the training process of LLMs includes three stages, *i.e.,* pre-training, supervised fine-tuning (SFT), and alignment (Ouyang et al., 2022; Touvron et al., 2023). In the training process, previous work has selected valuable data to train the LLMs via leveraging gradient (Xia et al., 2024) or perplexity (Lin et al., 2024; Xie et al., 2023), Besides, synthetic training data from powerful LLMs (*e.g.,* GPT-4, Claude 3) has been widely utilized for improving the weak LLMs (Xu et al., 2023; Ben Allal et al., 2024; Liu et al., 2024), especially for specific scenarios (*e.g.,* mathematical tasks or code synthesis tasks) (Yue et al., 2023; Zhou et al., 2024; Wei et al., 2023). However, given the large expenses of the LLM training, existing work (Hu et al., 2022; Li and Liang, 2021; Dettmers et al., 2023) has revealed that training only a small number of the parameters can achieve comparable performance with whole-parameters training. In this work, we focus on the alignment stage and leverage the low-redundant optimization to improve the existing LLMs.

LLMs Alignment. RLHF is a critical algorithm of LLM alignment (Christiano et al., 2017), usually leveraged to reduce hallucination (Chaudhari et al., 2024) or further enhance the capacities of LLMs (Chen et al., 2024b; Wang et al., 2023; Luo et al., 2023). Typically, a reward model will be trained on the preference data and leveraged to guide the reinforcement learning (RL) procedure (Ouyang et al., 2022; Touvron et al., 2023; Zheng et al., 2023). Proximal policy optimization (PPO) has been widely adopted in RLHF (Mnih et al., 2016; Zheng et al., 2023). Given the efficiency and expenses of the annotating process by human labeler, previous work has utilized the feedback from LLMs to instruct the RL process, named RLAIF (Bai et al., 2022b; Yuan et al., 2024). Furthermore, to prevent the instability of RL, a series of work (Park et al., 2024; Hong et al., 2024; Meng et al., 2024b) utilized a similar objective function with SFT to model human preference. Direct preference optimization (DPO) (Rafailov et al., 2023) is representative work of non-RL alignment. In this work, we consider about how to unleash the potential of the non-RL method.

Unlearning of LLMs. Machine unlearning (Cao and Yang, 2015; Bourtoule et al., 2019; Wang et al., 2024a; Chen et al., 2024a) is an important technique for artificial intelligence systems to remove the knowledge about the restricted data (*e.g.,* unauthorized books), while keeping other knowledge and abilities of the systems. To perform unlearning of LLMs, research has proposed several methods (*e.g.,* Gradient Ascent (Yao et al., 2023; Maini et al., 2024) and NPO (Zhang et al., 2024)), directly training LLMs on the invalid dataset to make LLMs forget relative knowledge. Following the unlearning mechanism, in this work, we utilize an unlearning algorithm to correct the unaligned knowledge stored in the neurons of LLMs.

3 Preliminary

LLMs alignment refers to aligning the behaviors of LLMs to human preference, *e.g.,* helpfulness, honesty, and harmlessness (Askell et al., 2021). Existing work typically utilizes RLHF methods (Christiano et al., 2017) to fine-tune LLMs using human preference data, for improving alignment. Formally, the human preference data is composed by input prompts, positive responses, and negative responses, denoted as $\mathcal{D} = \{\langle x_i, y_i^+, y_i^- \rangle\}_{i=1}^n$. The input prompt or response consists of a series of natural language tokens $\{t_1, t_2, \ldots, t_l\}$. Given the input prompt x, we aim to train LLMs that tend to generate the well-aligned positive response y^+, while avoiding generating the unaligned negative one y^-. In this work, we focus on devising an effective training algorithm to improve the alignment of LLMs, which can be utilized to satisfy the diverse requirements in real world (*e.g.,* instruction following and question answering).

According to our empirical study in Figure 1, updating only top-10% trainable neurons would achieve better performance than full-parameter tuning for alignment training. It indicates that there are redundant updates in the training process of LLMs, which may affect the alignment performance. To address it, in this work, we aim to perform parameter-efficient fine-tuning for reducing the redundant updates on unrelated neurons, to improve the alignment of LLMs. Given the training data, we first identify the highly-relevant neurons $\mathcal{N} = \{\theta_{i_1}, \ldots, \theta_{i_k}\}$ in the parameter matrices of LLMs, and perform low-redundant optimization on the LLM as:

$$\theta_i^{t+1} = \begin{cases} \text{Optimizer}(\theta_j^t, \nabla\theta_j^t), & \theta_j \in \mathcal{N} \\ \theta_j^t, & \theta_j \notin \mathcal{N} \end{cases}, \qquad (1)$$

where θ_j^t means the value of j-th neuron at the t-th step of training process, $\nabla\theta_j$ is the calculated gradient of j-th neuron for update.

4 Approach

In this section, we introduce our proposed method ALLO, a low-redundant alignment method for fine-tuning LLMs. In ALLO, we compute loss on selected key tokens, and perform sparse neuron optimization. Concretely, we first train a reference model to locate the important neurons through gradient. Then, we identify the key tokens related to unaligned knowledge, and utilize the unlearning algorithm to update few neurons for forgetting them. Next, we leverage DPO algorithm to improve the alignment of the LLM, where the DPO reward is used for selecting the key tokens. The framework of ALLO is presented in Figure 2.

4.1 Locating Key Neurons

We compute the importance of all the neurons for the human preference data to locate the related key neurons. We first train a reference model on the given data using DPO algorithm, and then design an efficient approximate estimation of the neuron importance based on its updated weights.

Training Reference Model. We train the reference model on the human preference data, to obtain the updated values of all neurons for importance estimation. Thus, we select the same LLM as the backbone, and perform full-parameter fine-tuning using DPO algorithm on the entire dataset for one epoch. The training objective is:

$$\mathcal{L}(d_i) = -\log\sigma\left(\beta\log\frac{P(y_i^+|x_i)}{P_{\text{ref}}(y_i^+|x_i)} - \beta\log\frac{P(y_i^-|x_i)}{P_{\text{ref}}(y_i^-|x_i)}\right), \qquad (2)$$

where β is a hyper-parameter, and $d_i = \langle x_i, y_i^+, y_i^- \rangle$ is a training instance. For the scenarios that only one human feedback is provided, we regard it as the positive one, and leverage the response generated from LLM as the negative one.

Neurons Importance Estimation. We aim to estimate the importance of each neuron for the given human preference dataset $\mathcal{D}$. As LLMs are generally trained by gradient descent algorithm, the

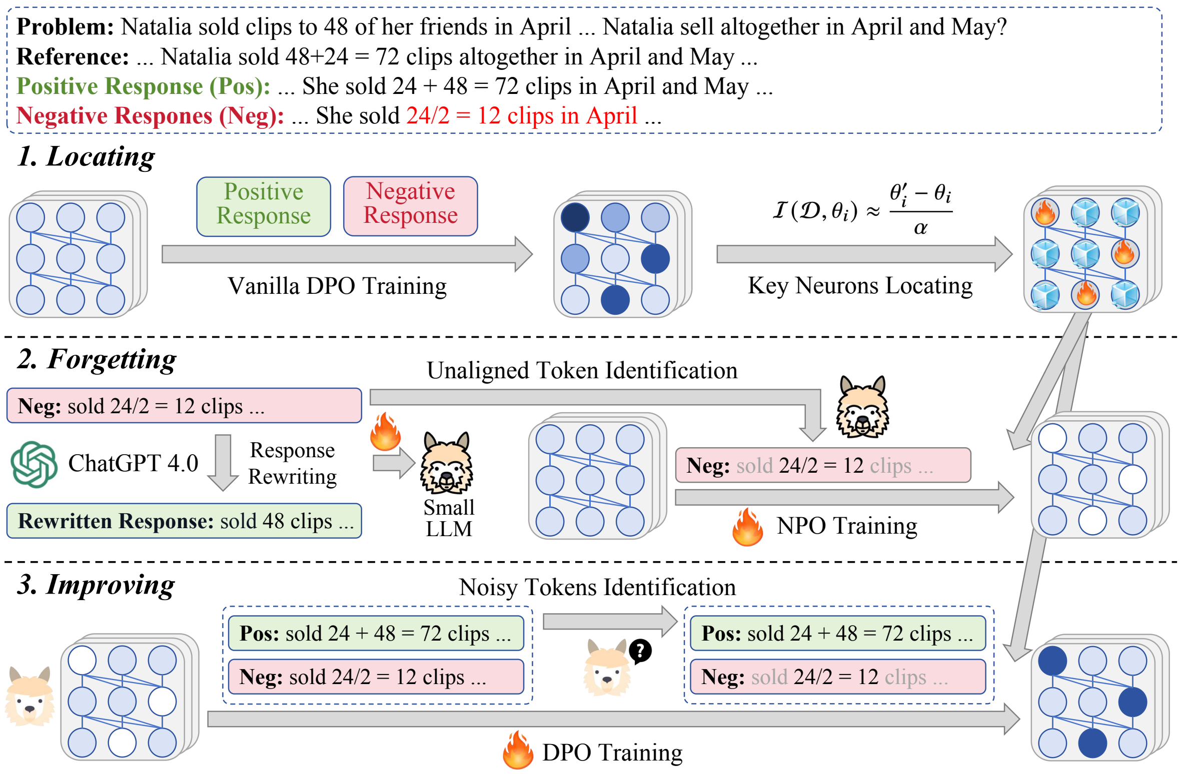

Figure 2: The framework of our proposed alignment method ALLO. We first locate the key neurons in LLMs by computing the weight changes of the reference model.Then, based on the selected key neurons, we perform a fine-grained unlearning using NPO to help LLMs forget unaligned knowledge, and fine-grained learning using DPO to further align LLMs to human preference.

gradient value of a training instance d_j on the neuron θ_i can reflect its influence on the neuron (Pruthi et al., 2020; Xia et al., 2024), denoted as:

$$\text{Influence}(d_j, \theta_i) \propto \nabla_{\theta_i} \mathcal{L}(d_j) \tag{3}$$

For human alignment, we use the DPO training loss in Eq. 2 for influence estimation. In this way, we can accumulate the gradients for all the instances from the human preference dataset, to estimate the influence of the dataset on the neuron. Actually, the influence value also reflects the importance of the neuron for learning the dataset, as a large accumulated gradient value can denote more focus on training the neuron (Pruthi et al., 2020). As we adopt the gradient descent algorithm, the gradients for all the instances have been computed and subtracted in the one-epoch training process. Thus, the difference between the neuron in the reference model θ_i' and original model θ_i can be regarded as the approximate value of the estimated importance score:

$$\mathcal{I}(\mathcal{D}, \theta_i) = \sum_{j=1}^{|\mathcal{D}|} \nabla_{\theta_i} \mathcal{L}(d_j) \approx \frac{\theta_i' - \theta_i}{\alpha}, \tag{4}$$

where α is the learning rate during DPO training. Based on the estimated importance score, we can rank all the neurons and select the most important ones for training.

4.2 Unaligned Knowledge Forgetting

For the forgetting stage, we utilize a token-level reward model that guides LLMs to focus on the tokens related to unaligned knowledge, and adopt a machine unlearning algorithm, *i.e.,* NPO (Zhang et al., 2024) that learns to forget them.

Unalignment-Related Tokens Identification. We train a token-level reward model to score tokens in the negative responses, according to their effect on unalignment. Following existing work (Chen et al., 2024b), we distill the capability of a strong LLM (*i.e.,* GPT-4 (OpenAI, 2023)) to revise the unaligned response (to a well-aligned one) with minimum editing constraint, into a small LLM. Then, we can utilize its output revision probability for each token, to compute the reward score as:

$$r_{i,j} = \begin{cases} 1, & P_{re}(y_{i,j}|p_i, x_i, y_i^+, y_{i,<j}^-) < u \\ 0, & \text{others} \end{cases}, \tag{5}$$

where p_i is the prompt to guide the reward model, $y_{i,j}$ is the j-th token in the negative response y_i^-, u is a hyper-parameter to control the threshold. In this way, we can select the key tokens about the unalignment according to the 0-1 reward score.

Fine-grained Unlearning with NPO. Based on the selected unalignment-related key tokens, we perform unlearning to remove the unaligned knowledge in the LLM and unleash the potential of learning aligned knowledge. Concretely, we utilize the NPO method, which is the revision based on DPO and only focuses on minimizing the likelihood of generating negative responses. The objective function of NPO is as follows,

$$\mathcal{L}_{NPO}(\theta) = \log \sigma \left(-\beta \log \frac{P(y_i^-|x_i)}{P_{\text{ref}}(y_i^-|x_i)} \right). \quad (6)$$

Whereas, the NPO loss would also punish the tokens that are irrelevant to the unalignment but exist in the negative response. To address it, we constrain that only the key tokens are involved into loss computation, to avoid unlearning the irrelevant tokens. Formally, we decompose the objective into the token level, and add the 0-1 reward score as the token weights. Thus, the objective function can be revised as follows, and we only optimize the top-k_1% most important neurons, denoted as $\mathcal{N}_1$,

$$\mathcal{L}_N(\mathcal{N}_1) = -\sum_{j=1}^{l_i} \log \sigma \left(-\beta \log \frac{P(y_{i,j}^-|x_i, y_{i,<j}^-)}{P_{\text{ref}}(y_{i,j}^-|x_i, y_{i,<j}^-)} \times r_{i,j} \right). \quad (7)$$

4.3 Alignment Improving

For the learning stage, we further improve the alignment of the LLM that has unlearned the unaligned knowledge. We adopt DPO (Rafailov et al., 2023) algorithm for training, and also leverage its computed reward score to distinguish the key tokens and noisy ones.

Noisy Tokens Identification. We also identify the noisy tokens in the negative responses using the reward score in DPO, for reducing their harmful influence on learning other key tokens. As DPO requires to compare the token probabilities of the current-step LLM and its original probability, the reward of the key tokens initially own small values and increase smoothly. However, the noisy ones typically lead to large reward values, and shock the training process (Chen et al., 2019). Therefore, we can utilize the reward scores dynamically computed

Task	Train / Test	Dataset	Num. Data
Math	Train	MetaMathQA	40,000
	Test	GSM8k MATH MAWPS TabMWP	1,319 5,000 2,065 1,000
QA	Train	ECQA QASC	7,598 8,134
	Test	ECQA QASC OBQA StrategyQA	2,194 926 500 687
IF	Train	UltraFeedback	23,976
	Test	AlpacaEval 2.0 Arena-Hard	805 500

Table 1: Statistics of the used datasets. "IF" denotes the instruction following tasks.

in the DPO process, to distinguish the key and noisy tokens, denoted as:

$$q_{i,j} = \begin{cases} 0, & r'_{i,j} \in \text{top } v\% \\ 1, & \text{others} \end{cases}, \quad r'_{i,j} = \frac{P(y_{i,j}^-|x_i, y_{i,<j}^-)}{P_{\text{ref}}(y_{i,j}^-|x_i, y_{i,<j}^-)}, \quad (8)$$

where v% is the hyper-parameter to control the threshold. In this way, we can identify the noisy tokens causing abnormal large rewards with weight 0, and key tokens with weight 1.

Fine-grained Learning with DPO. After obtaining the token weights, we also decompose the objective function of DPO into the token level, and add weights into the tokens from the negative response to provide fine-grained supervision. Formally, the revised objective function is as follows:

$$\begin{aligned} \mathcal{L}_D(\mathcal{N}_2) = -\log \sigma (\beta \sum_{j=1}^{l_i^+} \log \frac{P(y_{i,j}^+|x_i, y_{i,<j}^+)}{P_{\text{ref}}(y_{i,j}^+|x_i, y_{i,<j}^+)} \\ -\beta \sum_{j=1}^{l_i^-} \log \frac{P(y_{i,j}^-|x_i, y_{i,<j}^-)}{P_{\text{ref}}(y_{i,j}^-|x_i, y_{i,<j}^-)} \times q_{i,j}), \end{aligned} \quad (9)$$

where we only optimize the top-k_2% most important neurons, denoted as $\mathcal{N}_2$.

5 Experiment

5.1 Experimental Settings

In this section, we introduce the details of our evaluation process, including downstream datasets, baselines in the evaluation, and the implementation details of our proposed method.

Datasets. We conduct the three downstream scenarios for the comprehensive evaluation, *i.e.,* question-

Methods	Question-Answering Tasks					Mathematical Reasoning Tasks				
	ECQA	QASC	OBQA	StrategyQA	Avg.	GSM8k	MATH	MAWPS	TabMWP	Avg.
SFT LLM	69.92	55.51	52.60	55.75	58.45	55.9	11.8	79.9	56.7	51.1
+ SFT	69.14	55.40	49.80	59.24	58.40	56.2	11.8	80.0	57.4	51.4
+ RFT	71.15	57.24	54.40	56.33	59.78	54.7	12.0	80.2	55.2	50.5
+ DPO	75.07	60.37	57.00	59.53	62.99	56.6	12.2	81.7	57.3	52.0
+ R-DPO	75.52	61.56	58.40	59.83	63.83	56.9	12.3	82.3	57.2	52.2
+ IPO	47.86	43.20	41.80	43.38	44.06	**58.0**	12.9	82.4	55.5	52.2
+ BCO	68.87	55.18	45.40	57.21	56.67	57.2	12.4	81.8	56.3	51.9
+ SimPO	62.76	52.27	46.80	53.71	53.89	57.9	12.8	82.1	56.7	52.4
+ NPO	70.56	56.59	52.80	56.04	59.00	56.4	12.3	80.1	56.5	51.3
+ Vanilla PPO	70.65	55.29	53.40	56.33	58.92	55.2	11.6	79.4	56.5	50.7
+ PPO A2C	71.06	55.18	53.00	58.37	59.40	55.2	11.7	82.1	55.8	51.2
+ ALLO	**75.93**	**62.31**	**59.60**	**60.84**	**64.67**	56.6	**13.0**	**82.5**	**58.1**	**52.6**

Table 2: Experimental results on question answering tasks and mathematical reasoning tasks. Avg. is the average accuracy of all sub-tasks. The best is denoted in bold and the second best is underlined.

Methods	Instruction Following Tasks		
	AlpacalEval 2.0	Arena-Hard	Avg.
SFT LLM	50.00	50.00	50.00
+ SFT	49.44	61.50	55.47
+ RFT	50.06	53.70	51.88
+ DPO	53.80	68.30	61.05
+ R-DPO	54.00	72.20	63.10
+ IPO	56.35	71.00	63.68
+ BCO	54.79	71.80	63.30
+ SimPO	54.92	69.30	62.11
+ NPO	50.06	51.10	50.58
+ Vanilla PPO	48.75	48.20	48.48
+ PPO A2C	53.50	57.80	55.65
+ ALLO	55.78	**74.90**	**65.34**

Table 3: Experimental results on instruction following tasks. Avg. is the average win rate of all sub-tasks. The best are denoted in bold and the second-best are underlined.

answering (QA), mathematical reasoning, and instruction following. The statistics information of each task is presented in Table 1.

• *QA tasks* require LLMs to perform multi-step reasoning to solve problems. We adopt ECQA (Aggarwal et al., 2021), QASC (Khot et al., 2020), OpenbookQA (Mihaylov et al., 2018), and StrategyQA (Geva et al., 2021) as the evaluation tasks. LLMs are fine-tuned on the training set of ECQA and QASC to adapt to the QA tasks.

• *Mathematica reasoning tasks* include four challenge tasks, *i.e.,* GSM8k (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), MAWPS (Koncel-Kedziorski et al., 2016), and TabMWP (Lu et al., 2023), containing problems with different levels of difficulty. To complete the mathematical knowledge and ability of LLMs, MetaMathQA (Yu et al., 2023) has been utilized to fine-tune the LLMs.

• *Instruction following tasks* assess the capacity of LLMs to follow human instructions. AlpacaEval 2.0 (Li et al., 2023) and Arena-Hard (Tianle Li*, 2024) are considered as the downstream tasks. We adopt the alpaca dataset (Taori et al., 2023) to fine-tune the base LLMs and UltraFeedback dataset (Cui et al., 2023) for the further training process (*e.g.,*, DPO, ALLO).

For QA tasks and mathematical tasks, accuracy has been adopted as the evaluation metric. For the instruction following tasks, we employ `gpt-3.5-turbo` as the judge model and report the win rate over the backbone model (*i.e.,* SFT LLM).

Baselines. We incorporate three categories of methods in the evaluation, including supervised fine-tuning (*i.e.,* SFT (Ouyang et al., 2022) and RFT (Liu et al., 2023)), reinforcement learning (*i.e.,* Vanilla PPO (Schulman et al., 2017) and PPO A2C (Mnih et al., 2016)), and alignment without RL (*i.e.,* DPO (Rafailov et al., 2023), R-DPO (Park et al., 2024), IPO (Azar et al., 2024), BCO (Jung et al., 2024), SimPO (Meng et al., 2024b), and NPO (Zhang et al., 2024)).

Implementation Details. In the experiment, we fine-tune LLaMA 2 7B (Touvron et al., 2023) on instruction datasets corresponding to the downstream scenarios to obtain the backbone model (*i.e.,* SFT LLM), and conduct further training processes based on this model in the evaluation. The details of hyper-parameters are presented in Table 5.

Forgetting Stage		Learning Stage		QASC	OBQA	MATH	MAWPS	AlpaceEval 2.0	Arena-Hard
TLR	Mask	TLR	Mask	Acc. (%)	Acc. (%)	Acc. (%)	Acc. (%)	WR (%)	WR (%)
✔	Top-k	✔	Top-k	62.31	59.60	13.0	82.5	55.78	74.90
✔	Top-k	✗	Top-k	62.42	59.00	12.4	82.5	55.60	75.10
✗	Top-k	✔	Top-k	61.56	58.20	12.7	82.3	55.47	73.20
✔	✗	✔	Top-k	61.77	58.80	13.1	82.4	55.22	73.20
✔	Top-k	✔	✗	61.66	58.20	12.7	81.7	53.98	69.70
✔	Last-k	✔	Top-k	62.20	59.40	12.5	82.5	55.29	70.80
✔	Top-k	✔	Last-k	61.77	59.00	10.2	70.9	51.74	61.20
-	-	✔	Top-k	62.20	59.20	12.3	82.2	55.60	72.60
✔	Top-k	-	-	56.16	53.20	11.8	79.9	51.37	50.20

Table 4: The results of ablation study. "Acc." and "WR" denote accuracy and win rate, respectively. "TLR" denotes the whether adopting token-level rewards in each stage. "Mask" indicates the neuron masking mechanism.

5.2 Main Results

The results of ALLO and baseline approaches in our evaluation are presented in Table 2 and Table 3.

According to the evaluation, we can observe that ALLO outperforms other baselines in almost all downstream scenarios and makes a great improvement over NPO and DPO, which are the backbone methods of ALLO. That is because ALLO makes great efforts to reduce the redundant elements in the alignment process, including neurons in LLMs and tokens in training data. Experimental results have shown the effectiveness of ALLO.

Besides, comparing the performance between the algorithm with fine-grained supervision signals (*e.g.,* ALLO, PPO A2C) and the algorithm without them (*e.g.,* DPO, Vanilla PPO), the effectiveness of the fine-grained supervision signals has been verified. Specifically, PPO A2C has achieved a 55.65% average win rate in instruction following tasks, while Vanilla PPO only achieved 48.48%. Instance-level supervision cannot focus on the details in the training data, which will optimize the erroneous parts and hurt the performance of the training methods. In contrast, token-level supervision signals can better identify whether the token is worthy to be learned, which reduces the redundancy of training content.

Moreover, the improvement brought by the unlearning method (*i.e.,* NPO) has demonstrated that aligned and unaligned knowledge are both stored in LLMs. In the training process of NPO, the LLMs are not exposed to new knowledge and new capacities, and only are guided to forget the unaligned knowledge. This phenomenon further verifies the importance of the unlearning stage and the existence of redundant neurons in LLMs. Without the redundant neurons, is difficult of LLMs to learn both aligned and unaligned knowledge simultaneously.

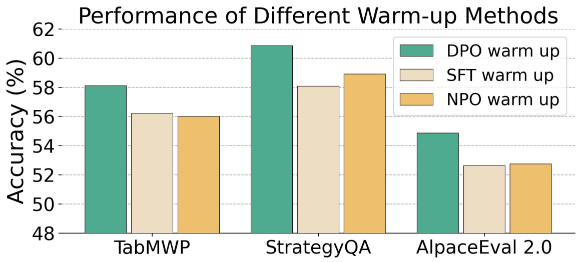

Figure 3: The experimental results of the influence of different warm-up methods on downstream tasks.

Finally, we can observe that ALLO outperforms DPO and its various (*e.g.,* R-DPO, SimPO) in all downstream scenarios, especially in the instruction following tasks. This is because DPO and its various guide LLMs to learn the positive and negative instances simultaneously, which will make LLMs confused about the aligned components in the negative instances. In contrast, ALLO first utilizes the unlearning process to lose the probability distribution in LLMs and leverage the fine-grained supervision signals to indicate the redundant tokens in the training data, to enhance the training efficiency.

5.3 Detailed Analysis

To further analyze our proposed ALLO, we conduct the ablation study, and analyze the influence of different warm-up methods and neuron mask ratio. Besides, we present a case study in Appendix D.

Ablation Study. To assess the effectiveness of each module in ALLO, we conduct the ablation study and present the evaluation results in Table 4.

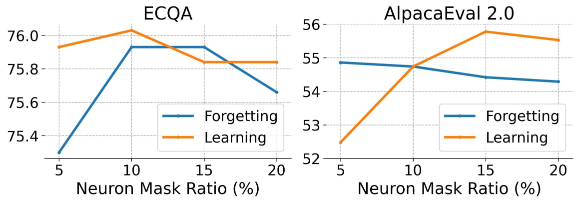

Figure 4: The experimental results of the different neuron mask ratios on ECQA and AlpaceEval 2.0, reporting the accuracy and win rate respectively. In the evaluation, we keep the mask ratio of one stage frozen and change the ratio of another stage.

According to the results, we can observe that removing any component of ALLO will hurt the performance, which has verified each module in ALLO is necessary and contributes to the final results of ALLO. Besides, in QA tasks, the results of removing the neuron mask and adopting the Last-k neuron mask indicate the existence of redundant neurons in LLMs, which is the same as our empirical study. For details, even adopting the Last-k neuron mask in Stage 2 (*e.g.,* 59.00% accuracy of OBQA) outperforms the variant without neuron masking (*e.g.,* 58.20% accuracy of OBQA). That is because training the whole neurons in LLMs will decrease the training efficiency, and redundant updates affect the performance of downstream tasks. Moreover, without the forgetting stage, ALLO still performs better than DPO in most tasks. The reason is that the token-level reward and the neuron masking mechanism reduce the redundancy and make the training process focus on effective details in the training instances, making better utilization of the information in the dataset.

Influence of Different Warm-up Methods. To assess the influence of different warm-up methods (*i.e.,* DPO, SFT, and NPO), we conduct the relative experiment and present the results in Figure 3. In all of the evaluation tasks, leveraging DPO to warm up LLMs and select important neurons has achieved the best performance that other warm-up methods. Whether SFT or NPO, these training methods only utilize a single part of the training dataset, *i.e.,* the positive responses or the negative responses, respectively. However, positive responses indicate the knowledge that LLMs should possess, and negative responses can locate unaligned knowledge stored in LLMs. These responses are both important and necessary in selecting the key neurons for the corresponding scenario. DPO can leverage the information in this data and guide LLMs to learn the aligned knowledge and eliminate unaligned one. In this warm-up process, the neurons related to downstream tasks will be modified largely, causing the large value of the gradient, which can more precisely locate the important neurons for the following training process.

Analysis the Ratio of Neuron Mask. We present the results of different ratios of neuron masks on the QA task (*i.e.,* ECQA) and the instruction following task (*i.e.,* AlpaceEval 2.0) in Figure 4. According to the evaluation results, we can observe that the performance first increases and then decreases, with the change of the neuron mask ratio. Concretely, for the ECQA task, selecting 10% neurons in the learning stage achieves the best performance, while selecting fewer or more neurons will hurt the accuracy of LLMs on downstream tasks. The increasing stage indicates that there are still several important neurons not been selected, which affects LLMs learning task-specific knowledge and abilities. After the increasing stage, the selected neurons set $\mathcal{N}$ contains more and more redundant neurons, interfering the learning process of other neurons and hurting the performance of the LLMs. The evaluation results have verified the existence of redundant updates in LLM alignment and shown that training an appropriate amount of neurons can reduce the redundancy and enhance the performance of LLMs.

6 Conclusion

In this paper, we proposed ALLO, an alignment method with low-redundant optimization, to train the most related neurons with the most useful supervised signals. In ALLO, we first estimated the importance of neurons in the LLM based on the weight changes of a reference model, and located the most related neurons for optimization. Then, we decomposed the alignment process into the forgetting and learning stages, where we leveraged token-level reward and DPO reward scores to identify the key tokens, and computing loss on them for training. Experimental results on question-answering tasks, mathematical reasoning tasks, and instruction following tasks have shown the effectiveness of ALLO.

As future work, we will consider leveraging ALLO on other important scenarios, *e.g.,* reducing hallucination. Besides, we will also implement ALLO in larger LLMs and multimodal LLMs to validate its effectiveness.

Limitations

In this section, we discuss the limitations of our work. First, we only conduct the experiment of ALLO on 7B LLMs, with the evaluation of the LLMs with larger scaling of parameters, because of the limitation of computation resources. Actually, we comprehensively assess the performance of ALLO and the existing competitive baseline methods in various downstream tasks, and the experiment results have verified the effectiveness of our proposed methods. Second, we adopt complex reasoning and human alignment tasks in our evaluation, which mainly assess the helpfulness of LLMs. The performance of ALLO on other aspects, *e.g.,* reducing hallucination and generating harmless response, has not been verified in this work. We leave it as future work. Finally, we do not consider the potential risk of ethics risk during LLM deployment and will investigate this issue in the future.

References

2024. Qwen2 technical report.

Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. 2021. Explanations for commonsenseqa: New dataset and models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3050–3065.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Benjamin Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. 2021. A general language assistant as a laboratory for alignment. *CoRR*, abs/2112.00861.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Rémi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics, 2-4 May 2024, Palau de Congressos, Valencia, Spain*, volume 238, pages 4447–4455.

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. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *CoRR*, abs/2204.05862.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosiute, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemí Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022b. Constitutional AI: harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*.

Loubna Ben Allal, Anton Lozhkov, Guilherme Penedo, Thomas Wolf, and Leandro von Werra. 2024. Cosmopedia.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2019. Machine unlearning. *CoRR*, abs/1912.03817.

Yinzhi Cao and Junfeng Yang. 2015. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 463–480. IEEE Computer Society.

Shreyas Chaudhari, Pranjal Aggarwal, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, Karthik Narasimhan, Ameet Deshpande, and Bruno Castro da Silva. 2024. RLHF deciphered: A critical analysis of reinforcement learning from human feedback for llms. *CoRR*, abs/2404.08555.

Kongyang Chen, Zixin Wang, Bing Mi, Waixi Liu, Shaowei Wang, Xiaojun Ren, and Jiaxing Shen. 2024a. Machine unlearning in large language models. *CoRR*, abs/2404.16841.

Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H. Chi. 2019. Top-k off-policy correction for a REINFORCE recommender system. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 456–464.

Zhipeng Chen, Kun Zhou, Wayne Xin Zhao, Junchen Wan, Fuzheng Zhang, Di Zhang, and Ji-Rong Wen. 2024b. Improving large language models via fine-grained reinforcement learning with minimum editing constraint. *CoRR*, abs/2401.06081.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4299–4307.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/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. *CoRR*, abs/2310.01377.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. 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*.

Mengnan Du, Fengxiang He, Na Zou, Dacheng Tao, and Xia Hu. 2024. Shortcut learning of large language models in natural language understanding. *Commun. ACM*, 67(1):110–120.

Duanyu Feng, Bowen Qin, Chen Huang, Zheng Zhang, and Wenqiang Lei. 2024. Towards analyzing and understanding the limitations of DPO: A theoretical perspective. *CoRR*, abs/2404.04626.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Zorik Gekhman, Gal Yona, Roee Aharoni, Matan Eyal, Amir Feder, Roi Reichart, and Jonathan Herzig. 2024. Does fine-tuning llms on new knowledge encourage hallucinations? *CoRR*, abs/2405.05904.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Trans. Assoc. Comput. Linguistics*, 9:346–361.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. ORPO: monolithic preference optimization without reference model. *CoRR*, abs/2403.07691.

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.

Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. 2023. Phi-2: The surprising power of small language models. *Microsoft Research Blog*.

Seungjae Jung, Gunsoo Han, Daniel Wontae Nam, and Kyoung-Woon On. 2024. Binary classifier optimization for large language model alignment. *CoRR*, abs/2404.04656.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. QASC: A dataset for question answering via sentence composition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8082–8090.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1152–1157.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

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

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. 2024. Rho-1: Not all tokens are what you need. *CoRR*, abs/2404.07965.

Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, and Andrew M. Dai. 2024. Best practices and lessons learned on synthetic data for language models. *CoRR*, abs/2404.07503.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J. Liu, and Jialu Liu. 2023. Statistical rejection sampling improves preference optimization. *CoRR*, abs/2309.06657.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2023. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *International Conference on Learning Representations (ICLR)*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *CoRR*, abs/2308.09583.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C. Lipton, and J. Zico Kolter. 2024. TOFU: A task of fictitious unlearning for llms. *CoRR*, abs/2401.06121.

Xiang Meng, Kayhan Behdin, Haoyue Wang, and Rahul Mazumder. 2024a. Alps: Improved optimization for highly sparse one-shot pruning for large language models. *CoRR*, abs/2406.07831.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024b. Simpo: Simple preference optimization with a reference-free reward.

Meta. 2024. Llama 3. *Meta Blog*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2381–2391.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1928–1937.

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 *NeurIPS*.

Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. 2024. Disentangling length from quality in direct preference optimization. *CoRR*, abs/2403.19159.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

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. *CoRR*, abs/2305.18290.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Jifeng Song, Kai Huang, Xiangyu Yin, Boyuan Yang, and Wei Gao. 2024. Large language model pruning. *CoRR*, abs/2406.06562.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Evan Frick Lisa Dunlap Banghua Zhu Joseph E. Gonzalez Ion Stoica Tianle Li*, Wei-Lin Chiang*. 2024. From live data to high-quality benchmarks: The arena-hard pipeline.

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, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Peiyi Wang, Lei Li, Zhihong Shao, R.X. Xu, Damai Dai, Yifei Li, Deli Chen, Y.Wu, and Zhifang Sui. 2023. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *CoRR*, abs/2312.08935.

Weiqi Wang, Zhiyi Tian, and Shui Yu. 2024a. Machine unlearning: A comprehensive survey.

Weixuan Wang, Barry Haddow, Wei Peng, and Alexandra Birch. 2024b. Sharing matters: Analysing neurons across languages and tasks in llms. *CoRR*, abs/2406.09265.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. 2023. Magicoder: Source code is all you need. *CoRR*, abs/2312.02120.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. LESS: selecting influential data for targeted instruction tuning. *CoRR*, abs/2402.04333.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. 2023. Doremi: Optimizing data mixtures speeds up language model pretraining. 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*.

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. *CoRR*, abs/2304.12244.

Yuanshun Yao, Xiaojun Xu, and Yang Liu. 2023. Large language model unlearning. *CoRR*, abs/2310.10683.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *CoRR*, abs/2309.12284.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. Mammoth: Building math generalist models through hybrid instruction tuning. *CoRR*, abs/2309.05653.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. 2024. Negative preference optimization: From catastrophic collapse to effective unlearning. *CoRR*, abs/2404.05868.

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.

Kun Zhou, Beichen Zhang, Jiapeng Wang, Zhipeng Chen, Wayne Xin Zhao, Jing Sha, Zhichao Sheng, Shijin Wang, and Ji-Rong Wen. 2024. Jiuzhang3.0: Efficiently improving mathematical reasoning by training small data synthesis models. *CoRR*, abs/2405.14365.

Algorithm 1: The ALLO algorithm.

Input : Training set $\mathcal{D} = \{\langle x_i, y_i^+, y_i^- \rangle\}_{i=1}^n$, the teacher model (GPT-4o), and the SFT model θ_{SFT}.
Output : A well-aligned model θ.

```
// 1. Locating Key Neurons
θ' ← DPO(θ_SFT);
for each neuron θ_i in warmed up model θ' do
    Calculate the importance of θ_i using Eq. 4;
Sort the importance of each neuron;
Select the top-k relative neurons into N;
// 2. Unaligned Knowledge Forgetting
for each instance ⟨x_i, y_i^+, y_i^-⟩ in D do
    if the data is sampled then
        The teacher model rewrites the negative response y_i^-;
Leverage the rewritten response to fine-tune the small LLM to obtain the θ_rm;
for each instance ⟨x_i, y_i^+, y_i^-⟩ in D do
    Identify the unaligned token using Eq. 5;
    Optimize the neurons in N using Eq. 7;
Obtain the model θ_forget forgetting unaligned knowledge;
// 3. Alignment Improving
for each instance ⟨x_i, y_i^+, y_i^-⟩ in D do
    Identify the noise token using Eq. 8;
    Optimize the neurons in N using Eq. 9;
Obtained the well-aligned model θ;
```

A Algorithm of ALLO

We present the pipeline of ALLO in Algorithm 1. The process of ALLO includes three stages, *i.e.,* locating key neurons, unaligned knowledge forgetting, and alignment improving.

B Details of Hyper-Parameters

To better understand and reproduce our proposed ALLO, we presented the hyper-parameters in ALLO in Table 5. The hyper-parameters are a little different between different downstream tasks, that is because these tasks are in different difficulty levels and require different abilities of LLMs. It should be noted that, to conduct a fair comparison, the hyper-parameters of baseline methods are also adjusted to adapt to the corresponding tasks for better performance.

C Prompt Templates of ALLO

In ALLO, we utilize prompts to guide the teacher model to rewrite the generated response from student models and induce the student model to solve the downstream tasks. The templates of the prompt in ALLO are presented in Table 6. For the solution rewriting process, we feed the problem, ground-truth reference, and generated response into the teacher model, with the instruction of rewriting in the prefix. Besides, for the downstream tasks, the instruction prefix and problem will be given into LLMs.

D Case Study

To better demonstrate our proposed ALLO, we present the case study on QA task (*i.e.,* ECQA) in Table 7. In this case, we can observe that LLM after DPO training still cannot catch the relation between "tickets" and the destination John needed to go to, and focus on the relation between "cross country" and "race track". This phenomenon has shown that unaligned knowledge is not eliminated and still exists in LLMs after DPO training. In contrast, after ALLO training, LLM can correctly seize on the key elements of the problem (*i.e.,* "ticket") and perform reasoning along the correct direction. That is because low-redundant optimization can reduce the redundant updates in the alignment process and make LLMs focus on the key knowledge and information.

Stage	Hyper-Parameter	Question-Answering	Mathematical Reasoning	Human Alignment
Stage 1	Learning Rate	1×10^{-7}	5×10^{-8}	1×10^{-7}
	Batch Size	32	512	128
	Selected Neuron Ratio	5%	5%	10%
	Threshold u	0.95	0.95	0.95
	β in NPO	0.1	0.1	0.1
Stage 2	Learning Rate	5×10^{-6}	1×10^{-6}	5×10^{-6}
	Batch Size	32	512	128
	Selected Neuron Ratio	10%	20%	15%
	Threshold v	20%	50%	20%
	β in DPO	0.1	0.1	0.1

Table 5: The details of hyper-parameters in the evaluation.

Distillation for Solution Rewriting	Given the problem and the correct solution, you need to correct the mistakes in the prediction to get the correct answer. You should write down the correct prediction and use "The answer is: " (without quotation mark) to identify the final answer. You should make minimal modifications. You should not copy the problem.
	
	### Problem:
	`{Problem}`
	
	### Correct solution:
	`{Ground-Truth Reference}`
	
	### Prediction:
	`{Generated Response from LLM}`
	
	### Correct prediction:

Prompt for Downstream Tasks	Below is an instruction that describes a task. Write a response that appropriately completes the request. ### Instruction: `{Problem to be solved}` ### Response: Let's think step by step.

Table 6: The template of prompt in ALLO.

Problem	John needed to go across the country so he ordered tickets online and got into his car. Where might he be going? Options: train station opera car rental shop show race track
Ground-Truth Solution	He was going to train station so he got into his car. John needed to go across the country by train so he ordered tickets online. The answer is train station
Generated Solution Before Training	Race tracks are in the outskirts of the city One can go to a race track by car Race tracks are in the country side. The answer is race track
Generated Solution After DPO	A person may go to another place by car When someone orders ticket online, it means that he is going to travel When someone goes from one place to another by car, he crosses the country When someone crosses the country, he might be going to the other end of the country. The answer is race track
Generated Solution After ALLO	John might be going to the train station as he ordered tickets online and got into his car. John needs to go across the country so he might be going to the train station. The answer is train station

Table 7: The case study for question-answering tasks.