# HyperDPO: Hypernetwork-based Multi-Objective Fine-Tuning Framework

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

In LLM alignment and many other ML applications, one often faces the *Multi-Objective Fine-Tuning (MOFT)* problem, *i.e.* fine-tuning an existing model with datasets labeled w.r.t. different objectives simultaneously. To address the challenge, we propose the *HyperDPO* framework, a hypernetwork-based approach that extends the Direct Preference Optimization (DPO) technique, originally developed for efficient LLM alignment with preference data, to accommodate the MOFT settings. By substituting the Bradley-Terry-Luce model in DPO with the Plackett-Luce model, our framework is capable of handling a wide range of MOFT tasks that involve listwise ranking datasets. Compared with previous approaches, HyperDPO enjoys an efficient one-shot training process for profiling the Pareto front of auxiliary objectives, and offers flexible post-training control over trade-offs. Additionally, we propose a novel *Hyper Prompt Tuning* design, that conveys continuous weight across objectives to transformer-based models without altering their architecture. We demonstrate the effectiveness and efficiency of the HyperDPO framework through its applications to various tasks, including Learning-to-Rank (LTR) and LLM alignment, highlighting its viability for large-scale ML deployments.

## 1 Introduction

*Direct Preference Optimization (DPO)* [42] has been introduced as a memory- and computation-efficient alternative to the traditional *Reinforcement Learning with Human Feedback (RLHF)* [11, 35, 50] in Large Language Model (LLM) alignment. The method fine-tunes a pre-trained LLM with additional data that indicates the preference between different proposals w.r.t. customized objectives, such as safety, verbosity, coherence, *etc.* [61]. The idea of DPO is to reparametrize the *reward function* in RLHF and guide the training process in a supervised learning manner with the preference data.

LLM alignment also intersects with the *Multi-Objective Optimization (MOO)* problem, which involves fine-tuning a model w.r.t. multiple objectives simultaneously [21, 43, 61, 65]. In many MOO scenarios within machine learning, a pre-existing model optimized for one or more *main objectives* is further aligned to a set of *auxiliary objectives* without significantly detracting the model's performance on the main objectives in order to achieve certain desirable properties [34, 45]. This specific scenario is termed the *Multi-Objective Fine-Tuning (MOFT)* problem. As auxiliary objectives may conflict with each other, the notion of alignment is generalized to achieving the *Pareto optimality* in the MOFT setting, where the goal is to profile the *Pareto front*, representing a spectrum of trade-off solutions where no single auxiliary objective can be improved without compromising another. For more related works in LLM Alignment and MOO, we refer to Appendix A.

Submitted to Workshop on Fine-Tuning in Machine Learning (FITML@NeurIPS 2024). Do not distribute.

In this work, we address the task of multi-objective fine-tuning in a broad context through our proposed *HyperDPO* framework. This hypernetwork-based multi-objective fine-tuning framework is designed to (1) generalize DPO to the MOFT setting, (2) profile the Pareto front of the auxiliary objectives while maintaining the model performance on the main objectives, and (3) offer as flexible post-training controls over the trade-offs as possible.

### 1.1 Contributions

The main contributions of this work are as follows:

- We propose the *HyperDPO* method, a hypernetwork-based multi-objective fine-tuning framework that generalizes DPO to the multi-objective setting, profiles the Pareto front through one-shot training, and offers flexible post-training control over trade-offs.
- The HyperDPO framework is tested across diverse tasks, including Learning-to-Rank (LTR) and LLM alignment tasks, demonstrating its state-of-the-art performance to achieve comprehensive Pareto fronts against existing baselines and its efficiency across a wide range of high-dimensional, multi-objective, large-scale applications.
- For LLM applications, we develop a novel *Hyper Prompt Tuning* design that translates the continuous preference weight into a mask applied to the prefix embedding, effectively conveying weights across auxiliary objectives to the LLM without altering its underlying architecture.
- We further investigate the potential of the *temperature hypernetwork* for enhancing the flexibility of post-training control over the trade-offs, promising broader application of the HyperDPO framework to more complex multi-objective fine-tuning scenarios.

## 2 Preliminaries

In this section, we briefly introduce the proximal and direct preference optimization frameworks for fine-tuning LLMs with preference data, the MOO problem in machine learning settings, and related definitions.

### 2.1 Proximal and Direct Preference Optimization

Suppose we have a base LLM $p_{\mathrm{base}}(y|\boldsymbol{x})$, with $\boldsymbol{x}$ and $y$ being the content and the proposal, respectively, and $p_{\mathrm{base}}(y|\boldsymbol{x})$ the probability of generating response $y$ given $\boldsymbol{x}$. The goal of DPO is to fine-tune the model $p_{\mathrm{base}}(y|\boldsymbol{x})$ with the preference data $\mathcal{D}_{\mathrm{DPO}} = \{(\boldsymbol{x}^{(k)}, y_1^{(k)} \succ y_2^{(k)})\}_{k\in[N]}$, where $y_1^{(k)} \succ y_2^{(k)}$ denotes $y_1^{(k)}$ is more preferred than $y_2^{(k)}$ in the context of $\boldsymbol{x}^{(k)}$.

**Proximal Preference Optimization.** In RLHF [11] or *Proximal Preference Optimization (PPO)* [46], one first models the preference data by the *Bradley-Terry-Luce (BTL) model* [4]:

$$\mathbb{P}(y_1 \succ y_2|\boldsymbol{x}) = \frac{\exp(r(y_1|\boldsymbol{x}))}{\exp(r(y_1|\boldsymbol{x})) + \exp(r(y_2|\boldsymbol{x}))} = \sigma\left(r(y_1|\boldsymbol{x}) - r(y_2|\boldsymbol{x})\right), \tag{1}$$

where $r(y|\boldsymbol{x})$ is the reward function and $\sigma(\cdot)$ is the sigmoid function. PPO is carried out in the following two steps:

*Step 1.* Parametrize $r(y|\boldsymbol{x})$ by a neural network $r_\phi(y|\boldsymbol{x})$, where the parameters $\phi$ are trained by maximizing the log-likelihood of the preference data:

$$-\mathcal{L}(r_\phi; \mathcal{D}_{\mathrm{DPO}}) = \mathbb{E}_{(\boldsymbol{x}, y_1 \succ y_2)}\left[\log \sigma(r_\phi(y_1|\boldsymbol{x}) - r_\phi(y_2|\boldsymbol{x}))\right]; \tag{2}$$

*Step 2.* Fine-tune the base model $p_{\mathrm{base}}(y|\boldsymbol{x})$ by maximizing the expected reward with respect to the preference data while maintaining the KL-divergence between the refined model and the base model:

$$-\mathcal{L}(p_\theta; p_{\mathrm{base}}, r_\phi, \beta) = \mathbb{E}\left[r_\phi(y|\boldsymbol{x})\right] - \beta D_{\mathrm{KL}}(p_\theta \| p_{\mathrm{base}}) = \mathbb{E}\left[r_\phi(y|\boldsymbol{x}) - \beta \log \frac{p_\theta(y|\boldsymbol{x})}{p_{\mathrm{base}}(y|\boldsymbol{x})}\right], \tag{3}$$

where $\beta > 0$ is called the *temperature parameter* that controls the scale of the fine-tuning.

**Direct Preference Optimization.** The observation that motivates DPO [42] is that the reward function $r_\phi(\boldsymbol{x}, y)$ in (3) can be solved explicitly by letting $r_\theta(y|\boldsymbol{x}) = \beta \log \frac{p_\theta(y|\boldsymbol{x})}{p_{\text{base}}(y|\boldsymbol{x})}$, and therefore, the training process can be simplified to a one-shot logistic regression problem:

$$-\mathcal{L}(p_\theta; p_{\text{base}}, \beta, \mathcal{D}_{\text{DPO}}) = \mathbb{E}_{(\boldsymbol{x}, y_1 > y_2)} \left[ \log \sigma \left( \beta \log \frac{p_\theta(y_1|\boldsymbol{x})}{p_{\text{base}}(y_1|\boldsymbol{x})} - \beta \log \frac{p_\theta(y_2|\boldsymbol{x})}{p_{\text{base}}(y_2|\boldsymbol{x})} \right) \right]. \tag{4}$$

For completeness, we provide the proofs of the claim above in Appendix B.2.

### 2.2 Multi-Objective Optimization

In contrast to its single-objective counterpart, MOO considers the optimization problem with multiple objectives $\min_{\theta \in \Theta} \boldsymbol{\mathcal{L}}(\theta) = (\mathcal{L}_1(\theta), \mathcal{L}_2(\theta), \ldots, \mathcal{L}_m(\theta))$, where $\Theta$ is the feasible region. The goal is to profile the Pareto front, which is defined as follows:

$$\mathcal{P} = \{\theta \in \Theta : \nexists \theta' \in \Theta \text{ s.t. } \forall i \in [m], \mathcal{L}_i(\theta') \leqslant \mathcal{L}_i(\theta) \text{ and } \exists j \in [m], \mathcal{L}_j(\theta') < \mathcal{L}_j(\theta)\},$$

intuitively translating to the set of trade-off solutions that cannot be improved in one without worsening another. This concept is motivated by the possible conflicts between the objectives, and one may observe the details of the trade-offs from the Pareto front and make informed decisions accordingly.

For many machine learning applications, the MOO problem can be formulated as follows: given a dataset in the form of $\mathcal{D}_{\text{MOO}} = \{\mathcal{D}^j_{\text{MOO}}\}_{j \in [m]} = \{\{\boldsymbol{y}^{(k)}, z^{j,(k)}\}_{k \in [N]}\}_{j \in [m]}$, where $\boldsymbol{y}^{(k)}$ is the feature vector and $z^{j,(k)}$ is the $j$-th label of the $k$-th data point, the goal is to learn a model $f_\theta(\boldsymbol{y})$ that optimizes the following objectives:

$$\min_{\theta \in \Theta} \boldsymbol{\mathcal{L}}(f_\theta; \mathcal{D}_{\text{MOO}}) := (\mathcal{L}_1(f_\theta; \mathcal{D}^1_{\text{MOO}}), \mathcal{L}_2(f_\theta; \mathcal{D}^2_{\text{MOO}}), \ldots, \mathcal{L}_m(f_\theta; \mathcal{D}^m_{\text{MOO}})), \tag{5}$$

where $\mathcal{L}_j(f_\theta; \mathcal{D}^j_{\text{MOO}})$ is the loss function for the model $f_\theta$ with respect to the $j$-th objective, and the feasible region $\Theta$ is over all possible model parameters.

## 3 Methodology

In this section, we first introduce the multi-objective fine-tuning problem and its relation to the LLM alignment problem. Then, we present the HyperDPO framework, a hypernetwork-based multi-objective fine-tuning framework that generalizes the DPO framework to the MOFT setting and profiles the Pareto front of the auxiliary objectives.

### 3.1 Multi-Objective Fine-Tuning

The MOFT problem is a generalization of the LLM alignment problem to the multi-objective setting, where the goal is to fine-tune an existing base model $p_{\text{base}}(y|\boldsymbol{x})$ with respect to multiple *auxiliary* objectives simultaneously while maintaining the model performance on the *main* objective(s) that the base model is optimized for.

In this work, we formulate the MOFT problem as follows: given a set of item groups, each of which contains a list of items and corresponding labels with respect to $m$ different objectives. The dataset is in the form of

$$\mathcal{D}_{\text{MOFT}} = \{\mathcal{D}^j_{\text{MOFT}}\}_{j \in [m]} = \left\{ \left\{ \boldsymbol{x}^{(k)}, (\boldsymbol{y}^{(k)}_i)_{i \in [n^{(k)}]}, (z^{j,(k)}_i)_{i \in [n^{(k)}]} \right\}_{k \in [N]} \right\}_{j \in [m]}, \tag{6}$$

where $n^{(k)}$ is the number of items, $\boldsymbol{x}^{(k)} \in \mathbb{R}^D$ denotes the context and $\boldsymbol{y}^{(k)}_i \in \mathbb{R}^d$ denotes the feature vector of the $i$-th item, and $z^{j,(k)}_i \in \mathbb{R}^{n^{(k)}}$ denotes the $j$-th label of the $i$-th item, in the $k$-th item group, which often indicates the preference tendency of each item with respect to the $j$-th aspect.

For the relationship between the MOFT taks and the Learning-to-Rank (LTR) task, the LLM alignment task, and the MOO problem, we refer to Appendix B.1.

### 3.2 From Preference to Ranking

Recall that the DPO framework is obtained by *reparametrizing* the reward function in the PPO framework (3) by the ratio of the model probabilities as in (4), one may generalize the DPO framework from preference to ranking datasets, by switching from the BTL model to the Plackett-Luce (PL) model (*cf.* (1) and (7)) [28].

**Plackett-Luce Model.** PL model [37] is one of the most popular ways to model the ranking data [6]. In the PL model, the probability of ranking of the $j$-th aspect is defined as:

$$\mathbb{P}^j(\boldsymbol{y}_{\pi_1} \succ \boldsymbol{y}_{\pi_2} \succ \cdots \succ \boldsymbol{y}_{\pi_n}|\boldsymbol{x}) := \prod_{i=1}^{n} \frac{\exp(s(\boldsymbol{y}_{\pi_i}|\boldsymbol{x}))}{\sum_{k=i}^{n} \exp(s(\boldsymbol{y}_{\pi_k}|\boldsymbol{x}))}, \tag{7}$$

where $s(\boldsymbol{y})$ is the score function. The model is trained by aligning the $j$-th label with the top-one probability of the PL model $\mathbb{P}^j(\boldsymbol{y}_i \succ \boldsymbol{y}_{i'},\ \forall i' \neq i|\boldsymbol{x}) = \frac{\exp(s(\boldsymbol{y}_i|\boldsymbol{x}))}{\sum_{i'=1}^{n} \exp(s(\boldsymbol{y}_{i'}|\boldsymbol{x}))}$, *i.e.* the ListNet loss [6]:

$$-\mathcal{L}_{\text{ListNet}}(s_\theta; \mathcal{D}^j_{\text{LTR}}) = \mathbb{E}\left[\sum_{i=1}^{n} t(z_i^j) \log\left(\frac{\exp(s_\theta(\boldsymbol{y}_i|\boldsymbol{x}))}{\sum_{i'=1}^{n} \exp(s_\theta(\boldsymbol{y}_{i'}|\boldsymbol{x}))}\right)\right], \tag{8}$$

where the expectation is taken over the data distribution of $\mathcal{D}_{\text{LTR}}$, and $t(\cdot)$ is an appropriate normalization of the label vector $\boldsymbol{z}$ s.t. $\sum_{i=1}^{n} t(z_i) = 1$. Common choices include the softmax function for dense labels and $L_1$ normalization for sparse labels, corresponding to different modeling of the ranking data.

The log-likelihood $\log p_\theta(\boldsymbol{y}|\boldsymbol{x})$ is related to the score function $s_\theta(\boldsymbol{y}|\boldsymbol{x})$ by the softmax function, mimicking the BTL model (1) in which $\log p_\theta(\boldsymbol{y}|\boldsymbol{x})$ is related to the reward function $r_\theta(\boldsymbol{y}|\boldsymbol{x})$ by the sigmoid function. Therefore, given the ranking dataset $\mathcal{D}_{\text{MOFT}}$ (6), the loss function (4) of the $j$-th aspect can be modified to, incorporating the ListNet loss (8):

$$\mathcal{L}_{\text{ListNet}}(s_\theta; s_{\text{base}}, \beta_j, \mathcal{D}^j_{\text{LTR}}) = \mathbb{E}\left[\sum_{i=1}^{n} t(z_i^j) \log\left(\frac{\exp\left(\beta_j(s_\theta(\boldsymbol{y}_i|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}_i|\boldsymbol{x}))\right)}{\sum_{i'=1}^{n} \exp\left(\beta_j(s_\theta(\boldsymbol{y}_{i'}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}_{i'}|\boldsymbol{x}))\right)}\right)\right]. \tag{9}$$

The proof of this claim is provided in Appendix B.2. One should notice that when $t(\cdot)$ is the $L^1$ normalization, the ListNet loss (8) applied to the preference dataset $\mathcal{D}_{\text{DPO}}$ in the form of binary labels is equivalent to the DPO loss (4).

### 3.3 Hypernetwork-based MOFT

With the introduction of the ListNet loss (8), we may rewrite the MOFT problem (13) in a more detailed form:

$$\min_{\theta \in \Theta} \boldsymbol{\mathcal{L}}_{\text{ListNet}}(s_\theta; s_{\text{base}}, \boldsymbol{\beta}, \mathcal{D}_{\text{MOFT}}) = (\mathcal{L}_{\text{ListNet}}(s_\theta; s_{\text{base}}, \beta_j, \mathcal{D}^j_{\text{MOFT}}))_{j \in [m]}. \tag{10}$$

We assume the temperature parameter $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m) \in \mathbb{R}^m_+$ that controls the trade-off between the main objective and each auxiliary objective is fixed for now.

The most straightforward way to solve this MOO problem is to train the model $s_\theta$ with a linear combination of the preference data [65]:

$$\mathcal{L}_{\text{ListNet},\boldsymbol{w}}(s_\theta; s_{\text{base}}, \boldsymbol{\beta}, \mathcal{D}_{\text{MOFT}}) := \boldsymbol{w}^\top \boldsymbol{\mathcal{L}}_{\text{ListNet}}(s_\theta; s_{\text{base}}, \boldsymbol{\beta}, \mathcal{D}_{\text{MOFT}}), \tag{11}$$

where $\boldsymbol{w} = (w_1, w_2, \ldots, w_m)^\top \in \Delta^m$ is the weight vector over objectives, and with $\Delta^m$ being the $m$-dimensional simplex. As $\boldsymbol{w}$ iterates over $\Delta^m$, the model $s_\theta$ will be optimized over a specific trade-off between the main objective and the auxiliary objectives and possibly land on the Pareto front. This approach is known as the *weighted sum* or *linear scalarization* method in MOO literature and is able to obtain the complete Pareto front when it is convex [19].

An efficient way to profile the Pareto front of this MOFT problem is to use *hypernetworks* [34, 45]. The idea of hypernetworks is to design and train neural networks $s_\theta$ that not only take in the data but also depend on the weight vector $\boldsymbol{w}$. Intuitively, it formulates the MOO problem as a meta-learning problem, where the model $s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$ is trained to optimize the objectives over a distribution of weight vectors. In practice, in order to foster the exploration of the Pareto front, one may also incorporate artificial penalization terms to the loss function, such as the cosine similarity between the loss vector $\boldsymbol{\mathcal{L}}(s_\theta; s_{\text{base}}, \boldsymbol{\beta}, \mathcal{D}_{\text{MOFT}})$ of the model and the weight vector [45]:

$$\mathcal{G}_{\boldsymbol{w}}(s_\theta; s_{\text{base}}, \boldsymbol{\beta}) := -\cos\angle\left(\boldsymbol{w}, \boldsymbol{\mathcal{L}}_{\text{ListNet}}(s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x}); s_{\text{base}}, \boldsymbol{\beta}, \mathcal{D}_{\text{MOFT}})\right).$$

This penalization term intuitively confines the loss vector $\boldsymbol{\mathcal{L}}_{\text{ListNet}}$ to converging along the direction of the weight vector $\boldsymbol{w}$, which empowers the model to profile possibly concave Pareto fronts [25]. The loss function of the hypernetwork is thus defined as:

$$\begin{aligned} &\mathcal{L}_{\text{Hypernet}}(s_\theta; s_{\text{base}}, \boldsymbol{\beta}, \mathcal{D}_{\text{MOFT}}, \boldsymbol{\alpha}, \lambda) \\ :=&\mathbb{E}_{\boldsymbol{w} \sim \text{Dir}(\boldsymbol{\alpha})}\left[\mathcal{L}_{\text{ListNet},\boldsymbol{w}}(s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x}); s_{\text{base}}, \boldsymbol{\beta}, \mathcal{D}_{\text{MOFT}}) + \lambda \mathcal{G}_{\boldsymbol{w}}(s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x}); s_{\text{base}}, \boldsymbol{\beta})\right], \end{aligned} \tag{12}$$

where $\boldsymbol{\alpha}$ is the concentration parameter of the Dirichlet distribution over $\Delta^m$, and $\lambda$ is the penalization coefficient.

Due to the linearity of the DPO framework, one can show the linear transformation property in Proposition B.1. Powered by this property, our framework also offers post-training control over the trade-offs in the MOFT problem. As illustrated in Figure 4, one can adjust the trade-offs between the auxiliary objectives by adjusting the weight vector $\boldsymbol{w}$, and those between the fidelity to the base model and its performance on the fine-tuning datasets of the new model by scaling the temperature parameter $\boldsymbol{\beta}$ with (17). Furthermore, this property will serve as the foundation for the design of the temperature hypernetwork, which will be discussed in Appendix D.

The whole HyperDPO framework is summarized in Algorithm 1, Appendix C.1.

# 4 Experiments

In this section, we provide the detailed experiment design and results of the HyperDPO framework for different applications, including the learning-to-rank task and the LLM alignment task. We also analyze the results and compare them with state-of-the-art methods.

**Baselines.** We compare the HyperDPO framework with the following state-of-the-art baselines:

- *DPO Linear Scalarization (DPO-LS):* We first sample several weight vectors $\boldsymbol{w}$ over the simplex $\Delta^m$ and train the model $s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$ with the weighted sum loss (10). Notably, when $\boldsymbol{w}$ are unit vectors, it returns the result of the single-objective fine-tuning for reference.
- *DPO Soup [43]:* The DPO Soup method first trains DPO models for each auxiliary objective and then combines the models by a weighted sum.
- *MO-DPO [65]:* The MO-DPO method first chooses a weight vector $\boldsymbol{w}$ and then adds a margin reward term depending on $\boldsymbol{w}$ to the DPO loss to ensure multi-objective optimization.

For each baseline, we will use the same number of weight vectors $\boldsymbol{w}$ for a fair comparison. For details and further discussion of these baselines, we refer to Appendix C.1.

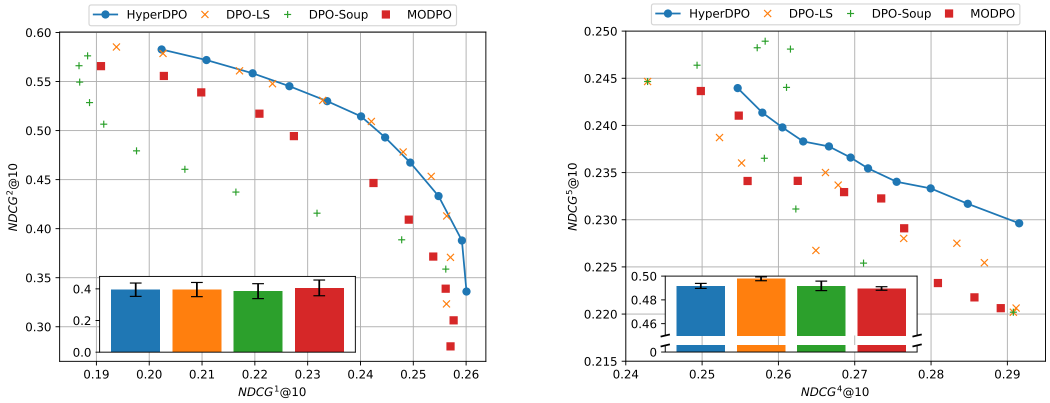
**Hypervolume Metric.** We adopt the *hypervolume (HV)* indicator [66] for evaluating the performance of MOO methods. Assuming the higher evaluation metrics indicate better performance, the hypervolume of the approximation $\hat{\mathcal{P}}$ to the real Pareto front $\mathcal{P}$ is defined as the volume of the dominated region of $\hat{\mathcal{P}}$ w.r.t. a reference point $\boldsymbol{r}$, *e.g.* when applied to minimization problems, the hypervolume is defined as follows: $\mathrm{HV}(\hat{\mathcal{P}}, \boldsymbol{r}) = \int_{\boldsymbol{x}<\boldsymbol{r}} \mathbf{1}_{\exists \boldsymbol{p}\in\hat{\mathcal{P}}, \boldsymbol{p}\leq\boldsymbol{x}} \mathrm{d}\boldsymbol{x}$. Higher hypervolume values indicate higher quality of the Pareto front.

## 4.1 Learning-to-Rank Task

We first test the HyperDPO framework on the learning-to-rank task. In this task, $\boldsymbol{x}^{(k)}$ in $\mathcal{D}_{\mathrm{MOFT}}$ denotes a query, and $\boldsymbol{y}_i^{(k)}$ denotes the feature vector of the $i$-th document, and $z_i^{j,(k)}$ denotes the score of the $i$-th document with respect to the $j$-th aspect. The goal is to provide a ranking $\boldsymbol{\pi}$ of the documents with respect to the scores $z_i^{j,(k)}$ for each query $\boldsymbol{x}^{(k)}$, for which the following Normalized Discounted Cumulative Gain (NDCG) (19) is used to evaluate its performance.

As the common practice in the LTR tasks, the information of the query $\boldsymbol{x}$ has often been incorporated into the feature vectors $\boldsymbol{y}_i$ in the upstream data processing. The hypernetwork $s_\theta(\cdot, \boldsymbol{w})$ is designed as a 2-layer transformer architecture of hidden dimension 64 with the weight vector $\boldsymbol{w}$ concatenated to the input of the first layer. We adopt the MSLR-WEB10K dataset [38] for the LTR task, with the main objective being the relevance label and the auxiliary objectives being (I) Query-URL Click Count, (II) URL Click Count, (III) URL Dwell Time, (IV) Quality Score 1, (V) Quality Score 2, with the relevance label, as 5 different auxiliary objectives ($m = 5$) for fine-tuning. For details of the experiment settings, we refer to Appendix C.2.

**Experiment Results.** We first apply the HyperDPO framework to the case where we only have 2 auxiliary objectives ($m = 2$) for better visualization. The results are shown in Figure 1, in which



(a) Objective I vs Objective II.

(b) Objective IV vs Objective V.

Figure 1: Comparison of Pareto fronts obtained by the HyperDPO framework and the baselines on the MSLR-WEB10K dataset with 2 auxiliary objectives. Two axes denote the NDCG@10 of the two auxiliary objectives (the higher, the better). The inset plot shows the average NDCG@10 of the main objective, with the error bar denoting the standard deviation across the 11 sampled points.

| Method | Aux. HV | Avg. Main Score (±Std) | Training Time (s) | # Parameters |
|---|---|---|---|---|
| DPO-LS | $1.648 \times 10^{-3}$ | 0.3553 (± 0.0290) | 14649.15 | 551,232 |
| DPO Soup | $1.468 \times 10^{-3}$ | 0.3823 (± 0.0317) | 6061.69 | 250,615 |
| MO-DPO | $1.263 \times 10^{-3}$ | 0.3595 (± 0.0242) | 27059.70 | 801,792 |
| **HyperDPO** | $\mathbf{2.039} \times 10^{-3}$ | **0.4320** (±0.0277) | **4043.47** | **50,432** |

Table 1: Hypervolume metric and training time[1]of HyperDPO and the baselines on the MSLR-WEB10K dataset with 5 auxiliary objectives. The reference point is set to $(0, 0)$, and 11 points are produced for the hypervolume calculation. The main score refers to the NDCG@10 of the main objective.

Figure 1a presents the Pareto front of two sparse labels ($t(\boldsymbol{z}) = \boldsymbol{z}/\|\boldsymbol{z}\|_1$ in (8)) with a relatively easy-to-learn convex Pareto front, while Figure 1b presents the Pareto front of two dense labels ($t(\boldsymbol{z}) = \mathrm{softmax}(\boldsymbol{z})$ in (8)) with a more ill-posed Pareto front. HyperDPO obtains comprehensive and competitive Pareto fronts that dominate those of the baselines in both pairs of objectives. Notably, HyperDPO is able to obtain a smooth Pareto front in Figure 1b while the baselines fail to do so. With a common temperature parameter $\boldsymbol{\beta}$ used across all methods, the inset plots demonstrate that the superior performance of the HyperDPO framework is not at the cost of the main objective, as the NDCG@10 of the main objective is comparable or even slightly better to some of the baselines.

We also test the HyperDPO framework on a more complicated case where we have 5 auxiliary objectives ($m = 5$). The results in Table 1 demonstrate our HyperDPO framework is able to achieve a higher hypervolume metric with significantly less training time and number of parameters compared to the baselines and comparably good preservation of the performance on the main objective. While the computational cost of traditional methods, such as DPO-LS and MO-DPO, grows exponentially with the number of objectives, HyperDPO models are able to maintain a linear growth with almost intact performance, indicating the efficiency and capability of the HyperDPO framework in handling high-dimensional MOFT problems in the LTR task.

**Ablation Studies.** We provide the ablation studies of the HyperDPO framework on the LTR task in Appendix C.3. Specifically, we evaluate the sensitivity of the HyperDPO framework to the concentration parameter $\boldsymbol{\alpha}$ (*cf.* Appendix C.3.1) and the depth (capacity) of the hypernetwork (*cf.* Appendix C.3.2). Furthermore, we will introduce, discuss the suitability, and compare the performance of two different NN parametrizations of the hypernetwork $s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$ in Appendix C.3.3, namely (a) *Hypernetwork from scratch* and (b) *Augmentation hypernetwork*, which exhibit different trade-

[1] The training time refers to the duration of all training jobs required for computing the 11-point Pareto front, and HyperDPO is allowed for more training epochs before its convergence.

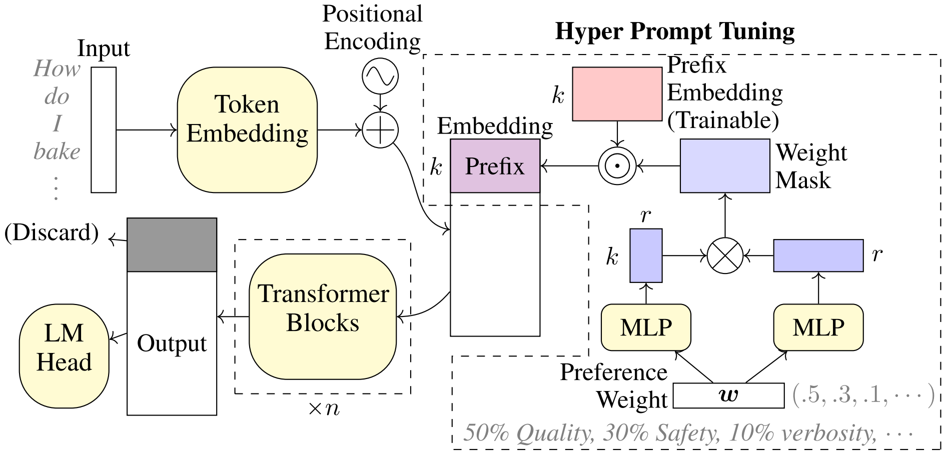


Figure 2: Illustration of the Hypernetwork Implementation in the HyperDPO Framework for the LLM Alignment Task. The proposed **Hyper Prompt Tuning** method, highlighted within the dashed box on the right, transforms the preference weight vector $\boldsymbol{w}$ into a weight mask and passes it to the LLM via prompt tuning. $k$ denotes the number of virtual tokens for prompt tuning, and $r$ is the rank of the weight mask.

offs between the performance and the computational cost and thus may serve different purposes in practice.

As discussed in Section B.3, besides the weight vector $\boldsymbol{w}$, the HyperDPO framework also offers post-training control over the temperature parameter $\boldsymbol{\beta}$ via the linear transformation property (Proposition B.1). We provide examples of the post-training control in Appendix D.1. However, the linear transformation property only offers proportional scaling of the temperature parameter $\boldsymbol{\beta}$, motivating the design and development of the more sophisticated ***Temperature Hypernetwork***. The details of our approach and some preliminary results are presented in Appendix D.

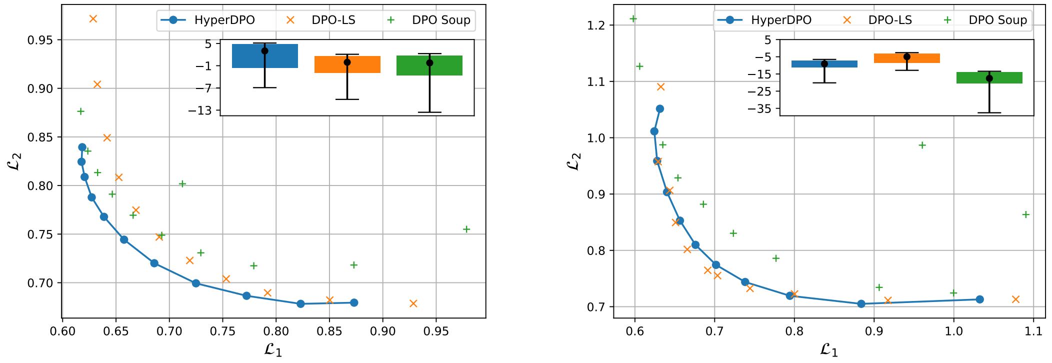
### 4.2 LLM Alignment Task

We then apply the HyperDPO framework to the LLM alignment task. In this task, $\boldsymbol{x}^{(k)}$ in $\mathcal{D}_{\mathrm{MOFT}}$ denotes a prompt, and $\boldsymbol{y}_i^{(k)}$ denotes the response generated by the LLM, and $z_i^{j,(k)}$ denotes the score of the $i$-th response with respect to the $j$-th aspect. The goal is to align the LLM to generate responses that satisfy the auxiliary objectives (*e.g.* verboseness, harmlessness, *etc.*) while maintaining its performance on general tasks (*e.g.* fluency, relevance, *etc.*).

The PKU-SafeRLHF dataset [21] is adopted for experiments, with each entry containing a prompt and a pair of responses annotated with preferences with respect to both harmlessness and helpfulness. The goal is to fine-tune the model to generate responses that are both harmless and helpful as a multi-objective optimization problem. We perform fine-tuning to the GPT-2 model [40] and the Alpaca-7B-Reproduced model [12] via Parameter-Efficient Fine-Tining (PEFT) with $\alpha = 8$ and $r = 4$ in the low-rank adaptions (LoRA) to the modules within the model. For HyperDPO, we adopt the Hyper Prompt Tuning technique with $k = 8$ and $r = 4$. To ensure a fair comparison, baseline methods will also be augmented with the prompt tuning of $k = 8$ on top of LoRA. For details of the experiment settings, we refer to Appendix C.2.

**Hypernetwork Implementation.** We incorporate the information of the weight vector $\boldsymbol{w}$ into the LLM via a novel design, called ***Hyper Prompt Tuning (HPT)*** and shown in Figure 2. Inspired by Prompt Tuning [24], HPT augments the input embedding obtained post token embedding and positional encoding with a trainable prefix embedding block that is controlled by the weight vector $\boldsymbol{w}$. Specifically, HPT follows the following steps:

*Step 1.* HPT takes in a weight vector $\boldsymbol{w} \in \Delta^m$ that indicates our preference across additional objectives and, through two simple trainable MLPs, produces two matrices, the matrix product of which forms the weight mask;



(a) GPT-2

(b) Alpaca-7B-Reproduced

Figure 3: Comparison of Pareto fronts obtained by the HyperDPO framework and the baselines on the PKU-SafeRLHF dataset[2]. Two axes denote the expected cross entropy error of the two auxiliary objectives (the lower, the better). The inset plot shows the interquartile range (IQR) of the deviation of the log-likelihood of the response from the reference model across the test dataset.

*Step 2.* The weight mask is multiplied entrywise with a trainable prefix embedding block with $k$ virtual tokens;

*Step 3.* The prefix embedding block is then concatenated to the input embedding as a prefix and fed into the transformer blocks of the LLM.

In contrast to Multi-Task Prompt Tuning [58], which can only handle a finite number of tasks, one can pass a wide spectrum of preference information by HPT into the LLM, offering flexibility and versatility for our hypernetwork implementation.

**Experiment Results.** For all experiments, we have chosen a common temperature $\beta = 0.1$ to balance the trade-offs between the main and auxiliary objectives. HyperDPO achieves smooth and comprehensive Pareto fronts (*cf.* Figure 3) with higher hypervolume metrics and less training time (*cf.* Table 2) for both LLM architectures compared to the baselines, demonstrating the effectiveness of the HyperDPO framework in the large-scale LLM alignment tasks. Notably, as HyperDPO tackles a “meta-learning” problem that is intrinsically more challenging and thus demands more expressive power, the HyperDPO framework is less prone to overfitting and more robust to the choice of the hyperparameters compared to the baselines. Several ablation studies are provided in Appendix C.3.

## 5 Discussion

In this work, we propose the HyperDPO framework for multi-objective fine-tuning, which is inspired by the DPO framework and the hypernetwork-based MOO to profile the Pareto front of a wide range of multi-objective fine-tuning (MOFT) problems. Our method presented superior performance in both the learning-to-rank and the large-scale LLM alignment tasks with multiple auxiliary objectives compared to the state-of-the-art methods, demonstrating the effectiveness and efficiency of the HyperDPO framework in handling high-dimensional MOFT problems. Our newly proposed Hyper Prompt Tuning technique also provides a novel way to incorporate preference information into the LLM, offering flexibility for both the hypernetwork implementation and further research in the LLM alignment task. We also explored the possibility of temperature hypernetwork in supplementary materials and presented preliminary results, opening up new directions for future research. Our work has proven the potential of the HyperDPO framework, and we expect it to be further explored in various MOFT problems in the future.

[2] Due to the possible conflict between the prompt tuning and the MO-DPO method, we were unable to reproduce competitive results for the method, and Figure 5 offers the best result that we achieved by hyperparameter optimization (*cf.* discussions in Appendix C.1).

# References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[3] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[4] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[5] Christopher Burges, Robert Ragno, and Quoc Le. Learning to rank with nonsmooth cost functions. *Advances in neural information processing systems*, 19, 2006.

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pp. 129–136, 2007.

[7] David Carmel, Elad Haramaty, Arnon Lazerson, and Liane Lewin-Eytan. Multi-objective ranking optimization for product search using stochastic label aggregation. In *Proceedings of The Web Conference 2020*, pp. 373–383, 2020.

[8] Angelica Chen, Sadhika Malladi, Lily H Zhang, Xinyi Chen, Qiuyi Zhang, Rajesh Ranganath, and Kyunghyun Cho. Preference learning algorithms do not learn preference rankings. *arXiv preprint arXiv:2405.19534*, 2024.

[9] Sirui Chen, Yuan Wang, Zijing Wen, Zhiyu Li, Changshuo Zhang, Xiao Zhang, Quan Lin, Cheng Zhu, and Jun Xu. Controllable multi-objective re-ranking with policy hypernetworks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3855–3864, 2023.

[10] Weiyu Chen and James Kwok. Multi-objective deep learning with adaptive reference vectors. *Advances in Neural Information Processing Systems*, 35:32723–32735, 2022.

[11] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[12] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

[13] Na Dai, Milad Shokouhi, and Brian D Davison. Multi-objective optimization in learning to rank. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 1241–1242, 2011.

[14] Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*, 2024.

[15] Adrian Gambier and Essameddin Badreddin. Multi-objective optimal control: An overview. In *2007 IEEE international conference on control applications*, pp. 170–175. IEEE, 2007.

[16] Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. A general theoretical paradigm to understand learning from human preferences. *arXiv e-prints*, pp. arXiv–2310, 2023.

[17] Long P Hoang, Dung D Le, Tran Anh Tuan, and Tran Ngoc Thang. Improving pareto front learning via multi-sample hypernetworks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7875–7883, 2023.

[18] Jun Hu and Ping Li. Collaborative multi-objective ranking. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 1363–1372, 2018.

[19] Wilfried Jakob and Christian Blume. Pareto optimization or cascaded weighted sum: A comparison of concepts. *Algorithms*, 7(1):166–185, 2014.

[20] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[21] Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36, 2024.

[22] Marco Laumanns and Jiri Ocenasek. Bayesian optimization algorithms for multi-objective optimization. In *International Conference on Parallel Problem Solving from Nature*, pp. 298–307. Springer, 2002.

[23] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

[24] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[25] Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. *Advances in neural information processing systems*, 32, 2019.

[26] Xi Lin, Zhiyuan Yang, Qingfu Zhang, and Sam Kwong. Controllable pareto multi-task learning. *arXiv preprint arXiv:2010.06313*, 2020.

[27] Suyun Liu and Luis Nunes Vicente. The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning. *Annals of Operations Research*, pp. 1–30, 2021.

[28] Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, et al. Lipo: Listwise preference optimization through learning-to-rank. *arXiv preprint arXiv:2402.01878*, 2024.

[29] Tie-Yan Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.

[30] Debabrata Mahapatra and Vaibhav Rajan. Multi-task learning with user preferences: Gradient descent with controlled ascent in pareto optimization. In *International Conference on Machine Learning*, pp. 6597–6607. PMLR, 2020.

[31] Debabrata Mahapatra, Chaosheng Dong, Yetian Chen, and Michinari Momma. Multi-label learning to rank through multi-objective optimization. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4605–4616, 2023.

[32] Debabrata Mahapatra, Chaosheng Dong, and Michinari Momma. Querywise fair learning to rank through multi-objective optimization. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1653–1664, 2023.

[33] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022.

[34] Aviv Navon, Aviv Shamsian, Gal Chechik, and Ethan Fetaya. Learning the pareto front with hypernetworks. *arXiv preprint arXiv:2010.04104*, 2020.

[35] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[36] Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*, 2024.

[37] Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202, 1975.

[38] Tao Qin and Tie-Yan Liu. Introducing letor 4.0 datasets. *arXiv preprint arXiv:1306.2597*, 2013.

[39] Zhen Qin, Le Yan, Honglei Zhuang, Yi Tay, Rama Kumar Pasumarthi, Xuanhui Wang, Mike Bendersky, and Marc Najork. Are neural rankers still outperformed by gradient boosted decision trees? In *International Conference on Learning Representations (ICLR)*, 2021.

[40] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[41] Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From $r$ to $q^*$: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*, 2024.

[42] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[43] Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36, 2024.

[44] Yinuo Ren, Tesi Xiao, Tanmay Gangwani, Anshuka Rangi, Holakou Rahmanian, Lexing Ying, and Subhajit Sanyal. Multi-objective optimization via wasserstein-fisher-rao gradient flow. In *International Conference on Artificial Intelligence and Statistics*, pp. 3862–3870. PMLR, 2024.

[45] Michael Ruchte and Josif Grabocka. Scalable pareto front approximation for deep multi-objective learning. In *2021 IEEE international conference on data mining (ICDM)*, pp. 1306–1311. IEEE, 2021.

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

[47] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

[48] Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. Large language model alignment: A survey. *arXiv preprint arXiv:2309.15025*, 2023.

[49] Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18990–18998, 2024.

[50] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

[51] Robin Swezey, Aditya Grover, Bruno Charron, and Stefano Ermon. Pirank: Scalable learning to rank via differentiable sorting. *Advances in Neural Information Processing Systems*, 34: 21644–21654, 2021.

[52] Jie Tang, Huiji Gao, Liwei He, and Sanjeev Katariya. Multi-objective learning to rank by model distillation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5783–5792, 2024.

[53] Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Rémi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Ávila Pires, and Bilal Piot. Generalized preference optimization: A unified approach to offline alignment. *arXiv preprint arXiv:2402.05749*, 2024.

[54] Ma Guadalupe Castillo Tapia and Carlos A Coello Coello. Applications of multi-objective evolutionary algorithms in economics and finance: A survey. In *2007 IEEE congress on evolutionary computation*, pp. 532–539. IEEE, 2007.

[55] Michael Taylor, John Guiver, Stephen Robertson, and Tom Minka. Softrank: optimizing non-smooth rank metrics. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pp. 77–86, 2008.

[56] Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

[57] Yining Wang, Liwei Wang, Yuanzhi Li, Di He, and Tie-Yan Liu. A theoretical analysis of ndcg type ranking measures. In *Conference on learning theory*, pp. 25–54. PMLR, 2013.

[58] Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, and Yoon Kim. Multitask prompt tuning enables parameter-efficient transfer learning. *arXiv preprint arXiv:2303.02861*, 2023.

[59] Zhichao Wang, Bin Bi, Shiva Kumar Pentyala, Kiran Ramnath, Sougata Chaudhuri, Shubham Mehrotra, Xiang-Bo Mao, Sitaram Asur, et al. A comprehensive survey of llm alignment techniques: Rlhf, rlaif, ppo, dpo and more. *arXiv preprint arXiv:2407.16216*, 2024.

[60] Junkang Wu, Yuexiang Xie, Zhengyi Yang, Jiancan Wu, Jinyang Gao, Bolin Ding, Xiang Wang, and Xiangnan He. $\beta$-dpo: Direct preference optimization with dynamic $\beta$. *arXiv preprint arXiv:2407.08639*, 2024.

[61] Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems*, 36, 2024.

[62] Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is dpo superior to ppo for llm alignment? a comprehensive study. *arXiv preprint arXiv:2404.10719*, 2024.

[63] Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. Token-level direct preference optimization. *arXiv preprint arXiv:2404.11999*, 2024.

[64] Aimin Zhou, Bo-Yang Qu, Hui Li, Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Qingfu Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and evolutionary computation*, 1(1):32–49, 2011.

[65] Zhanhui Zhou, Jie Liu, Chao Yang, Jing Shao, Yu Liu, Xiangyu Yue, Wanli Ouyang, and Yu Qiao. Beyond one-preference-fits-all alignment: Multi-objective direct preference optimization. *arXiv preprint ArXiv:2310.03708*, 2023.

[66] Eckart Zitzler and Simon Künzli. Indicator-based selection in multiobjective search. In *International conference on parallel problem solving from nature*, pp. 832–842. Springer, 2004.

# A Related Works

**LLM Alignment.** LLM alignment has been a popular topic in the machine learning community. Reinforcement Learning from Human Feedback (RLHF) has been a groundbreaking technique for alignment [2, 11, 35, 46], which serves as a foundation for training models like GPT-4 [1], and several advances have been made in this direction [3, 14, 23]. To reduce computational complexity, Direct Preference Optimization (DPO) [42] has been proposed as an alternative to RLHF, and further developed in [16, 28, 36, 41, 49, 53, 60, 63, 65]. We refer readers to [48, 59] for comprehensive reviews on LLM alignment.

**Multi-Objective Optimization.** Multi-Objective Optimization (MOO) has been actively studied in control systems [15] and economics [54]. The main focus of the related research is the development of algorithms to profile Pareto fronts efficiently so as to understand the trade-offs between objectives. Traditional methods include the evolutionary algorithms [64] and Bayesian optimization [22]. Recently, gradient-based MOO methods have been studied in the machine learning settings [25, 27, 30, 44, 47]. Hypernetwork-based methods are also explored by a series of works [10, 17, 26, 34, 45].

**Learning-to-Rank (LTR).** Learning to Rank (LTR) [29] tasks differ from traditional supervised learning in that they do not associate each sample with a simple label; instead, an optimal order of items within a group to maximize metrics, *e.g.* Normalized Discount Cumulative Gain (NDCG) [20, 57]. Typically, LTR models score documents and rank them thereby. To bridge LTR with supervised learning, various differentiable losses have been proposed as the proxy to these metrics [5, 6, 39, 51, 55]. In the context of Multi-Objective LTR, existing work includes label aggregation [7, 13], loss aggregation [18, 31, 32, 52], and hypernetwork [9].

# B Missing Remarks and Proofs

In this section, we provide the remarks and proofs of the propositions and theorems mentioned in the main text.

## B.1 Remarks on the Multi-Objective Fine-Tuning Task

**Relation to the Learning-to-Rank Task.** Datasets in this particular form are closely related to the Learning-to-Rank (LTR) problem, as one may immediately derive a ranking of the items in each group by sorting with respect to the labels $z_i^{j,(k)}$. In general, the dataset (6) may contain not only $\binom{n}{2}$ pairwise preference data but also the comparative intensity of the preferences, necessitating generalized models to handle the MOFT task. The LTR task will be discussed in more detail in Section 4.1 as we present the application of the HyperDPO framework to it.

**Relation to the LLM Alignment.** The preference dataset $\mathcal{D}_{\mathrm{DPO}}$ in LLM alignment can be viewed as a special case of the MOFT problem, where the number of auxiliary objectives $m = 1$, the number of items (proposals) in each group $n = 2$, and the label $z_i^{1,(k)}$ is binary, being 1 if the $i$-th item is preferred over the other, and 0 otherwise. We also refer to Liu et al. [28], Song et al. [49] for more discussions on LLM alignment with listwise data.

**Relation to the MOO task.** MOFT is a generalization of the MOO problem (5) to the fine-tuning setting, where the model $f_\theta(\boldsymbol{y})$ is the new model $p_\theta(\boldsymbol{y}|\boldsymbol{x})$, and the dataset $\mathcal{D}_{\mathrm{MOO}}$ is the preference dataset $\mathcal{D}_{\mathrm{MOFT}}$ (6). The MOFT problem can be formulated in the MOO language as follows:

$$\min_{\theta\in\Theta} \boldsymbol{\mathcal{L}}(p_\theta; p_{\mathrm{base}}, \boldsymbol{\beta}, \mathcal{D}_{\mathrm{MOFT}}) = (\mathcal{L}_j(p_\theta; p_{\mathrm{base}}, \beta_j, \mathcal{D}^j_{\mathrm{MOFT}}))_{j\in[m]}, \tag{13}$$

in which the specific choices of the loss functions should be carefully designed to reflect the preferences in the dataset $\mathcal{D}_{\mathrm{MOFT}}$.

### B.2 Proofs of Reparametrization-Related Arguments

*Proof of (4).* Recall that in the second step of PPO, we consider the loss function (3) as follows:

$$\begin{aligned}-\mathcal{L}(p_\theta; p_{\text{base}}, r_\phi, \beta) &= \mathbb{E}_{(\boldsymbol{x},y)}\left[r_\phi(y|\boldsymbol{x}) - \beta\log\frac{p_\theta(y|\boldsymbol{x})}{p_{\text{base}}(y|\boldsymbol{x})}\right]\\ &= \int\left(r_\phi(y|\boldsymbol{x}) - \beta\log\frac{p_\theta(y|\boldsymbol{x})}{p_{\text{base}}(y|\boldsymbol{x})}\right)p_\theta(y|\boldsymbol{x})\mathrm{d}y,\end{aligned}$$

we calculate the functional derivative of the loss w.r.t. the density function $p_\theta(y|\boldsymbol{x})$:

$$\begin{aligned}&\frac{\delta\mathcal{L}(p_\theta; p_{\text{base}}, r_\phi, \beta)}{\delta p_\theta(y|\boldsymbol{x})} = \lim_{\epsilon\to 0}\frac{\mathcal{L}(p_\theta + \epsilon\delta p_\theta; p_{\text{base}}, r_\phi, \beta) - \mathcal{L}(p_\theta; p_{\text{base}}, r_\phi, \beta)}{\epsilon}\\ &= \lim_{\epsilon\to 0}\frac{1}{\epsilon}\left[\int\left(r_\phi(y|\boldsymbol{x}) - \beta\log\frac{p_\theta(y|\boldsymbol{x})}{p_{\text{base}}(y|\boldsymbol{x})} - \beta\frac{\epsilon\delta p_\theta(y|\boldsymbol{x})}{p_\theta(y|\boldsymbol{x})}\right)(p_\theta(y|\boldsymbol{x}) + \epsilon\delta p_\theta(y|\boldsymbol{x}))\mathrm{d}y\right.\\ &\qquad\left. - \int\left(r_\phi(y|\boldsymbol{x}) - \beta\log\frac{p_\theta(y|\boldsymbol{x})}{p_{\text{base}}(y|\boldsymbol{x})}\right)p_\theta(y|\boldsymbol{x})\mathrm{d}y\right]\\ &= \int\left(r_\phi(y|\boldsymbol{x}) - \beta\log\frac{p_\theta(y|\boldsymbol{x})}{p_{\text{base}}(y|\boldsymbol{x})} - \beta\right)\delta p_\theta(y|\boldsymbol{x})\mathrm{d}y.\end{aligned}$$

Let the functional derivative vanish, we obtain

$$r_\phi(y|\boldsymbol{x}) = \beta\log\frac{p_\theta(y|\boldsymbol{x})}{p_{\text{base}}(y|\boldsymbol{x})} + \beta,$$

*i.e.*

$$p_\theta(y|\boldsymbol{x}) \propto p_{\text{base}}(y|\boldsymbol{x})\exp\left(\frac{r_\phi(y|\boldsymbol{x})}{\beta}\right).$$

Since the likelihood $\mathbb{P}(y_1 \succ y_2|\boldsymbol{x})$ (1) in the BTL model only depends on the difference of the reward functions, $r_\phi(y|\boldsymbol{x})$ admits an arbitrary constant shift, and thus we assume $r_\phi(y|\boldsymbol{x})$ to be normalized in a way such that

$$\mathbb{E}\left[p_{\text{base}}(y|\boldsymbol{x})\exp\left(\frac{r_\phi(y|\boldsymbol{x})}{\beta}\right)\right] = 1,$$

which leads to the reparametrization $r_\theta(y|\boldsymbol{x}) = \beta\log\frac{p_\theta(y|\boldsymbol{x})}{p_{\text{base}}(y|\boldsymbol{x})}$, plugging which into the PPO loss (3) yields the DPO loss (4). □

*Proof of (9).* As in the derivation of the DPO loss (4) under the BTL model, we first consider the PPO algorithm for the PL model:

*Step 1.* Find the optimal score function $s_\phi(\boldsymbol{y}|\boldsymbol{x})$ that minimizes the loss function:

$$-\mathcal{L}_{\text{ListNet}}(s_\theta; \mathcal{D}^j_{\text{LTR}}) = \mathbb{E}\left[\sum_{i=1}^n t(z^j_i)\log\left(\frac{\exp(s_\phi(\boldsymbol{y}_i|\boldsymbol{x}))}{\sum_{i'=1}^n\exp(s_\phi(\boldsymbol{y}_{i'}|\boldsymbol{x}))}\right)\right]; \tag{14}$$

*Step 2.* Fine-tune the base model $s_{\text{base}}$ with the optimal score function $s_\phi$ by maximizing the expected score value while penalizing the KL divergence between the new model and the base model:

$$-\mathcal{L}(p_\theta; p_{\text{base}}, r_\phi, \beta) = \mathbb{E}\left[s_\phi(\boldsymbol{y}|\boldsymbol{x})\right] - \beta D_{\text{KL}}(p_\theta||p_{\text{base}}) = \mathbb{E}\left[s_\phi(\boldsymbol{y}|\boldsymbol{x}) - \beta\log\frac{p_\theta(\boldsymbol{y}|\boldsymbol{x})}{p_{\text{base}}(\boldsymbol{y}|\boldsymbol{x})}\right]. \tag{15}$$

For the optimization problem in the second step (15), following the same procedure as in the proof of (4), we solve the optimal $p_\theta$ by letting the functional derivative of the loss w.r.t. the density function $p_\theta(y|\boldsymbol{x})$ vanish and obtain

$$p_\theta(\boldsymbol{y}|\boldsymbol{x}) \propto p_{\text{base}}(\boldsymbol{y}|\boldsymbol{x})\exp\left(\frac{s_\phi(\boldsymbol{y}|\boldsymbol{x})}{\beta}\right). \tag{16}$$

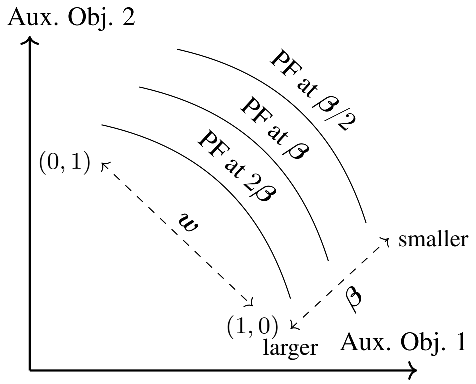


Figure 4: Conceptual Illustration of Available Post-Training Controls in the HyperDPO Framework with 2 auxiliary objectives.

By the assumption of the PL model and the ListNet loss, we have $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ modeled as the top-1 probability of the PL model and thus related to the score function $s_\theta(\boldsymbol{y}|\boldsymbol{x})$ via

$$p_\theta(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp(s_\theta(\boldsymbol{y}|\boldsymbol{x}))}{\sum_{i'=1}^n \exp(s_\theta(\boldsymbol{y}_{i'}|\boldsymbol{x}))}.$$

Let $p_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp(s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}))}{\sum_{i'=1}^n \exp(s_{\text{base}}(\boldsymbol{y}_{i'}|\boldsymbol{x}))}$, (16) can be rewritten as

$$\exp(s_\theta(\boldsymbol{y}|\boldsymbol{x})) \propto \exp\left(s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}) + \beta s_\phi(\boldsymbol{y}|\boldsymbol{x})\right),$$

*i.e.*

$$s_\theta(\boldsymbol{y}|\boldsymbol{x}) = s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}) + \beta s_\phi(\boldsymbol{y}|\boldsymbol{x}) + C,$$

where $C$ is a constant shift. By noticing that the softmax function in (14) is invariant to the constant shift of the score function $s_\phi(\boldsymbol{y}|\boldsymbol{x})$, we may choose certain normalization such that

$$s_\theta(\boldsymbol{y}|\boldsymbol{x}) = s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}) + \beta s_\phi(\boldsymbol{y}|\boldsymbol{x})$$

holds, plugging which into the loss (14) yields the reparametrized ListNet loss (9). □

### B.3 Linear Transformation Property

Due to the linearity of the DPO framework, one can show the following linear transformation property:

**Proposition B.1** (Linear Transformation Property)**.** *For any $\boldsymbol{\beta} \in \mathbb{R}_+^m$ and $\boldsymbol{w} \in \Delta^m$, we denote the hypernetwork trained by the hypernetwork loss (12) with temperature $\boldsymbol{\beta}$ as $s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x})$. Then $s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x})$ should satisfy the linear transformation that for any $c > 0$, we have*

$$s_{\theta,c\boldsymbol{\beta}}(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x}) = \left(1 - \frac{1}{c}\right) s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}) + \frac{1}{c} s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x}), \tag{17}$$

*up to a constant shift that does not depend on $\boldsymbol{y}$.*

*Proof of Proposition B.1.* By the definition of the hypernetwork $s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x})$, we have

$$\begin{aligned}
&s_{\theta,c\boldsymbol{\beta}}(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x}) \\
&= \operatorname*{arg\,min}_{s_\theta(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x})} \mathbb{E}\left[\sum_{i=1}^n t(z_i) \log\left(\frac{\exp\left(c\beta_j(s_\theta(\boldsymbol{y}_i, \boldsymbol{w}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}_i, \boldsymbol{w}|\boldsymbol{x}))\right)}{\sum_{i'=1}^n \exp\left(c\beta_j(s_\theta(\boldsymbol{y}_{i'}, \boldsymbol{w}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}_{i'}, \boldsymbol{w}|\boldsymbol{x}))\right)}\right)\right] \\
&= \operatorname*{arg\,min}_{s_\theta(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x})} \\
&\quad \mathbb{E}\left[\sum_{i=1}^n t(z_i) \log\left(\frac{\exp\left(\beta_j(cs_\theta(\boldsymbol{y}_i, \boldsymbol{w}|\boldsymbol{x}) + (1-c)s_{\text{base}}(\boldsymbol{y}_i, \boldsymbol{w}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}_i, \boldsymbol{w}|\boldsymbol{x}))\right)}{\sum_{i'=1}^n \exp\left(\beta_j(cs_\theta(\boldsymbol{y}_{i'}, \boldsymbol{w}|\boldsymbol{x}) + (1-c)s_{\text{base}}(\boldsymbol{y}_{i'}, \boldsymbol{w}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}_{i'}, \boldsymbol{w}|\boldsymbol{x}))\right)}\right)\right],
\end{aligned}$$

which, compared with the definition of $s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x})$

$$s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x}) = \underset{s_\theta(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x})}{\arg\min}\ \mathbb{E}\left[\sum_{i=1}^{n} t(z_i)\log\left(\frac{\exp\left(\beta_j(s_\theta(\boldsymbol{y}_i,\boldsymbol{w}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}_i,\boldsymbol{w}|\boldsymbol{x}))\right)}{\sum_{i'=1}^{n}\exp\left(\beta_j(s_\theta(\boldsymbol{y}_{i'},\boldsymbol{w}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}_{i'},\boldsymbol{w}|\boldsymbol{x}))\right)}\right)\right],$$

implies that

$$s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x}) = c s_{\theta,c\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x}) + (1-c)s_{\text{base}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x}),$$

rearranging which yields

$$s_{\theta,c\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x}) = \frac{1}{c}s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x}) - \frac{1-c}{c}s_{\text{base}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x}).$$

and the linear transformation property is proved. □

# C Additional Experiment Details

In this section, we present additional details of the experiments conducted in the main text, including further descriptions of the baseline implementations, and the ablation studies of the HyperDPO framework.

## C.1 Baseline Implementations

In the following, we will introduce and discuss the baseline methods used in the experiments in detail.

- *DPO Linear Scalarization (DPO-LS):* Given the base model $s_{\text{base}}$, for each weight vector $\boldsymbol{w} \in \mathbb{R}^m$, the DPO-LS method trains the new model $s_\theta$ with the loss function $\mathcal{L}_{\text{ListNet},\boldsymbol{w}}$ (11) and obtain $s_{\theta,\boldsymbol{w}}$ defined as

$$\begin{aligned} s_{\theta,\boldsymbol{w}} &= \underset{s_\theta}{\arg\min}\,\mathcal{L}_{\text{ListNet},\boldsymbol{w}}(s_\theta; s_{\text{base}},\boldsymbol{\beta},\mathcal{D}_{\text{MOFT}}) \\ &= \underset{s_\theta}{\arg\min}\,\boldsymbol{w}^\top\boldsymbol{\mathcal{L}}_{\text{ListNet}}(s_\theta; s_{\text{base}},\boldsymbol{\beta},\mathcal{D}_{\text{MOFT}}). \end{aligned}$$

  This model is a naive generalization from the weighted sum method in the MOO literature to the MOFT problem, and the main drawback is that it needs as many training jobs and models as the number of sampled weight vectors, which is computationally expensive.

- *DPO Soup [43]:* The DPO Soup model first trains $m$ models $s_{\theta,\boldsymbol{e}_i}$ for each unit vector $\boldsymbol{e}_i$ in the $m$-dimensional space, *i.e.* $m$ DPO models w.r.t. the $m$ auxiliary objectives, respectively, and then linearly combines the $m$ models to obtain the final model with the weight vector $\boldsymbol{w}$ in the parameter space. The DPO Soup method offers a more efficient way to combine the models trained with different auxiliary objectives, but it still requires $m$ training jobs and models for each auxiliary objective, and the performance of this model is largely dependent on the landscape of the parameter space of the neural network architecture. As depicted in Figure 1, the Pareto front obtained by the DPO Soup method may present unexpected curves, and Figure 3 shows that the DPO Soup method may even exhibit mode collapse for certain combinations.

- *MO-DPO [65]:* The MO-DPO method also starts with the training of $m$ models $s_{\theta,\boldsymbol{e}_i}$ for each unit vector $\boldsymbol{e}_i$ in the $m$-dimensional space, and then instead of linearly combining the parameters, MO-DPO conducts a new training job for each weight vector $\boldsymbol{w} \in \mathbb{R}^m$ with the following loss function:

$$\mathcal{L}_{\text{MO-DPO}}(s_\theta; s_{\text{base}},\boldsymbol{\beta},\mathcal{D}_{\text{MOFT}}) = \mathbb{E}\left[\sum_{i=1}^{n} t(z_i^j)\log\left(\frac{\exp\left(\beta_j r_{\theta,\boldsymbol{w}}^{\text{MO-DPO}}\right)}{\sum_{i'=1}^{n}\exp\left(\beta_j r_{\theta,\boldsymbol{w}}^{\text{MO-DPO}}\right)}\right)\right],$$

  where, for an arbitrary $i \in [m]$, $r_{\theta,\boldsymbol{w}}^{\text{MO-DPO}}$ is defined as

$$r_{\theta,\boldsymbol{w}}^{\text{MO-DPO}} := \frac{1}{w_i}\left(s_\theta(\boldsymbol{y}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}) - \sum_{i'\neq i} w_{i'}\left(s_{\theta,\boldsymbol{e}_i'}(\boldsymbol{y}|\boldsymbol{x}) - s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x})\right)\right). \tag{18}$$

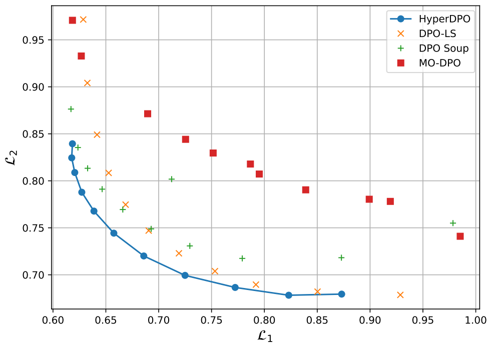


Figure 5: Comparison of Pareto fronts obtained by HyperDPO and the baselines on the PKU-SafeRLHF dataset with the GPT-2 model, including the MO-DPO method. The results for MO-DPO may not represent its best performance due to the possible conflict between the prompt tuning and the MO-DPO method.

As MO-DPO requires $m$ training jobs and one addition training job for each weight vector, it may require more training time and computational resources compared to the DPO-LS and DPO Soup methods. For the LLM alignment task, we observe MO-DPO suffers from unstable training caused by the $1/w_i$ vector in the expression (18) especially when $w_i$ is close to zero, and exhibit less competitive performance. We suspect that the conflict between the prompt tuning and the MO-DPO method may lead to the suboptimal performance of MO-DPO in the LLM alignment task.

## C.2 Experiment Settings

The HyperDPO framework is designed to address the limitations of the existing methods and provide a more efficient and effective way to profile the Pareto front of the MOFT problems, as summarized in Algorithm 1.

**Algorithm 1:** HyperDPO Framework

```
Data: Base model $s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x})$, dataset $\mathcal{D}_{\text{MOFT}}$, temperature $\boldsymbol{\beta}$, concentration parameter $\boldsymbol{\alpha}$,
      penalization coefficient $\lambda$ (Training); scale $c$, weight vector $\boldsymbol{w}$ (Post-Training Control).
Result: Hypernetwork $s_{\theta,\cdot\boldsymbol{\beta}}(\cdot,\cdot|\boldsymbol{x})$ (Training); $s_{\theta,c\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x})$ (Post-Training Control).
  // Training
  for $e = 1$ to $N_{\text{steps}}$ do
      Sample $\boldsymbol{w}' \sim \text{Dir}(\boldsymbol{\alpha})$;
      $\theta \leftarrow \theta - \eta\nabla_\theta \left[\mathcal{L}_{\text{ListNet},\boldsymbol{w}}(s_\theta(\cdot,\boldsymbol{w}'|\boldsymbol{x}); s_{\text{base}}, \boldsymbol{\beta}, \mathcal{D}_{\text{MOFT}}) + \lambda\mathcal{G}_{\boldsymbol{w}}(s_\theta(\cdot,\boldsymbol{w}'|\boldsymbol{x}); s_{\text{base}}, \boldsymbol{\beta})\right]$;
  end
  // Post-Training Control
  $s_{\theta,c\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x}) \leftarrow (1-1/c)\, s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}) + s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y},\boldsymbol{w}|\boldsymbol{x})/c$.
```

### C.2.1 Learning-to-Rank (LTR) Task.

**Normalized Discounted Cumulative Gain (NDCG).** The NDCG is a widely used metric in the LTR tasks, which measures the quality of the ranking of the items in the group. The NDCG is defined as

$$\text{NDCG}^j@\text{k}(\boldsymbol{\pi}) = \mathbb{E}_{(\boldsymbol{x},y,\boldsymbol{z}^j)}\left[\frac{\text{DCG}@\text{k}(\boldsymbol{\pi},\boldsymbol{z}^j)}{\max\limits_{\boldsymbol{\pi}'}\text{DCG}@\text{k}(\boldsymbol{\pi}',\boldsymbol{z}^j)}\right], \text{ where } \text{DCG}@\text{k}(\boldsymbol{\pi},\boldsymbol{z}^j) = \sum_{i=1}^{k}\frac{z^j_{\pi_i}}{\log_2(i+1)}. \tag{19}$$

| Method | GPT-2 | | Alpaca-7B-Reproduced | |
|---|---|---|---|---|
| | **HV** | **Training Time (s)** | **HV** | **Training Time (s)** |
| DPO-LS | 0.17668 | 15148.53 | 0.16873 | 94156.12 |
| DPO Soup | 0.18401 | 2755.51 | 0.14270 | 17138.74 |
| **HyperDPO** | **0.19424** | **1396.81** | **0.16885** | **8520.17** |

Table 2: Hypervolume metric and training time of HyperDPO and the baselines on the PKU-SafeRLHF dataset. The reference point for the hypervolume metric is set to $(1.1, 1.1)$, and 11 points are produced for the hypervolume calculation.

**NN architecture.** As the common practice in the LTR tasks, the information of the query $\boldsymbol{x}$ has often been incorporated into the feature vectors $\boldsymbol{y}_i$ by concatenation or other methods in the upstream data processing. We use a 2-layer transformer architecture of hidden dimension 128 for the base model $s_{\mathrm{base}}(\boldsymbol{y})$, and the hypernetwork $s_\theta(\cdot, \boldsymbol{w})$ is designed as a 2-layer transformer architecture of hidden dimension 64 with the weight vector $\boldsymbol{w}$ concatenated to the input of the first layer.

**Dataset.** We adopt the Microsoft Learning-to-Rank Web Search (MSLR-WEB10K) dataset [38] for the LTR task. The MSLR-WEB10K dataset consists of 10,000 groups ($N = 10^4$), each containing a list of webpages retrieved by the search engine in response to the query $\boldsymbol{x}^{(k)}$ and the corresponding features extracted from the webpage. Following the practice of [32], we treat the first 131 features as the feature vector ($\boldsymbol{y}_i^{(k)} \in \mathbb{R}^{131}$). We also identify the relevance label $\in [0 : 4]$ as the main objective used to train the base model, and the last 5 features, *viz.* (I) Query-URL Click Count, (II) URL Click Count, (III) URL Dwell Time, (IV) Quality Score 1, (V) Quality Score 2, with the relevance label, as 5 different auxiliary objectives ($m = 5$) for fine-tuning. The dataset is split into training (60%), validation (20%), and test (20%) datasets, and all results shown below are on the test split.

### C.2.2 LLM Alignment Task.

**Dataset.** The PKU-SafeRLHF dataset[3] [21] is adopted for experiments, which consists of 83.4k entries, each containing a prompt and a pair of responses annotated with preferences with respect to both harmlessness and helpfulness. The goal is to fine-tune the model to generate responses that are both harmless and helpful as a multi-objective optimization problem.

**Training Settings.** We perform fine-tuning to the GPT-2 model[4] [40] and the Alpaca-7B-Reproduced model[5] [12], following the practice of [65] via Parameter-Efficient Fine-Tining (PEFT) with $\alpha = 8$ and $r = 4$ in the low-rank adaptions (LoRA) to the modules within the model. For HyperDPO, we adopt the Hyper Prompt Tuning technique with $k = 8$ and $r = 4$. To ensure a fair comparison, baseline methods will also be augmented with the prompt tuning of $k = 8$ on top of LoRA. The HyperDPO framework is built upon the TRL package [56], and the implementation of the HPT is compatible with the PEFT package [33], which allows for easy integration with existing LLMs. All the experiments are conducted on a cluster with 8× NVIDIA A100 GPUs.

## C.3 Ablation Studies

In this section, we provide the ablation studies of the HyperDPO framework, including the sensitivity of the concentration parameter $\boldsymbol{\alpha}$ in the Dirichlet distribution, the depth of the hypernetwork, and the performance of two different NN parametrizations of the hypernetwork $s_{\theta,\boldsymbol{w},\boldsymbol{\beta}}(\cdot, \cdot|\boldsymbol{x})$, namely (a) *Hypernetwork from scratch* and (b) *Augmentation hypernetwork*.

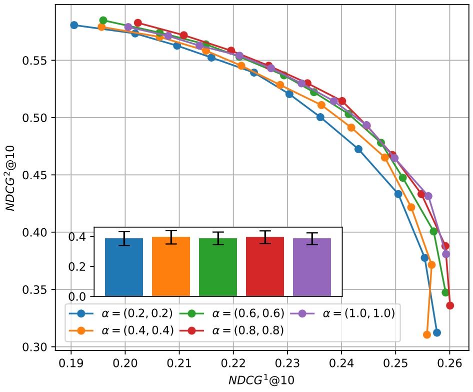
### C.3.1 Concentration Parameter $\boldsymbol{\alpha}$

The concentration parameter $\boldsymbol{\alpha}$ controls the span of the Dirichlet distribution from which the weight vector $\boldsymbol{w}$ is sampled and is the key parameter affecting the performance of the HyperDPO framework

[3] `https://huggingface.co/datasets/PKU-Alignment/PKU-SafeRLHF`

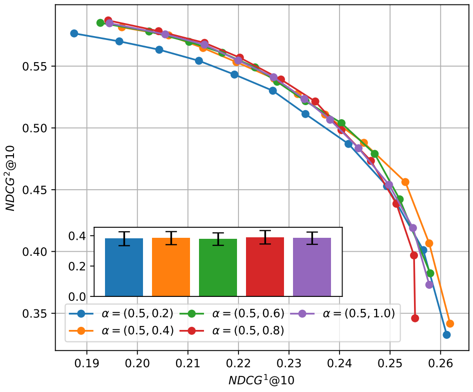
[4] `https://huggingface.co/openai-community/gpt2`

[5] `https://huggingface.co/PKU-Alignment/alpaca-7b-reproduced`



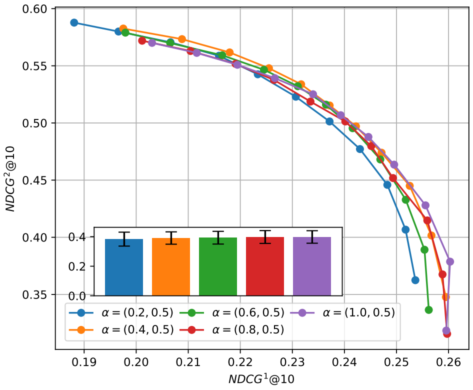
| $\boldsymbol{\alpha}$ | **Hypervolume** |
|---|---|
| $(0.2, 0.2)$ | 1.446e-01 |
| $(0.4, 0.4)$ | 1.445e-01 |
| $(0.6, 0.6)$ | 1.471e-01 |
| $(0.8, 0.8)$ | **1.473e-01** |
| $(1.0, 1.0)$ | 1.463e-01 |

(a) $\boldsymbol{\alpha} = (\alpha, \alpha)$ for $\alpha \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$.



| $\boldsymbol{\alpha}$ | **Hypervolume** |
|---|---|
| $(0.5, 0.2)$ | 1.451e-01 |
| $(0.5, 0.4)$ | **1.474e-01** |
| $(0.5, 0.6)$ | 1.466e-01 |
| $(0.5, 0.8)$ | 1.458e-01 |
| $(0.5, 1.0)$ | 1.464e-01 |

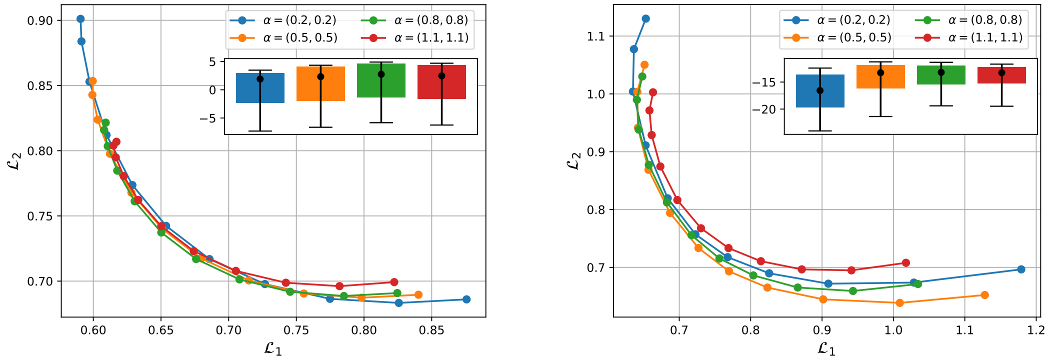
(b) $\boldsymbol{\alpha} = (0.5, \alpha)$ for $\alpha \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$.



| $\boldsymbol{\alpha}$ | **Hypervolume** |
|---|---|
| $(0.2, 0.5)$ | 1.447e-01 |
| $(0.4, 0.5)$ | **1.468e-01** |
| $(0.6, 0.5)$ | 1.445e-01 |
| $(0.8, 0.5)$ | 1.444e-01 |
| $(1.0, 0.5)$ | 1.445e-01 |

(c) $\boldsymbol{\alpha} = (\alpha, 0.5)$ for $\alpha \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$.

Figure 6: Ablation study on the impact of concentration parameter $\boldsymbol{\alpha}$ on the Pareto fronts obtained by the HyperDPO framework on the MSLR-WEB10K dataset (Objective I vs Objective II) with different settings of $\boldsymbol{\alpha}$. The hypervolume metric is shown in the table beside each figure.



(a) GPT2

(b) Alpaca-7B-Reproduced

Figure 7: Ablation study on the impact of the concentration parameter $\boldsymbol{\alpha}$ on the Pareto fronts obtained by the HyperDPO framework on the PKU-SafeRLHF dataset.

that should be carefully selected and validated. By the basic properties of the Dirichlet distribution, suppose $\boldsymbol{w} \sim \mathrm{Dir}(\boldsymbol{\alpha})$, then we have

$$\mathbb{E}[\boldsymbol{w}] = \frac{\boldsymbol{\alpha}}{\|\boldsymbol{\alpha}\|_1} := \overline{\boldsymbol{\alpha}}, \quad \mathrm{var}(\boldsymbol{w}) = \frac{\mathrm{diag}(\overline{\boldsymbol{\alpha}}) - \overline{\boldsymbol{\alpha}}\overline{\boldsymbol{\alpha}}^\top}{\|\boldsymbol{\alpha}\|_1 + 1}.$$

In general, the smaller the $\boldsymbol{\alpha}$, the more likely the weight vector $\boldsymbol{w}$ is close to the boundary of the simplex, and the larger the $\boldsymbol{\alpha}$, the more likely the weight vector $\boldsymbol{w}$ is concentrated around the expectation $\overline{\boldsymbol{\alpha}}$.

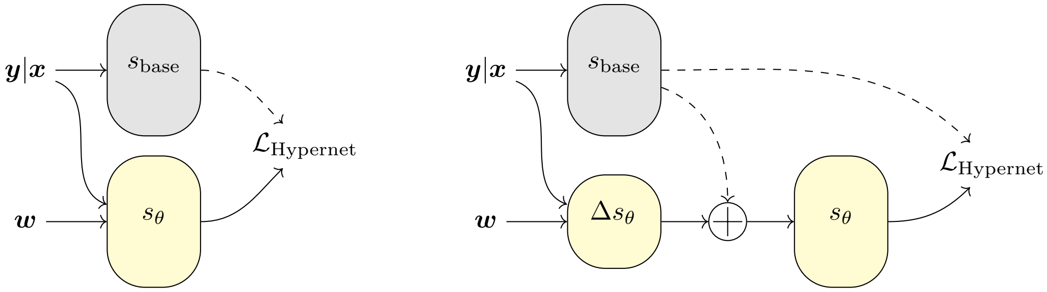
As the HyperDPO framework is generally robust to the choice of the concentration parameter $\boldsymbol{\alpha}$, we conduct ablation studies to investigate the impact of the concentration parameter $\boldsymbol{\alpha}$ on the performance of the HyperDPO framework in different settings. We first conduct experiments on the MSLR-WEB10K dataset with 2 auxiliary objectives (Query-URL Click Count vs URL Click Count) to investigate the impact of the concentration parameter $\boldsymbol{\alpha}$ on the performance of the HyperDPO framework. The results are shown in Figure 6. The experiment settings and plotting details are the same as in the main text.

As shown in Figure 6a, as the concentration parameter $\boldsymbol{\alpha}$ decreases, HyperDPO obtains a visually more comprehensive Pareto front thanks to more samples close to the boundary of the simplex. However, it is at the cost of a slightly undertrained model across the simplex, indicated by a lower hypervolume metric. It turns out that the choice of $\boldsymbol{\alpha}$ faces a trade-off between the diversity of the samples and the overall quality of the training, given a fixed training budget. Similar trade-offs are observed in Figure 6b and 6c when only one dimension of the concentration parameter $\boldsymbol{\alpha}$ is varied.

We also conducted experiments on the PKU-SafeRLHF dataset to investigate the impact of the concentration parameter $\boldsymbol{\alpha}$ on the performance of the HyperDPO framework on the LLM alignment task. The results are shown in Figure 7. A similar pattern is observed in this large-scale task, where a smaller choice of the concentration parameter $\boldsymbol{\alpha}$ leads to a more comprehensive Pareto front. However, it does not necessarily lead to a worse hypervolume metric, suggesting that the performance of HyperDPO here is less hindered by the expressive power of the model, which has already been abundant in the LLM, but rather by the diversity of the samples.

### C.3.2 Depth of the Hypernetwork

The depth of the hypernetwork structure is also crucial for the performance of the HyperDPO framework, as it determines the complexity of the hypernetwork structure and the expressiveness of the hypernetwork. We also use the MSLR-WEB10K dataset with 2 auxiliary objectives (Query-URL Click Count vs URL Click Count) to investigate the impact of the depth of the hypernetwork structure on the performance of the HyperDPO framework. The results are shown in Figure 9a, where the depth, referring to the number of transformer layers in the hypernetwork, is varied from 1 to 5. As shown in the figure, the performance of the HyperDPO framework is first significantly improved and gradually saturated with the increase of the depth of the hypernetwork structure. Besides, while


$\boldsymbol{y}|\boldsymbol{x}$
$s_{\text{base}}$
$\mathcal{L}_{\text{Hypernet}}$
$\boldsymbol{w}$
$s_\theta$
$\boldsymbol{y}|\boldsymbol{x}$
$s_{\text{base}}$
$\mathcal{L}_{\text{Hypernet}}$
$\boldsymbol{w}$
$\Delta s_\theta$
$s_\theta$

(a) Hypernetwork from Scratch

(b) Augmentation Hypernetwork

Figure 8: Illustration of two different parametrizations of the hypernetwork $s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$ in the HyperDPO framework. Dashed lines denote that backpropagation is not applied.

the hypervolume metric improves, the coverage of the Pareto front does not change significantly with the increase in the depth of the hypernetwork structure. This suggests that the concentration parameter $\boldsymbol{\alpha}$ may have a more significant impact on the diversity of the samples than the depth of the hypernetwork structure.

### C.3.3 Hypernetwork Parametrization

In general, one could adopt one of the two different parametrizations of the hypernetwork $s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$ in the HyperDPO framework.

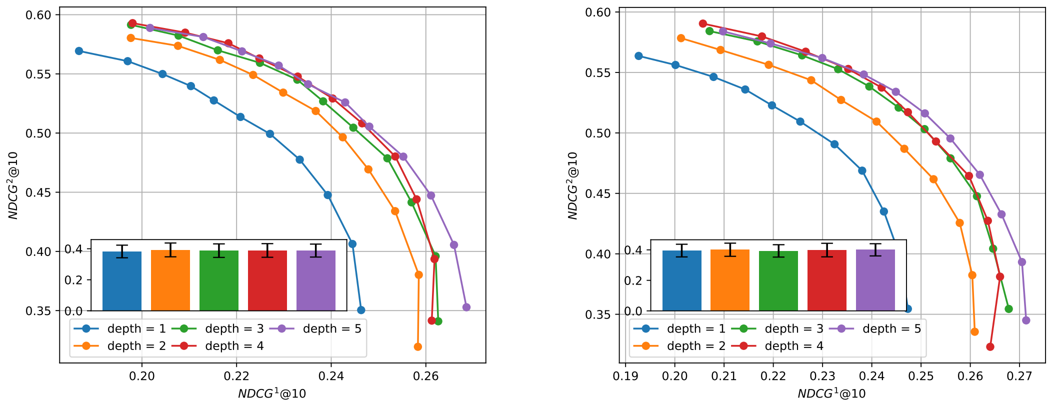
- *Hypernetwork from Scratch:* The hypernetwork $s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$ is a completely separate neural network from the base model $s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x})$. Depending on the specific design of the hypernetwork for additional inputs $\boldsymbol{w}$, the hypernetwork may or may not share the same architecture as the base model. The main advantage of this design is that it requires less memory and computation resources [42], and thus is more suitable for large-scale applications, *e.g.* LLMs.
- *Augmentation Hypernetwork:* As several works [8, 62] argue that DPO is prone to overfitting, one may curb the complexity of the hypernetwork for the score function $s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$ by only adding a first-order correction term to the base model $s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x})$ as:
$$s_\theta(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x}) = s_{\text{base}}(\boldsymbol{y}|\boldsymbol{x}) + \Delta s_\theta(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x}),$$
where the parameters in the base model are fixed, and the hypernetwork structure is only applied to the correction term $\Delta s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$. This design allows limited modification and reversibility to the base model and is thus suitable for applications where the fine-tuning is limited in budget, frequent, or expected to be minor.

The two parametrizations are illustrated in Figure 8a and 8b, respectively.

Both parametrizations can be seamlessly applied to the HyperDPO framework and easily switch between each other. In all the experiments presented in the main text, we have adopted the hypernetwork from scratch design for the HyperDPO framework. Figure 9b shows the results of the HyperDPO framework with the augmentation hypernetwork design on the same task as the previous ablation studies. Compared with Figure 9a, the augmentation hypernetwork achieves a roughly better performance than the hypernetwork from scratch design with the same depth, coinciding with the intuition that the augmentation hypernetwork benefited from the information provided by the base model and instead of learning the entire score function $s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$ from scratch, it only needs to learn the correction term $\Delta s_\theta(\cdot, \boldsymbol{w}|\boldsymbol{x})$. When the depth of the hypernetwork structure is increased, the performance of the augmentation hypernetwork is also improved, sharing the same trend as the hypernetwork from scratch design.

## D Towards Generalization to Temperature Hypernetwork

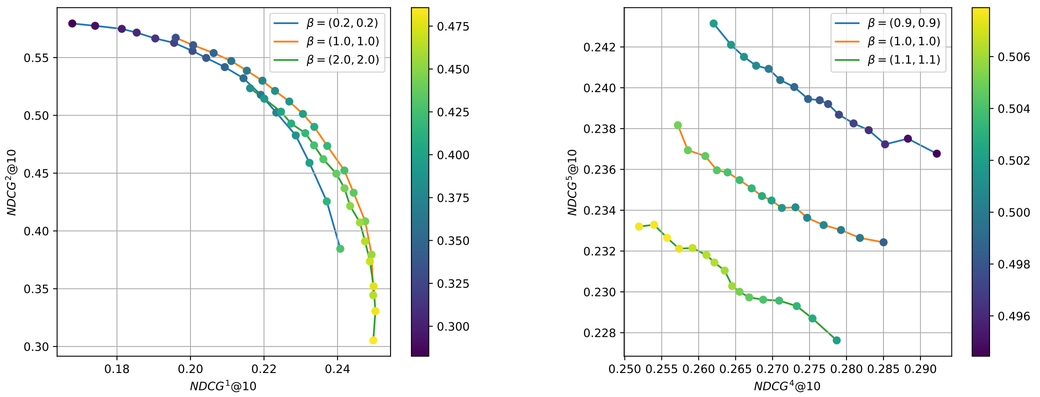
In this section, we consider further generalization of the hypernetwork structure to the temperature parameter $\boldsymbol{\beta}$. Generally speaking, the model should exhibit different Pareto fronts for different temperature parameters $\boldsymbol{\beta} \in \mathbb{R}_+^m$. By incorporating the temperature parameter $\boldsymbol{\beta}$ into the hypernetwork,



(a) Hypernetwork from Scratch

(b) Augmentation Hypernetwork

Figure 9: Ablation studies on the impact of the depth of the hypernetwork and the hypernetwork parametrizations on the Pareto fronts obtained by the HyperDPO framework on the MSLR-WEB10K dataset (Objective I vs Objective II).



(a) Objective I vs Objective II.

(b) Objective IV vs Objective V.

Figure 10: Examples of post-training control over temperature $\boldsymbol{\beta}$ on the MSLR-WEB10K dataset with 2 auxiliary objectives. Two axes denote the NDCG@10 of the two auxiliary objectives (the higher, the better). The colorbar denotes the NDCG@10 of the main objective.

we aim to output one score for each document $\boldsymbol{y}$, denoted by $s_\theta(\boldsymbol{y}, \boldsymbol{w}, \boldsymbol{\beta}|\boldsymbol{x})$, which reflects not only our preference $\boldsymbol{w}$ between different auxiliary objectives but also the trade-off between the main objective and the auxiliary objectives controlled by the vector $\boldsymbol{\beta}$.

### D.1 Current Post-Training Control over Temperature $\boldsymbol{\beta}$

Before we proceed to the training of the temperature hypernetwork, we would first present the current available post-training control over the temperature $\boldsymbol{\beta}$ in the HyperDPO framework without the temperature hypernetwork. As discussed in Appendix B.3 after Proposition B.1, the linear transformation property of the hypernetwork implies that the model can be scaled proportionally by a constant factor $c$ by a simple linear transformation of the output scores.

Figure 10 gives examples of the post-training control over the temperature $\boldsymbol{\beta}$ on the MSLR-WEB10K dataset with 2 auxiliary objectives. As the temperature $\boldsymbol{\beta}$ increases, the Pareto front shifts towards the direction where the main objective is more emphasized, which is consistent with our expectations. In Figure 10b, the two auxiliary objectives are in balance, and thus, the shifts of the Pareto fronts resemble that depicted in Figure 4. However, in Figure 10a, the unexpected shifting pattern is observed, which may reflect the complex interactions between the main and auxiliary objectives.

Motivated by the observation of complicated trade-offs between the main and auxiliary objectives, one may consider using different temperature $\beta$ for different objectives and also a disproportionate post-training scaling of the temperature parameter $\boldsymbol{\beta}$ to achieve more flexible control over the Pareto front. To this end, we propose to design the *temperature hypernetwork* to achieve this goal by incorporating the temperature parameter $\boldsymbol{\beta}$ into the hypernetwork structure in a similar manner as the weight vector $\boldsymbol{w}$.

### D.2 Temperature Hypernetwork Parametrization

Proposition B.1 implies that the temperature $\boldsymbol{\beta} \in \mathbb{R}_+^m$ actually has $m-1$ degrees of freedom, and thus we propose to use the following reparametrization by projecting $\boldsymbol{\beta}$ to its $L^1$-normalization $\overline{\boldsymbol{\beta}} := \boldsymbol{\beta}/\|\boldsymbol{\beta}\|_1 \in \Delta^m$, *i.e.*

$$s_\theta(\boldsymbol{y}, \boldsymbol{w}, \boldsymbol{\beta}|\boldsymbol{x}) = \left(1 - \frac{1}{\|\boldsymbol{\beta}\|_1}\right) s_{\mathrm{base}}(\boldsymbol{x}) + \frac{1}{\|\boldsymbol{\beta}\|_1} s_{\theta,\boldsymbol{w},\overline{\boldsymbol{\beta}}}(\boldsymbol{x}). \tag{20}$$

The training is then conducted by randomly sampling $\boldsymbol{\beta} \in \mathbb{R}_+^m$ over a certain distribution $\mathcal{D}(\beta)$ valued in $\mathbb{R}_+^m$, and the loss can be written as

$$\begin{aligned}&\mathcal{L}_{\mathrm{TempHypernet}}(s_\theta; s_{\mathrm{base}}, \mathcal{D}_{\mathrm{MOFT}}, \boldsymbol{\alpha}, \lambda)\\ :=&\mathbb{E}_{\boldsymbol{\beta}\sim\mathcal{D}(\boldsymbol{\beta})}\left[\mathbb{E}_{\boldsymbol{w}\sim\mathrm{Dir}(\boldsymbol{\alpha})}\left[\mathcal{L}_{\mathrm{ListNet},\boldsymbol{w}}(s_\theta(\cdot, \boldsymbol{w}, \boldsymbol{\beta}|\boldsymbol{x}); s_{\mathrm{base}}, \mathcal{D}_{\mathrm{MOFT}}) + \lambda\mathcal{G}_{\boldsymbol{w}}(s_\theta(\cdot, \boldsymbol{w}, \boldsymbol{\beta}|\boldsymbol{x}); s_{\mathrm{base}})\right]\right].\end{aligned} \tag{21}$$

The algorithm for the HyperDPO framework with the temperature hypernetwork is provided in Algorithm 2.

**Algorithm 2:** HyperDPO Framework with Temperature Hypernetwork

**Data:** Base model $s_{\mathrm{base}}(\boldsymbol{y}|\boldsymbol{x})$, dataset $\mathcal{D}_{\mathrm{MOFT}}$, concentration parameter $\boldsymbol{\alpha}$, penalization coefficient $\lambda$ (Training); temperature $\boldsymbol{\beta}$, weight vector $\boldsymbol{w}$ (Post-Training Control).

**Result:** Hypernetwork $s_\theta(\cdot, \cdot, \cdot|\boldsymbol{x})$ (Training); $s_\theta(\boldsymbol{y}, \boldsymbol{w}, \boldsymbol{\beta}|\boldsymbol{x})$ (Post-Training Control).

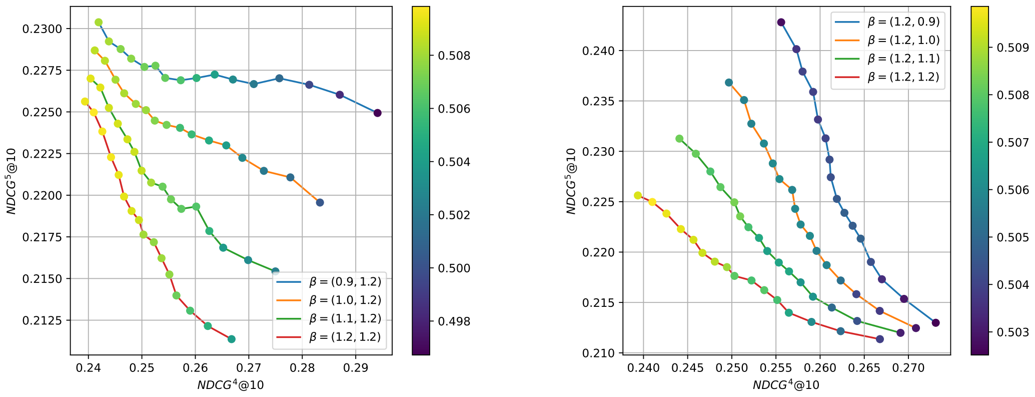
```
// Training
for $e = 1$ to $N_{\mathrm{steps}}$ do
    Sample $\boldsymbol{w}' \sim \mathrm{Dir}(\boldsymbol{\alpha})$, $\boldsymbol{\beta}' \sim \mathcal{D}(\boldsymbol{\beta})$;
    $\theta \leftarrow \theta - \eta\nabla_\theta\left[\mathcal{L}_{\mathrm{ListNet},\boldsymbol{w}}(s_\theta(\cdot, \boldsymbol{w}', \boldsymbol{\beta}'|\boldsymbol{x}); s_{\mathrm{base}}, \mathcal{D}_{\mathrm{MOFT}}) + \lambda\mathcal{G}_{\boldsymbol{w}}(s_\theta(\cdot, \boldsymbol{w}', \boldsymbol{\beta}'|\boldsymbol{x}); s_{\mathrm{base}})\right]$;
end
// Post-Training Control
$s_\theta(\boldsymbol{y}, \boldsymbol{w}, \boldsymbol{\beta}|\boldsymbol{x}) \leftarrow (1 - 1/c)\, s_{\mathrm{base}}(\boldsymbol{y}|\boldsymbol{x}) + s_{\theta,\boldsymbol{\beta}}(\boldsymbol{y}, \boldsymbol{w}|\boldsymbol{x})/c$.
```

In general, the distribution $\mathcal{D}(\boldsymbol{\beta})$ should be chosen to cover a reasonable range of temperature parameters $\boldsymbol{\beta}$ to ensure the problem is tractable, as our experiments reveal that the temperature hypernetwork may require highly expressive neural networks to capture the complex trade-offs both between the main and auxiliary objectives and across the auxiliary objectives.

### D.3 Preliminary Results

All experiments presented in this section are conducted on the MSLR-WEB10K dataset with 2 auxiliary objectives (Quality Score vs Quality Score 2) to investigate the performance of the HyperDPO framework with the temperature hypernetwork, as it provides better visualization and comparisons of the Pareto fronts with different temperature parameters $\boldsymbol{\beta}$. In particular, we adopt the augmentation hypernetwork design for the temperature hypernetwork for better expressive power and stability.

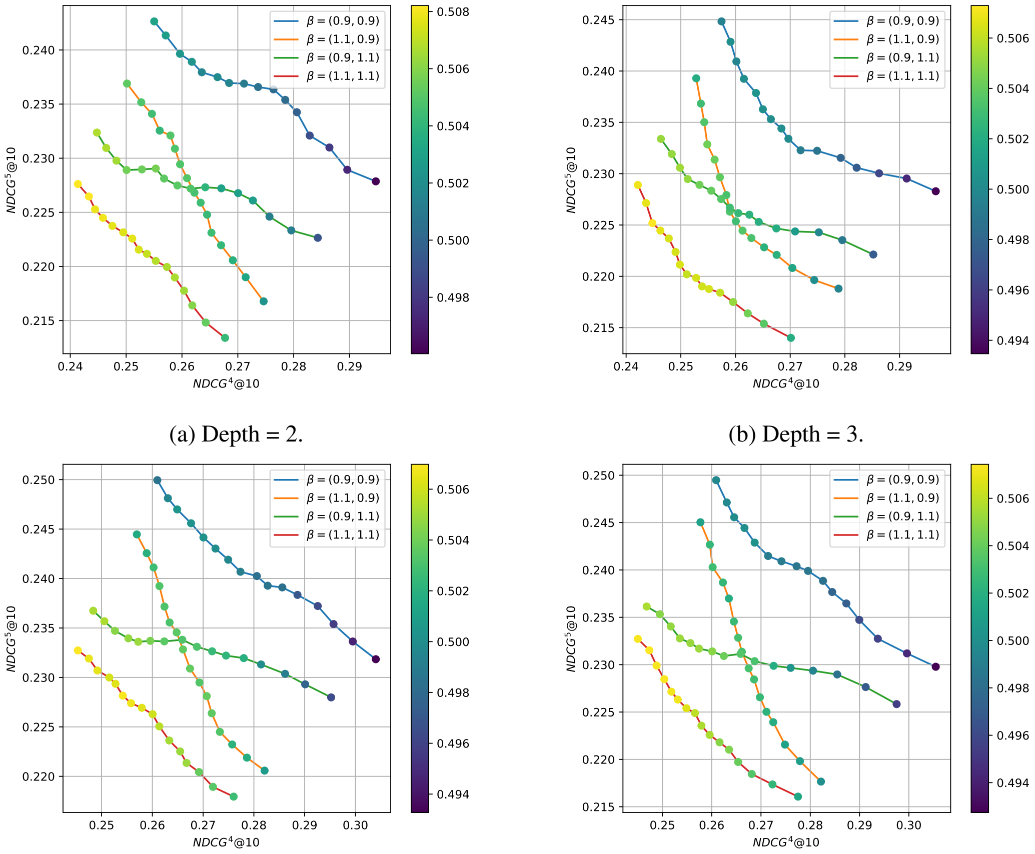
We provide the preliminary results of the HyperDPO framework with the temperature hypernetwork on the LTR task in Figure 11. The depth of the temperature hypernetwork is chosen to be 5, and the distribution $\mathcal{D}(\boldsymbol{\beta})$ is set to be $\mathrm{Unif}([0.67, 1.5]^2)$. The results demonstrate the temperature hypernetwork is capable of capturing the trade-off between the main objective and the auxiliary objectives for all kinds of temperature configurations $\boldsymbol{\beta}$, and the Pareto fronts exhibit expected behaviors with different $\boldsymbol{\beta}$. These results suggest that the temperature hypernetwork is a promising direction for the HyperDPO framework to achieve more flexible control over the Pareto front.



(a) $\boldsymbol{\beta} = (\beta, 1.2)$ for $\beta \in \{0.9, 1.0, 1.1, 1.2\}$.

(b) $\boldsymbol{\beta} = (1.2, \beta)$ for $\beta \in \{0.9, 1.0, 1.1, 1.2\}$.

Figure 11: Preliminary results of the HyperDPO framework with the temperature hypernetwork on the MSLR-WEB10K dataset (Objective IV vs Objective V). The colorbar denotes the NDCG@10 of the main objective.



(a) Depth = 2.

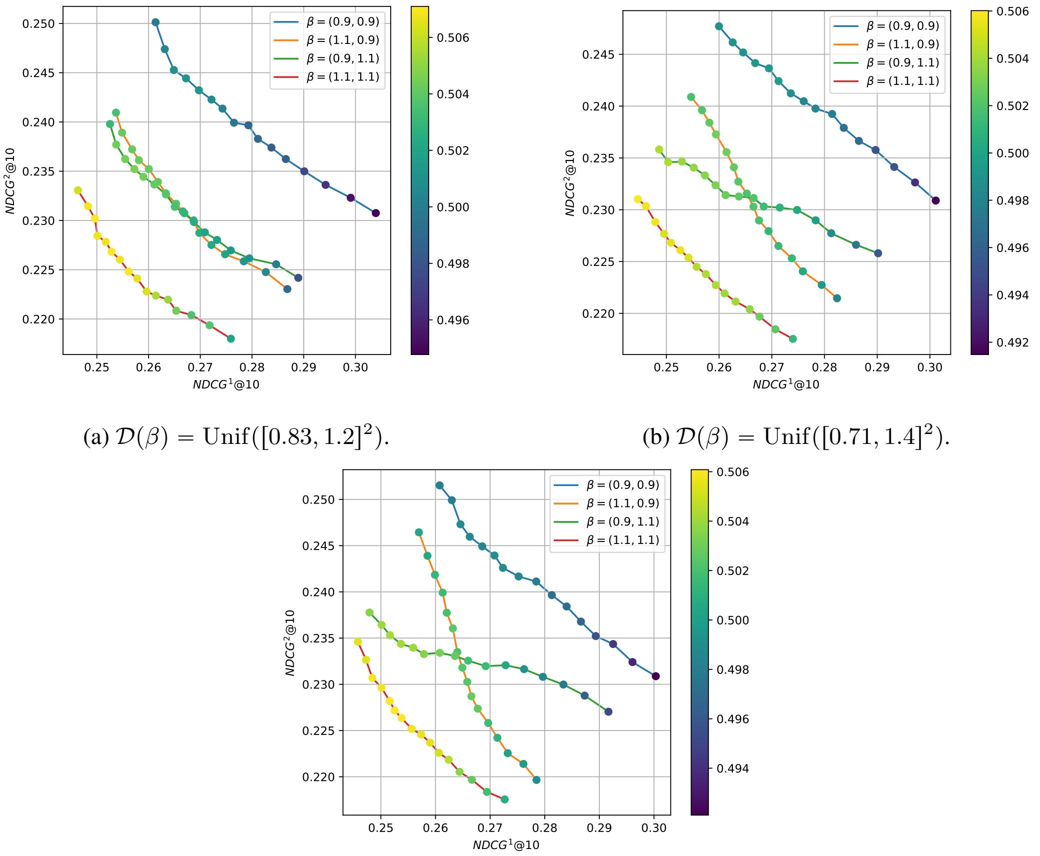
(b) Depth = 3.

(c) Depth = 4.

(d) Depth = 5.

Figure 12: Ablation study of the impact of the depth of the temperature hypernetwork on the Pareto fronts obtained by the HyperDPO framework on the MSLR-WEB10K dataset (Objective IV vs Objective V). The colorbar denotes the NDCG@10 of the main objective.

Given the choices of the temperature parameters, the Pareto fronts in both Figure 10a and 10b should merge into one single point, which refers to the solution of the single-objective fine-tuning task with



(a) $\mathcal{D}(\beta) = \mathrm{Unif}([0.83, 1.2]^2)$.

(b) $\mathcal{D}(\beta) = \mathrm{Unif}([0.71, 1.4]^2)$.

(c) $\mathcal{D}(\beta) = \mathrm{Unif}([0.63, 1.6]^2)$.

Figure 13: Ablation study of the impact of the distribution $\mathcal{D}(\beta)$ on the Pareto fronts obtained by the HyperDPO framework with the temperature hypernetwork on the MSLR-WEB10K dataset (Objective IV vs Objective V). The colorbar denotes the NDCG@10 of the main objective.

certain temperature parameter $\beta$. Although the results are roughly in accordance with the theoretical expectations, there are still small gaps that may be accounted for by the limit of the expressive power of the hypernetwork structure and insufficient exploration over the weight vector $\boldsymbol{w}$.

To explain this, we present ablation studies to investigate the effect of the expressiveness of the hypernetwork structure on the performance of the HyperDPO framework with the temperature hypernetwork. We applied hypernetworks with 2 to 5 layers of transformer architecture to the temperature hypernetwork, and the results show that the performance, indicated by the expected behaviors of the Pareto front, is drastically improved with the increase of the number of layers of the hypernetwork. While swallower hypernetworks yield Pareto fronts with less expected behaviors and more noise, *e.g.* the concavity of the Pareto fronts in Figure 12b partially indicates the insufficiency of the training of the temperature hypernetwork, the temperature hypernetwork with 5 layers of transformer architecture in Figure 12d exhibits improved scores and more expected behaviors according to different temperature configurations. This suggests and confirms the intuition that temperature hypernetworks require more expressive structures to capture the complex trade-offs between the main and auxiliary objectives.

The choice of the distribution $\mathcal{D}(\boldsymbol{\beta})$ also affects the performance of the temperature hypernetwork. Figure 13 shows the ablation study of the impact of the distribution $\mathcal{D}(\boldsymbol{\beta})$ on the Pareto fronts obtained by the HyperDPO framework with the temperature hypernetwork on the MSLR-WEB10K dataset. The results suggest that the distribution $\mathcal{D}(\boldsymbol{\beta})$ should cover a larger range than those interested in the temperature hypernetwork to ensure sufficient training.

Given the preliminary results and ablation studies, we conclude that despite requiring more expressive structures and more training resources, the temperature hypernetwork is a feasible and promising direction for the HyperDPO framework to achieve more flexible control over the Pareto front and we expect to further investigate the temperature hypernetwork in future work.