No More Tuning: Prioritized Multi-Task Learning with Lagrangian Differential Multiplier Methods

Zhengxing Cheng[1,2*], **Yuheng Huang**[2], **Zhixuan Zhang**[2], **Dan Ou**[2], **Qingwen Liu**[2]

[1]University of California, Berkeley
[2]Alibaba Group
zhengxing_cheng@berkeley.edu, {hexing.hyh,zhibing.zzx,oudan.od,xiangsheng.lqw}@alibaba-inc.com

Abstract

Given the ubiquity of multi-task in practical systems, Multi-Task Learning (MTL) has found widespread application across diverse domains. In real-world scenarios, these tasks often have different priorities. For instance, In web search, relevance is often prioritized over other metrics, such as click-through rates or user engagement. Existing frameworks pay insufficient attention to the prioritization among different tasks, which typically adjust task-specific loss function weights to differentiate task priorities. However, this approach encounters challenges as the number of tasks grows, leading to exponential increases in hyper-parameter tuning complexity. Furthermore, the simultaneous optimization of multiple objectives can negatively impact the performance of high-priority tasks due to interference from lower-priority tasks.

In this paper, we introduce a novel multi-task learning framework employing Lagrangian Differential Multiplier Methods for step-wise multi-task optimization. It is designed to boost the performance of high-priority tasks without interference from other tasks. Its primary advantage lies in its ability to automatically optimize multiple objectives without requiring balancing hyper-parameters for different tasks, thereby eliminating the need for manual tuning. Additionally, we provide theoretical analysis demonstrating that our method ensures optimization guarantees, enhancing the reliability of the process. We demonstrate its effectiveness through experiments on multiple public datasets and its application in Taobao search, a large-scale industrial search ranking system, resulting in significant improvements across various business metrics.

Extended version — https://arxiv.org/abs/2412.12092

Introduction

In the era of data-driven decision-making, Multi-Task Learning (MTL) has become an essential paradigm, offering notable advantages in managing multiple objectives concurrently (Chen, Zhang, and Yang 2024; Wang et al. 2023). This is especially relevant in practical systems where multiple objectives must be optimized simultaneously, even though these objectives may sometimes negatively influence each other.

A illustrative example is in the domain of web search, where relevance is typically prioritized over other metrics such as click-through rates (CTR) or user engagement. In a web search system, relevance refers to how well the returned results match the user's query, directly impacting user satisfaction and search quality. While metrics like CTR and engagement are also important, prioritizing relevance ensures that users find the most useful and accurate information first. However, optimizing for secondary objectives like CTR can negatively affect the primary goal of relevance. For example, to increase CTR, a system might prioritize results that are more attention-grabbing but less relevant to the query, thereby reducing the overall relevance of the search results. This trade-off underscores the need for a framework that can effectively balance these tasks without sacrificing the primary goal of relevance.

Despite its widespread adoption, traditional MTL frameworks often encounter challenges in task prioritization, which is typically addressed by adjusting task-specific loss function weights. However, the complexity of hyper-parameter tuning escalates exponentially as the number of tasks increases, especially when using grid search methods. In real-world industrial applications, this complexity is exacerbated by the necessity for online A/B testing to validate the final model effectiveness. A/B testing requires sustained online observation to gather feedback, rendering this exponential complexity impractical in industrial scenarios due to the extended duration and high resource consumption.

To tackle these challenges, we introduce a novel multi-objective optimization framework called No More Tuning (NMT). This framework manages task prioritization in multi-task learning (MTL) by ensuring that secondary tasks are optimized without compromising the performance of the primary task. This is achieved by framing the MTL problem as a constrained optimization problem, where the primary task's performance is maintained as a inequality constraint during the optimization of secondary tasks.

To solve this constrained problem, we use the method of Lagrange multipliers, which allows us to convert it into an unconstrained problem.To accommodate the gradient descent optimization methods widely used in most MTL approaches, we discuss solving the Lagrangian function using

*This work was completed during an internship at Alibaba Group.

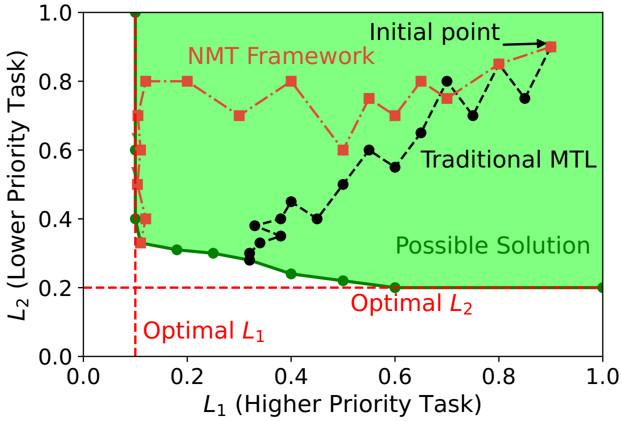

Figure 1: Optimization trajectories for two strategies. The traditional approach compromises between tasks, often sub-optimally affecting the primary objective. The NMT framework first optimizes the primary task, then refines secondary tasks while maintaining the primary task's performance.

gradient descent. Consequently, the NMT framework can seamlessly integrate with any MTL method that uses gradient descent optimization algorithms. As a result, NMT can be integrated with any gradient descent-based MTL method. Notably, since the task prioritization is incorporated into the inequality constraints, no additional hyper-parameters are required in NMT framework.

Figure 1 gives a visual representation of the optimization process with the NMT framework. The figure shows the trajectories of two different optimization strategies, highlighting how the NMT framework effectively balances multiple objectives. The traditional approach attempts to find a compromise between tasks, often resulting in sub-optimal performance for the primary objective. In contrast, the trajectory for the NMT framework demonstrates a distinct path through the optimization space. Initially, this path focuses solely on optimizing the primary task. Once the primary task achieves its optimal performance, the framework then optimizes secondary tasks while maintaining the performance of the primary task. This trajectory visually emphasizes how the NMT framework ensures that the primary objective remains uncompromised while improving secondary tasks.

The key advantages of our method include:

No Need for Parameter Adjustments: The NMT framework embeds task prioritization directly into the equality constraints of the optimization problem. This eliminates the need for manual parameter tuning or adjustments, which are often necessary in traditional MTL methods to balance different tasks. By removing this requirement, our framework significantly reduces the complexity and time involvxd in the model training process. This also reduces the risk of sub-optimal performance due to improper parameter settings.

Theoretical Background: A key advantage of the NMT framework is its ability to provide theoretical background regarding the performance of high-priority tasks. We give the theoretical proof in section Theoretical Analysis. By setting the primary task's performance as a strict inequality constraint, the framework ensures that the optimization direction adheres to these constraints. This contrasts with traditional methods that might require extensive empirical validation to ensure that primary tasks are not adversely affected.

Easy Integration: The NMT framework can be easily integrated with all gradient descent-based MTL methods, offering a simple yet effective solution for task prioritization.

Broad Effectiveness: We conducted experiments on multiple multi-task public datasets, to validate the effectiveness of our method. Furthermore, the NMT framework has been implemented in Taobao search, a large-scale industrial online systems with billions of active users, resulting in significant gains in online A/B testing. This demonstrates the framework's versatility and effectiveness across different domains and applications.

Related Works

Overview of MTL

Multi-Task Learning (MTL) has become increasingly prevalent in real world applications, particularly in recommendation and search engines where multiple objectives should be considered. Traditional methods that optimizing separate task-specific loss functions can result in conflicts and reduced performance across tasks (Caruana 1997). To overcome these challenges, various research has been conducted to balance the trade-offs between tasks.

In Multi-task learning, one active research area is parameter-sharing architecture. We provide an overview of the evolution of MTL architectures, which can be categorized into four principal types: Tower-level task-specific models (Caruana 1997; Misra et al. 2016), Gate-level task-specific models(Hazimeh et al. 2021), Expert-level task-specific models(Tang et al. 2020) and Embedding-level task-specific models (Su et al. 2024). The parameter-sharing framework enables multiple tasks to utilize a common set of experts, with task-specific gating networks directing the flow of information.

Various MTL optimization methods have been explored to address negative transfer between tasks. A prevalent approach is task weighting, which balances multiple tasks by combining loss objectives with specific weights, typically determined through extensive hyper-parameter tuning (Kokkinos 2016; Sermanet et al. 2014). Dynamic gradient-based methods (Chen et al. 2018; Yu et al. 2020; Javaloy and Valera 2022) have been developed to manage task weight balancing by employing techniques such as gradient normalization and adjustment to reduce disparities between tasks. Pareto optimization framework(Lin et al. 2019) has also been used for coordinate these objectives with a weighted aggregation. However, these methods can not specifically prioritize one task over others in the optimization process.

Recent studies have explored more principled approaches to MTL, such as the framework developed by (Mahapatra et al. 2022), which integrates first-order gradient-based algorithms into the ranking process. Further advancements in MTL, like those introduced by (Carmel et al. 2020), involve training separate models on distinct objectives and then aggregating their scores with stochastic label aggregation. Additionally, distillation-based ranking solutions (Tang et al. 2024; Woo, Lee, and Cho 2021) use the distilled models to balance the optimization of multiple tasks in the learning-to-rank setting. However, these approaches all still rely on parameter tuning to balance task performance and fail to explicitly guarantee the prioritization of critical tasks. When implementating in the real-world system, the practical challenges including tuning and computational complexity persist.

In contrast to these techniques, our optimization framework offers a novel approach that simplifies the optimization process and enhances the performance of high-priority tasks without requiring extensive manual tuning. It is straightforward to implement and applicable across various MTL architectures.

Constrained Optimization and Application

Constrained optimization is a classical optimization problem, with Lagrangian methods being among the most well-established approaches (Platt and Barr 1987). Constrained optimization has been widely applied in reinforcement learning (RL), particularly in safe RL tasks (Stooke, Achiam, and Abbeel 2020). By using Lagrangian methods, the original constrained problem is transformed into an unconstrained form, facilitating optimization. Theoretical convergence proofs and methods for updating Lagrange multipliers have been developed to support this process(Paternain et al. 2019; Tessler, Mankowitz, and Mannor 2018).

Inspired by the successful application of constrained optimization as well as the Lagrangian methods in RL, our work aims to adapt these techniques to Multi-Task Learning (MTL) problems. We propose reformulating the prioritized multi-objective optimization problem as a multi-step constrained optimization problem, utilizing the principles of constrained optimization to tackle challenges in MTL.

Problem Formulation

Without loss of generality, consider a scenario with two tasks, where Task 1 has higher priority than Task 2. Task 1 and Task 2 are represented by loss functions $f_1(\theta)$ and $f_2(\theta)$, respectively. The optimization problem can be formulated as:

$$\min_{\theta} f_1(\theta), f_2(\theta) \tag{1}$$

Our objective is as follows:

- Primary Task (Higher Priority): Minimize $f_1(\theta)$
- Secondary Task (Lower Priority): Minimize $f_2(\theta)$ while ensuring that the performance of $f_1(\theta)$ is not compromised.

The question can be extended to any number of tasks. Consider a scenario with n tasks, where each task i has an associated loss function $f_i(\theta)$ and a priority level p_i. The priority levels are such that p_1 is the highest priority and p_n is the lowest, with p_i being monotonic (i.e., $p_1 > p_2 > \cdots > p_n$). The optimization problem can be formulated as:

$$\min_{\theta} f_1(\theta), f_2(\theta), \ldots, f_n(\theta) \tag{2}$$

The requirement is to optimize each subsequent tasks $f_i(\theta)$ for $i > 1$ while ensuring that the performance of higher priority tasks $f_j(\theta)$ for $j < i$ is maintained.

Proposed No More Tuning (NMT) Optimization Framework

Optimization Under Two Tasks

Given that $f_1(\theta)$ is prioritized, we need $f_2(\theta)$ to be optimized while keeping $f_1(\theta)$ at its optimal value $f_1(\theta^*)$, found from:

$$\theta^* = \arg\min_{\theta} f_1(\theta) \tag{3}$$

The optimization problem for $f_2(\theta)$ is then:

$$\begin{aligned} \min_{\theta} \quad & f_2(\theta) \\ \text{s.t.} \quad & f_1(\theta) \le f_1(\theta^*) \end{aligned} \tag{4}$$

Solving this problem ensures that the performance of the high-priority task remains unaffected while optimizing the low-priority task. The NMT framework aims to address this optimization problem by employing gradient descent.

Extension to Arbitrary m Tasks

The above framework can naturally be extended to any number of tasks, represented by loss functions $f_1(\theta), f_2(\theta), \ldots, f_m(\theta)$, with Task 1 having the highest priority and Task m the lowest. The problem can be formulated as:

$$\min_{\theta} f_1(\theta), f_2(\theta), \ldots, f_m(\theta) \tag{5}$$

The optimization steps are:

1. Minimize $f_1(\theta)$ (Highest Priority) to obtain θ_1^* and $f_1(\theta_1^*)$.
2. Minimize $f_2(\theta)$ (Secondary Priority) subject to $f_1(\theta) \le f_1(\theta_1^*)$, to obtain θ_2^*.
3. Minimize $f_3(\theta)$ (Tertiary Priority) subject to $f_1(\theta) \le f_1(\theta_2^*)$ and $f_2(\theta) \le f_2(\theta_2^*)$, to obtain θ_3^*.
4. Continue this process for subsequent tasks.

For each task $f_i(\theta)$, the problem is formulated as:

$$\begin{aligned} \min_{\theta} \quad & f_i(\theta) \\ \text{s.t.} \quad & f_1(\theta) \le f_1(\theta_{i-1}^*) \\ & f_2(\theta) \le f_2(\theta_{i-1}^*) \\ & \vdots \\ & f_{i-1}(\theta) \le f_{i-1}(\theta_{i-1}^*) \end{aligned} \tag{6}$$

This approach ensures that each task $f_i(\theta)$ is optimized in sequence until the m_{th} task while preserving the optimal values of all higher-priority tasks.

Algorithm 1: NMT Algorithm for m Tasks

Input:

- η: Learning rate for model parameters
- τ: Learning rate for Lagrange multipliers
- $f_k(\theta)$: Objective function for task k ($k \in \{1, 2, \ldots, m\}$)
- λ_{init}: Initial value for Lagrange multipliers

```
Step 1: Optimize the primary task (task 1) to minimize f_1(θ) until convergence. Save the optimized parameters as θ*.
for each task k from 2 to m do
    Initialize θ with θ* and λ_j with λ_init for j = 1, ..., k − 1.
    repeat
        Compute the aggregate loss for task k:
            L = f_k(θ) + Σ_{j=1}^{k−1} λ_j · (f_j(θ) − f_j(θ*))
        Update θ using gradient descent:
            θ ← θ − η · ∇_θ L
        Update each λ_j using gradient ascent:
            λ_j ← λ_j + τ · (f_j(θ) − f_j(θ*))
    until convergence
    Update θ* with the optimized θ
end for
```

No More Tuning (NMT) Algorithm

The overall No More Tuning (NMT) optimization framework for m tasks is illustrated in Algorithm 1. We convert the inequality constraint problem (6) into an unconstrained optimization problem using the Lagrange multiplier method and solve it with a gradient-based approach.

The algorithm is divided into two main stages:

1. **Optimize the main task**: First, we perform optimization for the first task (the main task) to find the parameters θ^* that minimize the objective function $f_1(\theta)$. This stage lasts until the objective function converges.
2. **Iteratively optimize the remaining tasks**: After the optimization of the main task, we optimize each of the remaining tasks in turn. For each task k (starting from the second task), we initialize the model parameters to θ^* and set the Lagrange multiplier to the initial value λ_{init}. Then, the model parameters are updated using the gradient descent algorithm, while the Lagrange multiplier is updated using the gradient ascent algorithm. This process continues until convergence. After each iteration, we update the current optimized parameters to θ^* in preparation for the optimization of the next task.

The NMT framework is very easy to implement because it only requires gradient ascent of the Lagrange multiplier λ based on the optimization of the secondary task. No additional hyper-parameters are introduced in the whole framework.

Re-Scaling Method When implementing the algorithm, we find that when λ is large, the loss function can become excessively large, leading to substantial updates in the parameters θ during gradient descent. These large updates can destabilize the learning process and violate the assumption outlined in the next chapter, where we aim for small changes in θ. Thus, we apply a re-scaled loss function when updating θ. When performing gradient descent on θ,

$$\mathcal{L} = \frac{1}{1 + \sum_{j=1}^{k-1} \lambda_j} \left(f_k(\theta) + \sum_{j=1}^{k-1} \lambda_j \left(f_j(\theta) - f_j(\theta_j^\star) \right) \right)$$

This re-scaling formulation ensures the combination of each loss is a normalized convex combination, preventing any potential explosion of the loss function. All our experiments incorporate this re-scaling formulation for the loss function during the optimization process.

Theoretical Analysis

We now consider a relaxed version of the problem that allows for some tolerance in the constraints. Specifically, we examine the constrained optimization problem (CO) with m objectives, where the i_{th} objective is allowed to perform at most r_i worse than $f_i(\theta^*)$:

$$\begin{aligned} \min_{\theta} \quad & f_m(\theta) \\ \text{s.t.} \quad & f_i(\theta)) \leq f_i(\theta^*)) + r_i, i = 1, ...m-1, \end{aligned} \tag{CO}$$

where f_i is the i_{th} objective function and r_i is a small positive tolerance term for the i_{th} objective.

From the constrained problem, we can define the Lagrangian function as:

$$L(\theta, \lambda) = f_m(\theta) + \sum_{i=1}^{m-1} \lambda_i \left(f_i(\theta) - f_i(\theta^*) - r_i \right). \tag{7}$$

Then the unconstrained dual optimization problem (DO) can then be formulated as:

$$\max_{\lambda} \min_{\theta} L(\theta, \lambda) \tag{DO}$$

If f_i is a convex function (e.g. in logistic regression), the strong duality between CO and DO holds (Platt and Barr 1987) and the convergence of the NMT optimization is guaranteed .

However, in many practical scenarios, the objective functions are not convex. We aim to show that, under appropriate assumptions, strong duality still holds despite non-convexity, guiding our optimization method.

Assumption 1. *The model training is free from over-fitting.*

The first assumption ensures that the optimal solution corresponds to the minimum of the objective function. It guarantees monotonicity between the objective function and the target performance.

Assumption 2. *Within the feasible region satisfying the constraints, the difference of θ is bounded by ϵ, and f_i is Lipschitz continuous.*

This assumption specifies properties of the objective function useful for analyzing duality.

We also introduce the definition of the perturbation function associated with CO for later proofs.

$$P(\xi) \triangleq \min_{\theta} f_m(\theta) \quad \text{subject to } f_i(\theta) \leq f_i(\theta^*) + r_i - \xi_i, i = 1 \ldots m-1 \tag{8}$$

We now demonstrate two properties under the given assumptions:

Proposition 1. *Slater's condition holds for CO*

Proof. As θ^* is a feasible solution and r_i is positive, $f_i(\theta^*) \leq f_i(\theta^*) + r_i$ satisfies trivially. □

Proposition 2. *Under assumption 2, perturbation function $P(\xi)$ is approximately convex when ϵ is small enough.*

The detailed proof of Proposition 2 will be provided in the Appendix of the extended version, which has been published on arXiv. Denote $\theta^*(\xi)$ as optimal θ for given specific ξ. We derive that when $\|\theta^*(\xi_2) - \theta^*(\xi_1)\|$ which is bounded by the ϵ is sufficiently close to 0, the convexity holds.

Theorem 1. *Under Assumption 1,2, when ϵ is small enough, the strong duality holds for CO and DO.*

Proof. The strong duality holds when two conditions are satisfied Slater's condition and the convexity of the perturbation function(Rockafellar 1970). We have demonstrated both conditions in the preceding propositions. □

Theorem 1 indicates that if the change in θ^* is sufficiently small, the optimal solution of DO will also be the optimal solution of CO. Then we can get a feasible solution by optimizing the unconstrained dual problem DO which is more convenient. The NMT framework aims to find the local minimum θ for f_m that satisfies the constraints, ensuring minimal deviation from previously identified optimal θ^*. The min-max optimization requirement suggests a two-step approach: gradient descent is performed on θ to minimize f_i, while gradient ascent is applied to λ to satisfy the maximization requirement. Consequently, our NMT optimization framework, which performs gradient descent on θ and gradient ascent on λ, is expected to converge to an optimal solution.

Experiments

Experimental Results on Public Datasets

Experimental Setup In our experiments, we selected two public MTL recommendation datasets, namely TikTok and QK-Video (Yuan et al. 2023) for performance evaluation. The TikTok dataset includes two objectives: Finish and Like, while the QK-Video dataset contains two objectives: Click and Like. In our experiments, we prioritize the Like objective as the primary task, with Finish and Click serving as the respective secondary tasks in the two datasets.

NMT is designed to be fully compatible with most existing MTL approaches. Rather than directly competing with these methods, it serves as a complementary framework that enhances their performance by emphasizing task prioritization. In this section, we integrate the NMT framework with parameter-sharing MTL architectures, including Shared-Bottom (Caruana 1997), OMoE (Ma et al. 2018), MMoE (Ma et al. 2018), and PLE (Tang et al. 2020). Additionally, we apply the NMT framework to a gradient-based MTL method (Liu et al. 2024), with implementation details provided in the appendix.

Performance Evaluation As shown in Table 1, NMT led to significant improvements in Like AUC across all models in the TikTok dataset. This demonstrates that NMT effectively enhances the performance of the high-priority task. Meanwhile, the other metrics, such as Finish AUC and Average AUC, also saw modest improvements, indicating that the performance of these secondary tasks remains stable and does not deteriorate when prioritizing the Like task. The PLE model, particularly when paired with NMT, stands out with the highest AUC across all metrics.

Table 2 shows the performance on the QK-Video dataset. Similar to the TikTok dataset, Like AUC increased across all models with NMT While there were slight decreases in Click AUC for some models, these reductions were minimal, indicating that the prioritization of the Like task does not severely impact the performance of other metrics. The Average AUC remained largely stable, demonstrating that the overall model performance is well-balanced and not significantly compromised by the prioritization of the high-priority Like task.

Overall, these results demonstrate that NMT effectively prioritizes the high-priority task, improving its performance while maintaining satisfactory levels of performance in other metrics. In some conditions, it can even lead to simultaneous improvements for all the tasks, demonstrating the possibility of the NMT in enhancing overall task performance without compromising on any specific objective.

We also compares NMT with traditional weights adjustment methods. The weights adjustment methods directly sums the loss functions of the two tasks and manually adjusts the coefficients of the loss functions respectively:

$$L = \alpha_1 L_1 + \alpha_2 L_2 \tag{9}$$

where α_1 and α_2 are tuned manually.

In order to fully demonstrate all the possibilities of the weights adjustment method, we adjusted as many hyperparameter combinations as possible. Figure 2 show the AUC performance on the TikTok dataset with different combination of loss weights (from 0.1:0.9 to 0.5:0.5). The dotted lines represent the trade-offs between the two metrics as the weights shift. The star markers indicate the performance of models optimized by NMT. Notably, these NMT-optimized models significantly outperform their standard counterparts, achieving results far beyond the capabilities of the traditional loss weight adjustments. The NMT optimization framework consistently pushing the boundaries of individual tasks' performance which highlight the effectiveness of the NMT framework in optimizing both high-priority and secondary tasks simultaneously.

Model	Without NMT			With NMT		
	Finish AUC	Like AUC	Average AUC	Finish AUC	Like AUC	Average AUC
Shared-Bottom	0.7504	0.9031	0.8267	0.7510 (+0.06%)	0.9069 (**+0.38%**)	0.8289 (+0.22%)
OMoE	0.7505	0.9021	0.8263	0.7509 (+0.04%)	0.9059 (**+0.38%**)	0.8284 (+0.21%)
MMoE	0.7503	0.9015	0.8259	0.7506 (+0.03%)	0.9056 (**+0.41%**)	0.8281 (+0.22%)
PLE	0.7506	0.9027	0.8266	0.7511 (+0.05%)	0.9076 (**+0.49%**)	0.8293 (+0.27%)

Table 1: Overall performance on TikTok.

Model	Without NMT			With NMT		
	Click AUC	Like AUC	Average AUC	Click AUC	Like AUC	Average AUC
Shared-Bottom	0.9128	0.9409	0.9268	0.9126 (-0.02%)	0.9417 (**+0.07%**)	0.9271 (+0.03%)
OMoE	0.9125	0.9414	0.9270	0.9119 (-0.07%)	0.9424 (**+0.1%**)	0.9272 (+0.02%)
MMoE	0.9126	0.9412	0.9269	0.9125 (-0.01%)	0.9417 (**+0.05%**)	0.9271 (+0.02%)
PLE	0.9126	0.9422	0.9274	0.9122 (-0.04%)	0.9425 (**+0.03%**)	0.9273 (-0.01%)

Table 2: Overall performance on QK-Video

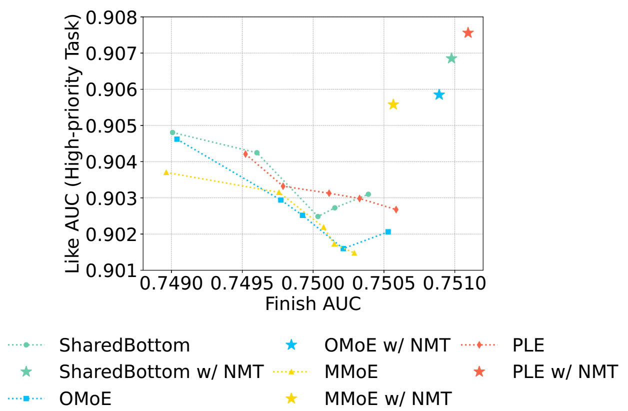

Figure 2: AUC performance comparison for different model configurations across Like and Finish tasks. The colored lines are the performance of different models under different adjusted weights in loss function, and the star markers with the same color are the performance of respective NMT optimized model. The NMT-enhanced models demonstrate a significant improvement over their standard counterparts.

Online Experimental Results

Experimental Setup In our experiments, we aim to balance multiple business objectives within Taobao search, a large-scale online e-commerce search system that serves billions of active users. Our objectives are order volume, GMV (Gross Merchandise Value), and relevance, with order volume being the highest priority, followed by GMV, and relevance being the lowest priority. Order volume represents the number of successful transactions, GMV denotes the total monetary value of these transactions, and relevance assesses how well the search results align with user intent.

In contrast to public datasets commonly used in MTL research, where models typically output multiple scores to address different tasks, online ranking systems must provide a single score for final ranking. This requirement intensifies the conflict between tasks, as they must be integrated into a unified output score.

We utilize a standard Learning to Rank (LTR) paradigm with a deep neural network (DNN) architecture and the pairwise loss function (Cao et al. 2007). For the order volume task (*pay*), we generate sample pairs by comparing positive samples (transactions with purchases) to negative samples (non-purchase interactions). For the GMV task (*amount*), pairs are created by contrasting high-value transactions with lower-value transactions. For the relevance task (*relevance*), pairs consist of relevant items versus non-relevant items, where relevant items are considered positive and non-relevant items are considered negative. The pairwise loss is formulated as:

$$L = -\log\left(\text{sigmoid}(z_{pos} - z_{neg})\right) \tag{10}$$

where z is the output logit of the DNN.

In our experiment, we consider *pay* as the first priority task, *amount* as the second priority task, and *relevance* as the third priority task. We use a single-task model with only the *pay* task as the baseline.

Performance Evaluation Table 3 shows the differences in business metrics between the multi-task ranking model and the baseline ranking model.

We assessed the performance of integrating multiple tasks into a unified model using our proposed NMT framework. For the *pay* + *relevance* task setup, incorporating the relevance task resulted in a significant improvement in relevance metrics. Notably, the number of orders, reflecting the *pay* task, did not decrease but showed a slight increase, suggesting that the prioritization of the *pay* task was maintained while improving relevance.

In contrast, when we applied a weight adjustment method to balance task weights, we observed a clear enhancement

Task	Order Volume	GMV	Relevance
pay + *relevance*	+0.26%	+0.15%	+0.72%
pay + *relevance* (weight adjustment)	-0.35%	-0.13%	+0.42%
pay + *amount*	-0.05%	+0.57%	-0.24%
pay + *amount* + *relevance*	-0.04%	+0.49%	+0.51%

Table 3: Business Metrics of A/B Tests. Weight adjustment balances task weight manually. Experiments without annotations use the NMT framework to optimize multiple tasks. Baseline is a ranking model with only *pay* task.

in relevance metrics. However, this came at the cost of a notable decline in the number of orders. This decline can be attributed to the simple weighted summation of loss functions, which adversely impacted the primary *pay* objective due to conflicting influences from the additional tasks.

The experiments with the *pay* + *amount* task demonstrated similar findings. We achieved a significant improvement in the *amount* task while preserving the metrics for the *pay* task. This confirms that our approach effectively maintains the performance of higher-priority tasks while enhancing secondary objectives.

To further validate the effectiveness of the NMT framework across multiple tasks, we extended our analysis to the *pay* + *amount* + *relevance* setup. The results indicate that even when optimizing for all three tasks simultaneously, the NMT framework successfully preserved the performance of the highest-priority task (*pay*) without any loss. Simultaneously, there was noticeable improvement in the metrics for the *amount* and *relevance* tasks.

Optimization Process Visualization To further illustrate the optimization process of the NMT framework, we analyze the *pay* + *relevance* task. In NMT framework, the optimization object of constrained *relevance* task can be written as:

$$\begin{aligned} \min_{\theta} \quad & L(\theta) \\ \text{where} \quad & L(\theta) = L_{rel}(\theta) + \lambda(L_{pay}(\theta) - L_{pay}(\theta^*)) \end{aligned} \tag{11}$$

The optimization process is depicted in Fig.3. Initially, in Fig.3.a, the constraint coefficient λ is set to 0, leading to a gradual increase in its value during the early stages of training. This increase indicates that while optimizing for the *relevance* task alone, the *pay* task struggles to maintain its optimal performance. However, as λ rises, the *pay* task loss stabilizes near its optimal value, with minor fluctuations. Throughout training, as Fig.3.b shows, the *pay* task loss remains close to its optimal point, demonstrating that the NMT framework effectively prioritizes the *pay* task. Meanwhile, as we can see in Fig.3.c, the loss of the *relevance* task gradually decreases, indicating steady *relevance* improvement as the training progresses. As λ gradually converges, the model finds a balance between the *pay* and *relevance* tasks, ensuring that the *pay* task achieves optimal performance while also enhancing the *relevance* task.

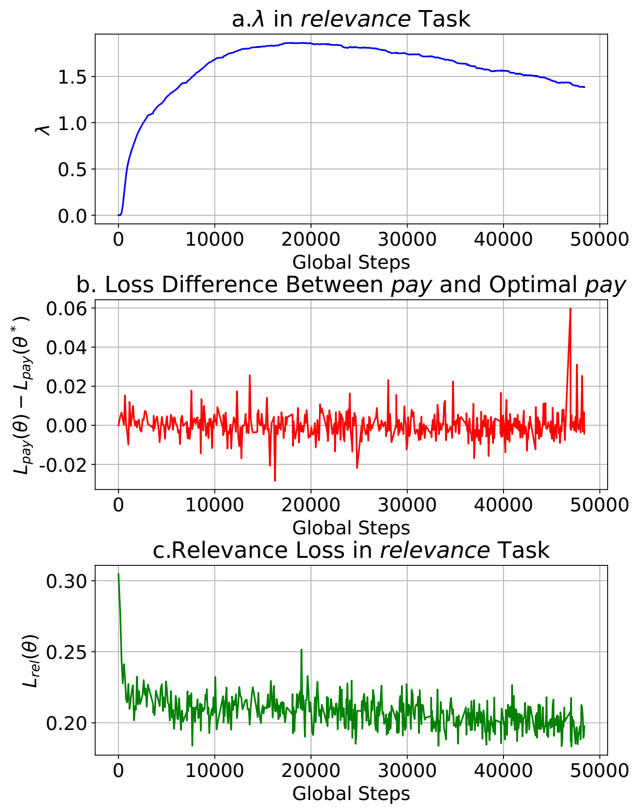

Figure 3: Training metrics of secondary task *relevance* of *pay* (Primary) + *relevance* (Secondary). The top line shows the λ during training. The middle line illustrates the fluctuation of the *pay* loss around its optimal value. The bottom line displays the loss function of *relevance* during the training.

Overall, the experiments underscore the efficacy of the NMT framework in emphasizing primary objectives. By prioritizing the primary task while achieving substantial improvements in secondary tasks, the framework demonstrates robust performance in ranking scenarios in industrial systems, reflecting its practical utility in real-world.

Conclusion

No More Tuning (NMT) aims to solve a problem of multi-task learning with different priorities that has long been ignored by the community. NMT addressed the challenges of task prioritization and tedious hyper-parameter tuning. Through a constrained optimization, NMT ensures that primary tasks are optimized without compromising their performance while still refining secondary tasks. This method eliminates the need for hyper-parameter tunings, provides theoretical analysis for task performance, and offers easy integration with existing MTL methods. The broad applicability and proven effectiveness of NMT across various domains, from public datasets to industrial systems.

References

Cao, Z.; Qin, T.; Liu, T.-Y.; Tsai, M.-F.; and Li, H. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, 129–136. New York, NY, USA: Association for Computing Machinery. ISBN 9781595937933.

Carmel, D.; Haramaty, E.; Lazerson, A.; and Lewin-Eytan, L. 2020. Multi-Objective Ranking Optimization for Product Search Using Stochastic Label Aggregation. In *Proceedings of The Web Conference 2020*, WWW '20, 373–383. New York, NY, USA: Association for Computing Machinery. ISBN 9781450370233.

Caruana, R. 1997. Multitask Learning. *Machine Learning*, 28: 41–75.

Chen, S.; Zhang, Y.; and Yang, Q. 2024. Multi-task learning in natural language processing: An overview. *ACM Computing Surveys*, 56(12): 1–32.

Chen, Z.; Badrinarayanan, V.; Lee, C.-Y.; and Rabinovich, A. 2018. GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks. arXiv:1711.02257.

Hazimeh, H.; Zhao, Z.; Chowdhery, A.; Sathiamoorthy, M.; Chen, Y.; Mazumder, R.; Hong, L.; and Chi, E. H. 2021. DSelect-k: Differentiable Selection in the Mixture of Experts with Applications to Multi-Task Learning. arXiv:2106.03760.

Javaloy, A.; and Valera, I. 2022. RotoGrad: Gradient Homogenization in Multitask Learning. arXiv:2103.02631.

Kokkinos, I. 2016. UberNet: Training a 'Universal' Convolutional Neural Network for Low-, Mid-, and High-Level Vision using Diverse Datasets and Limited Memory. arXiv:1609.02132.

Lin, X.; Chen, H.; Pei, C.; Sun, F.; Xiao, X.; Sun, H.; Zhang, Y.; Ou, W.; and Jiang, P. 2019. A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, 20–28. New York, NY, USA: Association for Computing Machinery. ISBN 9781450362436.

Liu, B.; Feng, Y.; Stone, P.; and Liu, Q. 2024. Famo: Fast adaptive multitask optimization. *Advances in Neural Information Processing Systems*, 36.

Ma, J.; Zhao, Z.; Yi, X.; Chen, J.; Hong, L.; and Chi, E. H. 2018. Modeling Task Relationships in Multi-task Learning with Multi-gate Mixture-of-Experts. KDD '18, 1930–1939. New York, NY, USA: Association for Computing Machinery. ISBN 9781450355520.

Mahapatra, D.; Dong, C.; Chen, Y.; Meng, D.; and Momma, M. 2022. Multi-Label Learning to Rank through Multi-Objective Optimization. arXiv:2207.03060.

Misra, I.; Shrivastava, A.; Gupta, A.; and Hebert, M. 2016. Cross-stitch Networks for Multi-task Learning. arXiv:1604.03539.

Paternain, S.; Chamon, L. F. O.; Calvo-Fullana, M.; and Ribeiro, A. 2019. Constrained Reinforcement Learning Has Zero Duality Gap. arXiv:1910.13393.

Platt, J.; and Barr, A. 1987. Constrained Differential Optimization. In Anderson, D., ed., *Neural Information Processing Systems*, volume 0. American Institute of Physics.

Rockafellar, R. T. 1970. *Convex analysis*. Princeton Mathematical Series. Princeton, N. J.: Princeton University Press.

Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; and LeCun, Y. 2014. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. arXiv:1312.6229.

Stooke, A.; Achiam, J.; and Abbeel, P. 2020. Responsive Safety in Reinforcement Learning by PID Lagrangian Methods. arXiv:2007.03964.

Su, L.; Pan, J.; Wang, X.; Xiao, X.; Quan, S.; Chen, X.; and Jiang, J. 2024. STEM: Unleashing the Power of Embeddings for Multi-task Recommendation. arXiv:2308.13537.

Tang, H.; Liu, J.; Zhao, M.; and Gong, X. 2020. Progressive Layered Extraction (PLE): A Novel Multi-Task Learning (MTL) Model for Personalized Recommendations. In *Proceedings of the 14th ACM Conference on Recommender Systems*, RecSys '20, 269–278. New York, NY, USA: Association for Computing Machinery. ISBN 9781450375832.

Tang, J.; Gao, H.; He, L.; and Katariya, S. 2024. Multi-objective Learning to Rank by Model Distillation. *arXiv preprint arXiv:2407.07181*.

Tessler, C.; Mankowitz, D. J.; and Mannor, S. 2018. Reward Constrained Policy Optimization. arXiv:1805.11074.

Wang, Y.; Lam, H. T.; Wong, Y.; Liu, Z.; Zhao, X.; Wang, Y.; Chen, B.; Guo, H.; and Tang, R. 2023. Multi-task deep recommender systems: A survey. *arXiv preprint arXiv:2302.03525*.

Woo, H.; Lee, H.; and Cho, S. 2021. An Efficient Combinatorial Optimization Model Using Learning-to-Rank Distillation. arXiv:2201.00695.

Yu, T.; Kumar, S.; Gupta, A.; Levine, S.; Hausman, K.; and Finn, C. 2020. Gradient Surgery for Multi-Task Learning. arXiv:2001.06782.

Yuan, G.; Yuan, F.; Li, Y.; Kong, B.; Li, S.; Chen, L.; Yang, M.; Yu, C.; Hu, B.; Li, Z.; Xu, Y.; and Qie, X. 2023. Tenrec: A Large-scale Multipurpose Benchmark Dataset for Recommender Systems. arXiv:2210.10629.