STEM: Unleashing the Power of Embeddings for Multi-task Recommendation

Liangcai Su [*1], **Junwei Pan** [*2], **Ximei Wang**[2], **Xi Xiao**[1†], **Shijie Quan**[2], **Xihua Chen**[2], **Jie Jiang**[2],

[1]Shenzhen International Graduate School, Tsinghua University
[2]Tencent Inc.
sulc21@mails.tsinghua.edu.cn, xiaox@sz.tsinghua.edu.cn,
{jonaspan, messiwang, justinquan, tinychen, zeus}@tencent.com

Abstract

Multi-task learning (MTL) has gained significant popularity in recommendation systems as it enables the simultaneous optimization of multiple objectives. A key challenge in MTL is the occurrence of negative transfer, where the performance of certain tasks deteriorates due to conflicts between tasks. Existing research has explored negative transfer by treating all samples as a whole, overlooking the inherent complexities within them. To this end, we delve into the intricacies of samples by splitting them based on the relative amount of positive feedback among tasks. Surprisingly, negative transfer still occurs in existing MTL methods on samples that receive comparable feedback across tasks. It is worth noting that existing methods commonly employ a shared-embedding paradigm, and we hypothesize that their failure can be attributed to the limited capacity of modeling diverse user preferences across tasks using such universal embeddings.

In this paper, we introduce a novel paradigm called Shared and Task-specific EMbeddings (STEM) that aims to incorporate both shared and task-specific embeddings to effectively capture task-specific user preferences. Under this paradigm, we propose a simple model STEM-Net, which is equipped with shared and task-specific embedding tables, along with a customized gating network with stop-gradient operations to facilitate the learning of these embeddings. Remarkably, STEM-Net demonstrates exceptional performance on comparable samples, surpassing the Single-Task Like model and achieves positive transfer. Comprehensive evaluation on three public MTL recommendation datasets demonstrates that STEM-Net outperforms state-of-the-art models by a substantial margin, providing evidence of its effectiveness and superiority.

Introduction

Recently, multi-task learning has drawn great interest in recommender systems since they often require the optimization of multiple objectives (*e.g.*, `Like`, `Share`, `Finish`) simultaneously. Obviously, the effectiveness of multi-task recommendation (MTR) critically depends on leveraging knowledge from other tasks to *help* the learning of each task. However, prior works have identified the negative transfer (Torrey et al. 2010) and seesaw phenomenon (Tang et al. 2020), whereby multi-task learning models may not always outperform their corresponding single-task ones. To address negative transfer, MMoE (Ma et al. 2018a) and PLE (Tang et al. 2020) incorporate *task-specific* gate networks and experts, respectively. Additionally, techniques such as gradient clipping (Yu et al. 2020; Chen et al. 2018; He et al. 2022) and optimizer separation (Yang et al. 2023) have also been employed to alleviate gradient conflicts.

Note that, existing methods tend to treat all samples in a task as a whole, overlooking the inherent intricacies within them. For instance, in a popular TikTok dataset involving two tasks: `Like` and `Finish` with various positive signals, negative transfer may happen while learning both tasks together, since these tasks may have different user preferences. However, it remains unclear *where the negative transfer phenomenon occurs*. To address this concern, we split the test set of TikTok into three distinct zones based on the relative amount of feedback between them: *Finish-Overwhelming*, *Comparable*, and *Like-Overwhelming*. Since `Task Like` comprises significantly fewer positive samples, we then evaluate the performance of `Task Like` on these zones by comparing two popular MTL models, MMoE and PLE, with the single-task `Like` model. As shown in Figure 1, both MMoE and PLE demonstrate a notable improvement in per-

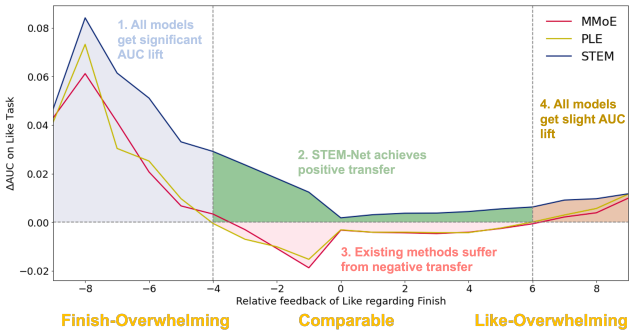


Figure 1: Our STEM-Net model addresses the negative transfers on task-competitive samples on TikTok dataset, where existing MTL models such as MMoE and PLE suffer. STEM-Net consistently outperforms MMoE (Ma et al. 2018a) and PLE (Tang et al. 2020) across both task-overwhelming and task-competitive samples.

[*]These authors contributed equally.
[†]Corresponding author.

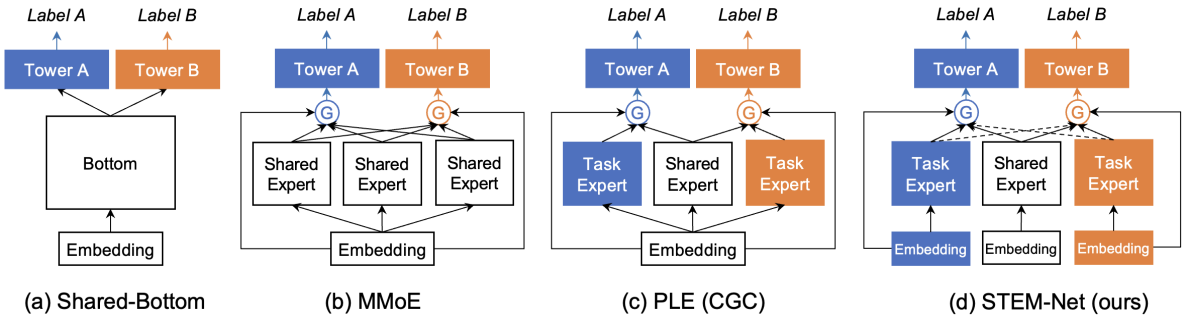


Figure 2: Comparison between representative MTL models and our proposed STEM-Net. Dot lines denote connections with stop-gradient operation.

formance on the Finish-Overwhelming zone where `Task Like` benefits from incorporating additional knowledge of `Task Finish`, as well as a slight performance boost on the Like-Overwhelming zone where `Task Like` itself can already learn well. However, to our surprise, in the zone that receives comparable positive feedback from both tasks, the performance of existing methods is *inferior* to that of the single-task model.

The performance drop of existing MTL methods on the comparative zone prompts us to reconsider the design of MTL recommendation models in order to address negative transfer effectively. Intuitively, users may possess diverse and sometimes even conflicting preferences for items across various tasks, which become more pronounced when there are sufficient signals from multiple tasks, as in the comparable zone here. Meanwhile, it is well believed that the preference of users is captured through user and item embeddings in MTL recommendation models. However, existing MTL methods, including MMoE and PLE, all follow a *shared-embedding paradigm*, that is, learning a universal embedding for each user and item feature, shared across tasks. Such a paradigm can only capture a single preference of users, hindering the ability to capture the diversity of preferences across tasks.

Motivated by the limitations of the shared-embedding paradigm, this paper introduces a Shared and Task-specific EMbeddings (STEM) paradigm. STEM aims to *incorporate both shared and task-specific embeddings to learn task-specific user preferences*. Under this paradigm, we design a simple model, namely STEM-Net, which begins by introducing the Shared and Task-specific Embedding layer, which performs a lookup operation for each feature ID to retrieve both shared and task-specific embeddings. This layer serves as the foundation for subsequent steps in the model. STEM-Net then constructs a set of experts based on these embeddings. Each expert is designed to be either shared, utilizing only shared embeddings, or task-specific, utilizing embeddings specific to a particular task. Furthermore, in order to facilitate comprehensive knowledge transfer among tasks, STEM-Net employs a customized gating mechanism, which enables each task tower to receive input from both shared experts and all task-specific experts. To ensure the learning of task-specific embeddings, the gating function deployed for each task tower employs a *stop-gradient operation* on the experts from other tasks. This operation prevents the gradients from flowing back to the experts of other tasks, pushing the task-specific tower to update its corresponding embeddings.

We conduct an extensive evaluation of the proposed STEM-Netmodel in the three zones of the TikTok dataset. Our results demonstrate that STEM-Net consistently outperforms both MMoE and PLE on the Finish and Like-Overwhelming samples. Moreover, STEM-Net exhibits remarkable performance in the comparable zone, surpassing the Single-Task Like model. *This achievement signifies the successful mitigation of negative transfer, marking a notable advancement in the field of MTL.* Furthermore, we assess STEM-Net on three public MTL recommendation datasets. In all cases, STEM-Net surpasses state-of-the-art models by a substantial margin, further validating its effectiveness and superiority. We outline our contributions as follows:

- We delve into the analysis of samples to investigate the negative transfer in MTL recommendation models. Surprisingly, our investigation reveals the presence of negative transfer in existing models when applied to samples that receive comparative feedback from multiple tasks.
- We propose a novel paradigm called STEM (Shared and Task-specific Embeddings) for a multi-task recommendation. Under this paradigm, a simple model STEM-Net is designed with shared and task-specific embeddings to effectively capture and preserve task-specific user preferences.
- We conduct comprehensive experiments and ablation studies on three MTL recommendation datasets. Through these experiments, we provide compelling evidence of the effectiveness of our proposed model. Additionally, we deploy the STEM-Net in an industrial advertising system for A/B testing, which yields a significant improvement in GMV (Gross Mechanise Value).

Related Work

Evolution of the MTL Model Architectures

In this section, we provide a comprehensive review of multi-task recommendation models that are based on the widely

adopted *Embedding-Expert-Gate-Tower* architecture, as referenced in previous works (Caruana 1997; Ma et al. 2018a; Tang et al. 2020; Yang et al. 2022; Hazimeh et al. 2021; Qin et al. 2020). This architecture comprises four key modules: *Embedding*, which employs dense vectors to represent sparse features; *Expert*, which extracts knowledge from the input features; *Gate*, which aggregates the outputs of the experts using attentive weights; and *Tower*, which generates the final output for each task. Figure 2 illustrates representative models at various levels of task specificity. Notably, recent works have witnessed a gradual shift in the placement of task-specific modules, with an increasing emphasis on capturing task-specific characteristics at the lower levels of the architecture.

Tower-level task-specific models, meaning that each task has an independent tower and is updated only by its own task loss. As shown in Figure 2(a), the input of each task-specific tower is shared across tasks, but the parameters of the towers are separated. Shared-Bottom (Caruana 1997) is representative of this type of models, also known as hard parameter sharing. OMoE (Ma et al. 2018a), Cross-stitch (Misra et al. 2016) are variants of Shared-Bottom with shared experts (*i.e*. bottom).

Gate-level task-specific models refers to multi-task models that assigns a gate network to each task in order to weight the outputs of experts. The classical representative of this type of model is MMoE as shown in Figure 2(b). Dselect-k (Hazimeh et al. 2021), MoSE (Qin et al. 2020) and MT-FwFM (Pan et al. 2019) also fit into this category.

In the **Expert-level task-specific models**, each task has its own experts on the basis of the Tower/Gate-level task-specific model. PLE (Tang et al. 2020) serves as a representative model for this approach, which utilizes both task-specific and shared experts to mitigate the seesaw phenomenon. Numerous subsequent works follow this design including PFE (Xin et al. 2022), MLPR (Wu et al. 2022) and TAML (Liu et al. 2023). Moreover, some methods based on sparse routings, such as SNR (Ma et al. 2019) and CSRec (Bai et al. 2022), also utilize part of the parameters of the experts for each task, making them a type of Expert-level task-specific model.

In contrast to the above studies, we focus on the **Embedding-level task-specific model**. Specifically, each task has its own embeddings, which is exclusively updated by the corresponding task, as illustrated in Figure 2(d).

Other MTL Methods for Recommender

Multi-task models for cascading tasks. ESMM (Ma et al. 2018b) is the classical model that pays attention to the sequential dependencies between tasks. To address sample selection bias and data sparsity, many studies have proposed variants of ESMM, such as ESM^2 (Wen et al. 2020), Multi-IPW (Zhang et al. 2020), ESCM^2 (Wang et al. 2022), ESDF (Wang et al. 2020), and HM^3 (Wen et al. 2021). AITM (Xi et al. 2021) realizes adaptive information migration for multilayer cascading tasks.

Multi-task optimization methods for conflict mitigation. The vast majority of MTL optimization methods are designed for dealing with gradients, such as gradient balancing (Kendall, Gal, and Cipolla 2018; Liu, Johns, and Davison 2019; Liu, Liang, and Gitter 2019; Chen et al. 2018; He et al. 2022; Yang et al. 2023) or handling gradient conflicts (Yu et al. 2020; Chen et al. 2020; Liu et al. 2021). Recently, ForkMerge (Jiang et al. 2023) employs distinct task weights to optimize multiple models. Subsequently, it merges and synchronizes the parameters of these models, effectively filtering out harmful parameters.

Delve into Negative Transfer in MTL Recommenders

In the context of recommendation, negative transfer refers to the phenomenon where knowledge or information learned from one task adversely affects the performance of another task within a MTL framework. Understanding and mitigating negative transfer is crucial for improving the effectiveness and efficiency of recommender systems. Existing research primarily focuses on investigating negative transfer across all samples as a whole, often overlooking the inherent intricacies within samples. To illustrate this point, let us consider an MTL setting involving two tasks, denoted as `task A` and `task B`. Samples with limited positive feedback from `task A` may intuitively benefit from MTL by receiving additional feedback from `task B`. Further, samples with abundant positive feedback from `task A` can also potentially improve their performance by incorporating complementary feedback from `task B`. However, when a comparable amount of feedback exists for both tasks `A` and `B`, the negative transfer may occur due to the possible contradictory user preferences regarding items between the two tasks.

To verify this hypothesis, we propose to split the testing samples into three distinct zones according to the relative amount of positive feedback between `task A` and `B`: A-Overwhelming sample set $\mathcal{D}_{\text{A-O}}$, B-Overwhelming sample set $\mathcal{D}_{\text{B-O}}$ and Comparable sample set $\mathcal{D}_{\text{Comp}}$. $\mathcal{D}_{\text{A-O}}$ and $\mathcal{D}_{\text{B-O}}$ include all samples with *overwhelming positive feedback* from `Task A` or `B`, respectively. While $\mathcal{D}_{\text{Comp}}$ consists of samples with *comparable positive feedback* from both tasks.

To begin, we measure the amount of positive feedback for each sample from each task separately and then use the gap between them to split the samples into different buckets. Since the *empirical* positive feedback for a sample x is binary, directly using it for dividing buckets is infeasible. Instead, we employ the *estimated* positive feedback from each *single task model*, denoted as $f_A(x)$ and $f_B(x)$. We then discretize the estimated feedback into buckets with equal frequency. The relative amount of positive feedback between `Task A` and `Task B` can then be quantified by the difference in bucket indices: $b(f_A(x)) - b(f_B(x))$. This measurement, referred to as the *Measurement of Relative Feedback* of `Task A` regarding `Task B`, is abbreviated as $\text{MRF}^{A\|B}$.

For a sample x, a low negative value of $\text{MRF}^{A\|B}$ indicates that there is significantly less (estimated) positive feedback from `task A` compared to `task B`. Consequently, sample x belongs to the zone of $\mathcal{D}_{\text{B-O}}$. Conversely, a high positive value of $\text{MRF}^{A\|B}$ suggests a substantial amount of positive feedback from `task A` in comparison to `task B`, leading to the inclusion of sample x in the zone of $\mathcal{D}_{\text{A-O}}$. On

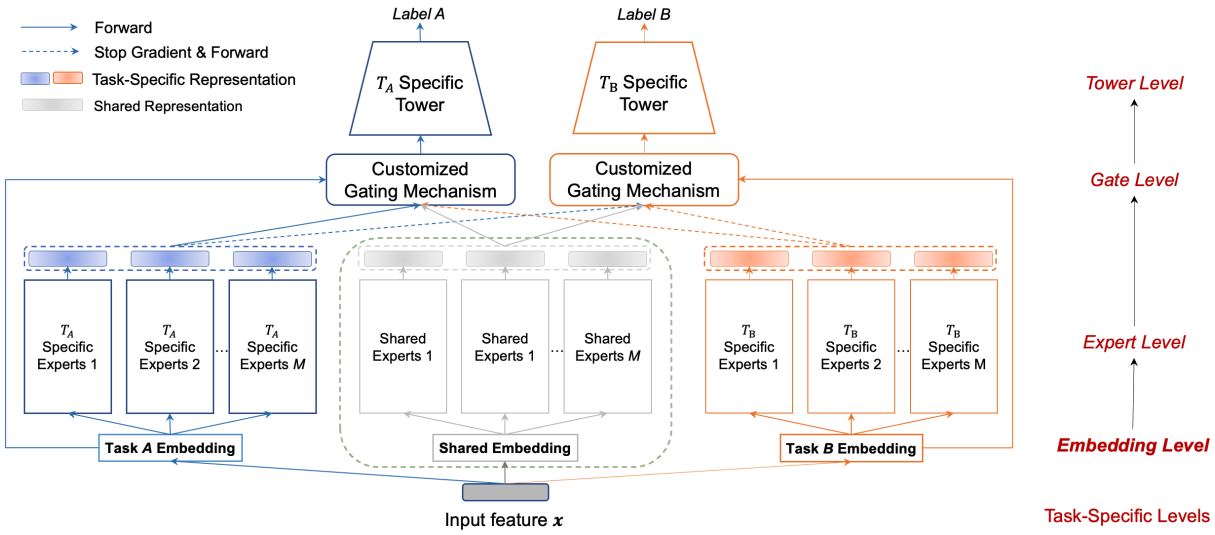


Figure 3: Overview of STEM-Net.

the other hand, a moderate value of $\text{MRF}^{A\|B}$ indicates that sample x falls into the zone $\mathcal{D}_{\text{Comp}}$.

To illustrate this concept, let us consider an example where sample x receives prediction scores of 0.05 and 0.91 from `task A` and `task B`, respectively. Assuming that the prediction scores for each task follow a uniform distribution $U(0,1)$, we can split the percentile buckets into equal ranges of 0.1. In this case, the percentile bucket index for `task A` is $b(f_A(x)) = 0$, and for `task B` is $b(f_B(x)) = 9$. Consequently, the value of $\text{MRF}^{A\|B}$ is calculated as $b(f_A(x)) - b(f_B(x)) = 0 - 9 = -9$, indicating that sample x belongs to the category $\mathcal{D}_{\text{B-O}}$.

We conducted an empirical analysis on the TikTok dataset, which comprises two tasks: `Like` and `Finish`. Our primary objective was to investigate the presence of negative transfer specifically on the `Like` task, as it exhibits a significantly lower number of positive samples compared to the `Finish` task, rendering it susceptible to be dominated. We evaluate the performance of two existing MTL models, MMoE and PLE, and assessed their Area Under the Curve (AUC) on the `Like` task, and compared these results with that obtained from the `Like` single task model. As depicted in Figure 1, both MMoE and PLE demonstrate improved performance when there is overwhelming feedback from either task, *i.e.*, in the categories $\mathcal{D}_{\text{A-O}}$ and $\mathcal{D}_{\text{B-O}}$. However, it is noteworthy that both models exhibit inferior performance compared to the single task model in the category $\mathcal{D}_{\text{Comp}}$, which represents scenarios where there is comparable feedback from both tasks.

These findings shed light on the limitations of the MTL models, MMoE and PLE, in effectively handling situations where there is a balance of feedback from multiple tasks. The observed performance degradation in the $\mathcal{D}_{\text{Comp}}$ zone emphasizes the presence of negative transfer and highlights the need for further investigation and improvement of MTL recommenders.

Methodology

In this section, we describe the architecture of STEM-Net, which is shown in Figure 3.

Shared and Task-specific Embedding Layer

The raw input features $x = \{x_1, x_2, \cdots, x_M\}$ contains M feature fields where x_i represents the active feature of the i-th field. We assign a task-specific embedding table $E^t \in \mathcal{R}^{N \times K}$ for each task t, and a shared embedding table $E^S \in \mathcal{R}^{N \times K}$, where N denotes the total number of features across all fields, and K denotes the embedding dimension. For a given feature x_i, we get the task-specific embedding set $\{\boldsymbol{v}_i^t\}$ as well as the shared embedding $\boldsymbol{v}_i^S$ as follows :

$$\begin{aligned} \boldsymbol{v}_i^t &= \texttt{Lookup}(x_i, E^t) \\ \boldsymbol{v}_i^S &= \texttt{Lookup}(x_i, E^S) \end{aligned} \tag{1}$$

We concatenate the shared and task-specific embeddings across all active features of x as follows:

$$\begin{aligned} \boldsymbol{h}_0^t &= [\boldsymbol{v}_1^t, \ldots, \boldsymbol{v}_i^t, \ldots, \boldsymbol{v}_M^t] \\ \boldsymbol{h}_0^S &= [\boldsymbol{v}_1^S, \ldots, \boldsymbol{v}_i^S, \ldots, \boldsymbol{v}_M^S] \end{aligned} \tag{2}$$

where T denotes the number of tasks, and h_0^t is the embeddings for task $t \in T$.

Shared & Task-Specific Experts

Recently, multi-task models have been utilizing experts to capture shared or task-specific to model the common and different knowledge of multiple tasks. Follow PLE (Tang et al. 2020), we design shared and task-specific experts. However, each shared or specific expert group of a task in our approach is equipped with a corresponding embedding table, so as to prevent any parameters interference. In particular, the shared experts only takes $\boldsymbol{h}_0^S$ as the input, which consists of embeddings from the E^S. In contrast, experts for task t inly takes $\boldsymbol{h}_0^t$ as the input, consisting embeddings from

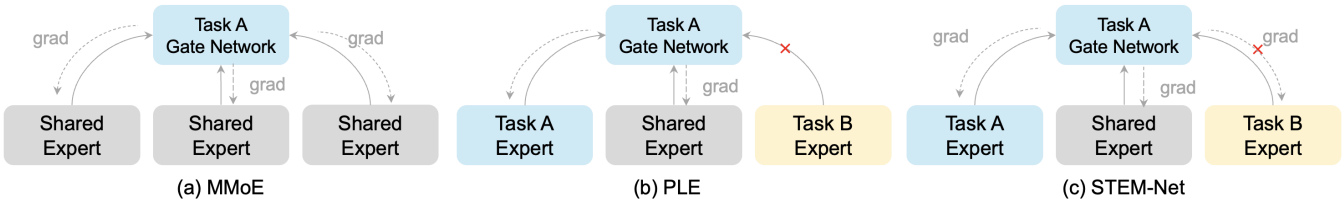


Figure 4: Comparison of gate networks of MMoE, PLE and our STEM-Net.

E^t. For brevity, we assume that each task-specific expert group and shared expert group contains K_1 and K_2 experts respectively. Each expert is a multi-layer perceptros (MLPs) over the input:

$$\begin{aligned} \boldsymbol{h}_i^t &= \texttt{MLP}\texttt{s}_i^t(\boldsymbol{h}_0^t), \forall i = 1, \ldots, K_1 \\ \boldsymbol{h}_j^S &= \texttt{MLP}\texttt{s}_j^S(\boldsymbol{h}_0^S), \forall j = 1, \ldots, K_2 \end{aligned} \tag{3}$$

where $\boldsymbol{h}_i^t$ represents output of the i-th expert for task t and $\boldsymbol{h}_j^t$ represents the output of the j-th shared expert.

Customized Gating Mechanism

The gating mechanism is commonly considered integrate representations from different experts. In STEM-Net, the gating mechanism should ensure that task-specific experts and embeddings are not updated by gradients from other tasks to main specificity, enabling learning task-specific embeddings. We achieve this by designing a customized gating mechanism, with a stop gradient operation to prevent the gradients from the other tasks to update the embeddings of a specific task.

The output of the gating network for task t is formalized as:

$$\begin{aligned} \boldsymbol{o}^t = &\sum_i^{K_1} \boldsymbol{g}_i^{t\to t} \boldsymbol{h}_i^t + \sum_i^{K_2} \boldsymbol{g}_i^{S\to t} \boldsymbol{h}_i^S + \\ &\sum_{t'\in\mathcal{T}, t'\neq t} \sum_i^{K_1} \boldsymbol{g}_i^{t'\to t} \texttt{SG}(\boldsymbol{h}_i^{t'}) \end{aligned} \tag{4}$$

where $\texttt{SG}(\cdot)$ is the stop gradient operation, and $\boldsymbol{g}^{t\to t} \in \mathcal{R}^{K_1}$ denotes the weight on connections between task t's corresponding tower and experts, $\boldsymbol{g}^{t'\to t} \in \mathcal{R}^{K_1}$ denotes the connection weight between the expert of task t' and tower of task t, $\boldsymbol{g}_S^t \in \mathcal{R}^{K_2}$ denotes the connection weight between shared experts and tower of task t, respectively. These gating weights are computed with a softmax over the concatenated embeddings:

$$[\boldsymbol{g}^{t\to t}, \{\boldsymbol{g}^{t'\to t}\}, \boldsymbol{g}^{S\to t}] = \texttt{Softmax}(W_g^t(\boldsymbol{h}_0^t + \boldsymbol{h}_0^S)) \tag{5}$$

where $W_g^t \in \mathcal{R}^{d\times(K_1\times T+K_2)}$.

Towers and Loss Function Finally, we assign each task an independent tower and obtain the final output as:

$$\hat{y}^t = \sigma(\text{MLPs}^t(\boldsymbol{o}^t)) \tag{6}$$

where σ is the sigmoid function to convert the prediction into probability of binary classification.

In this work, we assume that all multi-tasks are binary classification tasks. Therefore, we choose binary cross-entropy loss function as the objective function for each task, and the final loss function is formulated as follows:

$$\mathcal{L} = \sum_i^{|\mathcal{T}|} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \tag{7}$$

Comparison of Gating Networks The comparison of our gating mechanism with those of MMoE and PLE is presented in Figure 4. Our gating mechanism, as opposed to MMoE's, is optimized for both task-specific and shared experts, whereas MMoE's gate network updates all experts without distinction. Compared to PLE, our gate network allows towers to directly transfer knowledge from other task experts, while PLE's gate network requires shared experts as intermediaries for knowledge transfer, which may result in information loss.

Performance Evaluation

Experimental Setup

Public Datasets. In this study, we have carefully chosen three public datasets, namely Tiktok, QK-Video (Yuan et al. 2022), and KuaiRand1K (Yuan et al. 2022) to conduct our experiment. We removed features from all datasets that appeared less than 10 times in the training set and replaced them with a default value. The statistical information of the processed dataset is presented in Table 1.

Baselines. To establish a performance benchmark for comparison, we implement a **Single-Task** model and popular MTL methods including **Shared-Bottom** (Caruana 1997), **OMoE** (Ma et al. 2018a), **MMoE** (Ma et al. 2018a) and **PLE** (Tang et al. 2020). Single-Task adopts the Shared-Bottom but trains only one task. For the Tiktok and QK-Video datasets, we introduce two additional baselines: **ESMM** (Ma et al. 2018b) and **AITM** (Xi et al. 2021). Furthermore, we provide the Multi-Embedding versions of MMoE and PLE as strong baselines to study the effect of simply including multiple embeddings, namely **ME-MMoE** and **ME-PLE**. For ME-MMoE, we allocate independent embeddings for each expert, while for ME-PLE, we assign separate embeddings for task-specific and shared experts.

Hyper-Parameter Settings. We implement all methods refers to the open source recommendation library Fuxi-CTR (Zhu et al. 2021, 2022) and use Adam (Kingma

Table 1: Statistics of processed datasets.

Dataset	#User	#Items	#Samples	#Fields	#Tasks	Positive Ratio (%)
TikTok	560K	1800K	223.4M/24.8M/27.6M	8	2	28.31/1.60
QK-Video	970K	760K	95.9M/12.0M/12.5M	16	2	24.01/2.03
KuaiRand1K	1K	189K	10.9M/0.39M/0.42M	32	8	37.76/1.54/0.10/0.26/0.08/0.10/26.17/1.78

Table 2: Overall performance on TikTok.

Model	Finish		Like		Average
	Logloss↓	AUC ↑	Logloss↓	AUC↑	AUC ↑
Single-Task	0.5111	0.7505	0.0558	0.9058	0.8281
Shared-Bottom	0.5112	0.7504	0.0560	0.9022	0.8263
OMoE	**0.5103**	**0.7516**	0.0559	0.9029	0.8273
MMoE	0.5105	0.7511	0.0560	0.9018	0.8265
PLE	0.5105	0.7511	0.0560	0.9016	0.8264
ESMM	0.5111	0.7503	0.0564	0.9012	0.8258
AITM	0.5109	0.7506	0.0560	0.9026	0.8266
ME-MMoE	0.5114	0.7502	0.0557	0.9045	0.8274
ME-PLE	0.5120	0.7492	0.0560	0.9058	0.8275
STEM-Net	0.5104	0.7513	**0.0553**	**0.9095**	**0.8304**

Table 3: Overall performance on QK-Video.

Model	Click		Like		Average
	Logloss↓	AUC ↑	Logloss↓	AUC↑	AUC ↑
Single-Task	0.2826	0.9234	0.0378	0.9400	0.9317
Shared-Bottom	0.2857	0.9235	0.0380	0.9389	0.9312
OMoE	0.2826	0.9238	**0.0373**	0.9394	0.9316
MMoE	0.2813	0.9238	0.0379	0.9401	0.9319
PLE	0.2832	0.9238	0.0375	0.9399	0.9318
ESMM	0.2847	0.9208	0.0378	0.9368	0.9288
AITM	0.2836	0.9237	0.0386	0.9398	0.9318
ME-MMoE	0.2815	**0.9239**	0.0375	0.9407	0.9323
ME-PLE	0.2818	0.9238	0.0374	0.9410	0.9324
STEM-Net	0.2816	0.9237	0.0381	**0.9426**	**0.9331**

Table 4: Overall performance on KuaiRand1K.

Model	Task A	Task B	Task C	Task D	Task E	Task F	Task G	Task H	Avg. AUC	MTL Gain
Single-Task	0.7534	0.9293	0.8294	0.8943	0.8572	0.8821	0.7650	0.8358	0.8433	-
Shared-Bottom	0.7535	0.9261	0.8162	0.8881	0.8228	0.7820	0.7642	0.8340	0.8234	-0.0199
OMoE	0.7549	0.9273	0.8404	0.8923	0.8352	0.8750	0.7655	0.8349	0.8407	-0.0026
MMoE	0.7541	0.9278	0.8268	0.8901	0.8591	0.8908	0.7647	0.8360	0.8437	+0.0003
PLE	0.7537	0.9290	0.8362	0.8885	0.8449	0.8940	0.7643	0.8374	0.8435	+0.0002
ME-MMoE	**0.7555**	0.9288	0.8310	0.8912	0.8500	0.8668	**0.7658**	**0.8385**	0.8410	-0.0024
ME-PLE	0.7536	**0.9294**	0.8353	**0.8970**	0.8521	0.8871	0.7637	0.8381	0.8445	+0.0012
STEM-Net	0.7523	0.9282	**0.8420**	0.8910	**0.8635**	**0.9070**	0.7637	0.8359	**0.8480**	**+0.0047**

and Ba 2015) as the optimizer. For a fair comparison, we set the learning rate from $\{1e^{-3}, 5e^{-4}, 1e^{-4}\}$, the batch size as 4096, and the l_2 regularization factor of embedding to be $1e^{-6}$. We set the dimension of the embedding to 16, and each expert/bottom is an MLP with hidden units of $[512, 512, 512]$. The towers and the gate networks of all methods are MLPs with hidden units of $[128, 64]$. The number of task-specific and shared experts is chosen from $\{1, 2, 4, 8\}$. Grid search is used to find optimal hyperparameters for all methods.

Overall Performance

The comparison between STEM-Net and the baselines on the TikTok, QK-Video, and KuaiRand1K datasets is respectively showcased in Tables 2, 3 and 4. For AUC, a **0.001**-magnitude lift is generally considered a substantial improvement.

We have the following observations in our study: (1) Our proposed model achieved the best average AUC across all datasets, particularly in tasks with a low positive sample ratio. Specifically, STEM-Net exhibited an average AUC improvement of approximately 0.004, 0.002, and 0.004 compared to the state-of-the-art model PLE on three datasets, respectively. At a fine-grained level, these boosts come mainly from the Like tasks of Tiktok and QK-Video, and Task C to Task F of KuaiRand1K. Our argument is that STEM-Net explicitly preserves the task-specific preferences of these tasks, whereas, in Shared-Embedding-based models, embeddings may be dominated by other tasks. (2) Task-specific embeddings are crucial for effective enhancement, rather than multiple embeddings. Our two proposed strong baselines ME-MMoE, and ME-PLE perform well on all datasets, but neither of them outperforms STEM. ME-MMoE, despite using multiple embeddings, cannot explicitly learn task-specific preferences at the embedding level; ME-PLE improves the task-specificity of embeddings, knowledge transfer between tasks remains implicit, resulting in information loss.

Which feature fields should be task-specific?

We investigate which feature fields should be task-specific to learn diverse user preferences across tasks. To this end, we designed STEM-Net variants that only learning task-specific embeddings for selected fields F, denoted as STEM-Net-(F). Experimental results on the TikTok dataset are presented in Table 5 and Figure 5.

First, if we make all embeddings shared in STEM-Net, denotes as STEM-Net-$\varnothing$, its performance is comparative with that of MMoE and PLE. This demonstrates that our structural change is not the main factor for the performance gain of STEM-Net.

Second, our main purpose is to capture user's preference on items in STEM-Net, so we wonder if we can still achieve decent performance lift when only making embeddings of USER ID and ITEM ID task-specific. We evaluate the per-

Table 5: Performance of STEM-Net variants where only selected feature fields are task-specific.

Variants	Tiktok		KuaiRand1K	
	AUC	#Param	AUC	#Param
STEM-Net-∅	0.8261	1.00x	0.8382	1.00x
STEM-Net-(user id, item id)	0.8302	1.95x	0.8448	1.36x
STEM-Net-(user side)	0.8301	1.23x	0.8437	1.00x
STEM-Net-(item side)	0.8260	2.62x	0.8193	2.05x
STEM-Net-(all features)	0.8304	2.85x	0.8480	2.06x

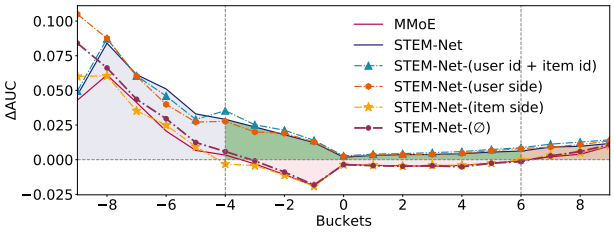


Figure 5: AUC lift of STEM-Net variants compared to the Single-Task model of the task `Like` on task-overwhelming and task-comparable samples.

formance of the corresponding model STEM-Net-(user id, item id), observing that it also achieves competitive performance with STEM-Net. This verify our hypothesis that STEM-Net's performance lift is mainly attributed to its ability to capture diverse user preference.

Further, we are curious about which feature among USER ID and ITEM ID is more critical to deploy task-specific embeddings. We design two new variants, STEM-Net-(user side) and STEM-Net-(item side), and observe the former one exhibit better performance, indicating that user side features are more effective than item in capturing diverse user preference among tasks.

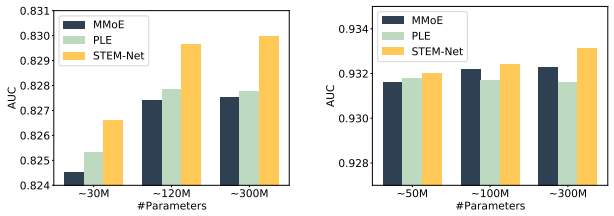
Performance v.s. Model Capacity

To ensure a fair comparison that avoids attributing model performance improvements solely to an increase in model capacity, we take measures to control the number of parameters. Specifically, on the TikTok and QK-Video datasets, we maintain comparability between the baselines and STEM-Net by adjusting the dimension of the embedding. This adjustment allows us to achieve a similar total number of parameters across all models.

As depicted in Figure 6, STEM-Net consistently exhibits superior performance compared to MMoE and PLE, even when the model capacity varies from 30M to 300M on TikTok and 50M to 300M on QK-Video. Moreover, STEM-Net demonstrates better performance scaling-up, as its performance improves with an increase in the number of parameters. In contrast, MMoE shows performance saturation, and PLE experiences a decline in performance.

Is the customized gating mechanism effective?

In STEM-Net, our proposed customized gating mechanism should ensure that embeddings effectively capture task-



(a) Tiktok (b) QK-Video

Figure 6: The effect of parameter capacity.

Table 6: The effect of gate network structure. SG: Stop-Gradient.

Input	SG	Finish AUC	Finish Logloss	Like AUC	Like Logloss	Average AUC
h_0^t	✓	0.7509	0.5107	0.9077	0.0555	0.8293
	✗	0.7511	0.5106	0.9023	0.0560	0.8267
	Δ	+0.0001	-0.0001	-0.0054	+0.0005	-0.0026
h_0^S	✓	0.7510	0.5106	0.9089	0.0553	0.8300
	✗	0.7511	0.5105	0.9037	0.0558	0.8274
	Δ	+0.0001	0.0000	0.0052	+0.0005	-0.0026
$h_0^t + h_0^S$	✓	0.7513	0.5104	0.9095	0.0553	0.8304
	✗	0.7510	0.5106	0.9008	0.0562	0.8259
	Δ	-0.0002	+0.0003	-0.0080	+0.0009	-0.0045

specific user preferences. Here, we verify the necessity of our gating mechanism. Furthermore, the input to the gate network is also worth discussing (Fei et al. 2021; Chang et al. 2023). Thus, we present three variants by replacing the input of Eq 5 from $h_0^t + h_0^S$ to h_0^t and h_0^S. Experimental results are shown in Table 6.

The stop-gradient operation proves to be beneficial for learning task-specific embeddings and improving performance. As shown in Table 6, across all input settings, introducing the stop-gradient operation (highlighted in gray) leads to an AUC improvement of approximately 0.005 to 0.008 for the Like task. Without stop-gradient, the embeddings are essentially shared between tasks. In this case, the performance is close to that of MMoE (shown as Table 2). Thus, we believe that the stop-gradient in our gate mechanism improves the model by ensuring that embedding is task-specific.

Online A/B Test

Online Deployment Since 2022, STEM-Net has been developed on a large display advertising system over various scenarios. These advertising scenarios commonly contain several different tasks including *Follow*, *Activation*, *Fulfill Sheet*, and *Pay*.

Performance The production model follows an MMoE architecture, where each expert is NFwFM, a variant of NFM (He and Chua 2017) which replaces the vanilla FM (Rendle 2010) by FwFM (Pan et al. 2018). We equip the production model with shared and task-specific embeddings. The overall improvements of all tasks are shown in Table 7, indicating a significant improvement of the proposed method over the production model. The introduction

Table 7: AUC LIFT of Online A/B Test

Scenario	Follow	Activation	Fulfill Sheet	Pay	Avg.
Scenario 1	+0.29%	+0.33%	+0.29%	+0.35%	+0.32%
Scenario 2	+0.13%	+0.22%	+0.33%	+0.27%	+0.24%
Scenario 3	+0.47%	+0.39%	+0.28%	+0.78%	+0.48%

of multi-embedding brings 0.32%, 0.24%, and 0.48% average AUC lifts for three representative scenarios, leading to 4.2%, 3.9%, and 7.1% GMV lift in our online A/B test.

Conclusion

In this paper, we propose a novel Share and Task-specific EMbedding paradigm, aiming to incorporate both shared and task-specific embeddings to tackle the negative transfer in MTL recommendation. We design a simple model under such paradigm, namely STEM, which demonstrates compelling performance on comparable samples, achieving positive transfer. In three public datasets and industrial deployments, we validate that our proposed STEM-Net achieves significant AUC improvements over state-of-the-art MTL recommendation models.

References

Bai, T.; Xiao, Y.; Wu, B.; Yang, G.; Yu, H.; and Nie, J. 2022. A Contrastive Sharing Model for Multi-Task Recommendation. In Laforest, F.; Troncy, R.; Simperl, E.; Agarwal, D.; Gionis, A.; Herman, I.; and Médini, L., eds., *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, 3239–3247. ACM.

Caruana, R. 1997. Multitask learning. *Machine learning*, 28(1): 41–75.

Chang, J.; Zhang, C.; Hui, Y.; Leng, D.; Niu, Y.; Song, Y.; and Gai, K. 2023. PEPNet: Parameter and Embedding Personalized Network for Infusing with Personalized Prior Information. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, 3795–3804. New York, NY, USA: Association for Computing Machinery. ISBN 9798400701030.

Chen, Z.; Badrinarayanan, V.; Lee, C.-Y.; and Rabinovich, A. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, 794–803. PMLR.

Chen, Z.; Ngiam, J.; Huang, Y.; Luong, T.; Kretzschmar, H.; Chai, Y.; and Anguelov, D. 2020. Just Pick a Sign: Optimizing Deep Multitask Models with Gradient Sign Dropout. In *NeurIPS*.

Fei, H.; Zhang, J.; Zhou, X.; Zhao, J.; Qi, X.; and Li, P. 2021. GemNN: Gating-enhanced multi-task neural networks with feature interaction learning for CTR prediction. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, 2166–2171.

Hazimeh, H.; Zhao, Z.; Chowdhery, A.; Sathiamoorthy, M.; Chen, Y.; Mazumder, R.; Hong, L.; and Chi, E. 2021. Dselect-k: Differentiable selection in the mixture of experts with applications to multi-task learning. *Advances in Neural Information Processing Systems*, 34: 29335–29347.

He, X.; and Chua, T.-S. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, 355–364.

He, Y.; Feng, X.; Cheng, C.; Ji, G.; Guo, Y.; and Caverlee, J. 2022. MetaBalance: Improving Multi-Task Recommendations via Adapting Gradient Magnitudes of Auxiliary Tasks. *WWW*, 2205–2215.

Jiang, J.; Chen, B.; Pan, J.; Wang, X.; Dapeng, L.; Jiang, J.; and Long, M. 2023. ForkMerge: Overcoming Negative Transfer in Multi-Task Learning. *arXiv preprint arXiv:2301.12618*.

Kendall, A.; Gal, Y.; and Cipolla, R. 2018. Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In *CVPR*, 7482–7491. IEEE Computer Society.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

Liu, B.; Liu, X.; Jin, X.; Stone, P.; and Liu, Q. 2021. Conflict-Averse Gradient Descent for Multi-task Learning. *NeurIPS*, 34.

Liu, S.; Johns, E.; and Davison, A. J. 2019. End-To-End Multi-Task Learning With Attention. In *CVPR*, 1871–1880. Computer Vision Foundation / IEEE.

Liu, S.; Liang, Y.; and Gitter, A. 2019. Loss-Balanced Task Weighting to Reduce Negative Transfer in Multi-Task Learning. In *AAAI*, 9977–9978. AAAI Press.

Liu, X.; Jia, Q.; Wu, C.; Li, J.; Quanyu, D.; Bo, L.; Zhang, R.; and Tang, R. 2023. Task Adaptive Multi-learner Network for Joint CTR and CVR Estimation. In *Companion Proceedings of the ACM Web Conference 2023*, 490–494.

Ma, J.; Zhao, Z.; Chen, J.; Li, A.; Hong, L.; and Chi, E. H. 2019. SNR: Sub-Network Routing for Flexible Parameter Sharing in Multi-Task Learning. In *AAAI*, 216–223. AAAI Press.

Ma, J.; Zhao, Z.; Yi, X.; Chen, J.; Hong, L.; and Chi, E. H. 2018a. Modeling Task Relationships in Multi-task Learning with Multi-gate Mixture-of-Experts. In *KDD*, 1930–1939. ACM.

Ma, X.; Zhao, L.; Huang, G.; Wang, Z.; Hu, Z.; Zhu, X.; and Gai, K. 2018b. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. In *SIGIR*, 1137–1140. ACM.

Misra, I.; Shrivastava, A.; Gupta, A.; and Hebert, M. 2016. Cross-Stitch Networks for Multi-task Learning. In *CVPR*, 3994–4003. IEEE Computer Society.

Pan, J.; Mao, Y.; Ruiz, A. L.; Sun, Y.; and Flores, A. 2019. Predicting different types of conversions with multi-task learning in online advertising. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2689–2697.

Pan, J.; Xu, J.; Ruiz, A. L.; Zhao, W.; Pan, S.; Sun, Y.; and Lu, Q. 2018. Field-weighted factorization machines for click-through rate prediction in display advertising. In *WWW*, 1349–1357.

Qin, Z.; Cheng, Y.; Zhao, Z.; Chen, Z.; Metzler, D.; and Qin, J. 2020. Multitask mixture of sequential experts for user activity streams. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 3083–3091.

Rendle, S. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining (ICDM)*, 995–1000. IEEE.

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 *RecSys*, 269–278. ACM.

Torrey, L.; Shavlik, J. W.; Walker, T.; and Maclin, R. 2010. Transfer Learning via Advice Taking. In Koronacki, J.; Ras, Z. W.; Wierzchon, S. T.; and Kacprzyk, J., eds., *Advances in Machine Learning I: Dedicated to the Memory of Professor Ryszard S. Michalski*, volume 262 of *Studies in Computational Intelligence*, 147–170. Springer.

Wang, H.; Chang, T.; Liu, T.; Huang, J.; Chen, Z.; Yu, C.; Li, R.; and Chu, W. 2022. ESCM2: Entire Space Counterfactual Multi-Task Model for Post-Click Conversion Rate Estimation. In Amigó, E.; Castells, P.; Gonzalo, J.; Carterette, B.; Culpepper, J. S.; and Kazai, G., eds., *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, 363–372. ACM.

Wang, Y.; Zhang, J.; Da, Q.; and Zeng, A. 2020. Delayed Feedback Modeling for the Entire Space Conversion Rate Prediction. *CoRR*, abs/2011.11826.

Wen, H.; Zhang, J.; Lv, F.; Bao, W.; Wang, T.; and Chen, Z. 2021. Hierarchically Modeling Micro and Macro Behaviors via Multi-Task Learning for Conversion Rate Prediction. In Diaz, F.; Shah, C.; Suel, T.; Castells, P.; Jones, R.; and Sakai, T., eds., *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, 2187–2191. ACM.

Wen, H.; Zhang, J.; Wang, Y.; Lv, F.; Bao, W.; Lin, Q.; and Yang, K. 2020. Entire Space Multi-Task Modeling via Post-Click Behavior Decomposition for Conversion Rate Prediction. In *SIGIR*, 2377–2386. ACM.

Wu, X.; Magnani, A.; Chaidaroon, S.; Puthenputhussery, A.; Liao, C.; and Fang, Y. 2022. A Multi-task Learning Framework for Product Ranking with BERT. In Laforest, F.; Troncy, R.; Simperl, E.; Agarwal, D.; Gionis, A.; Herman, I.; and Médini, L., eds., *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, 493–501. ACM.

Xi, D.; Chen, Z.; Yan, P.; Zhang, Y.; Zhu, Y.; Zhuang, F.; and Chen, Y. 2021. Modeling the Sequential Dependence among Audience Multi-step Conversions with Multi-task Learning in Targeted Display Advertising. *KDD*.

Xin, S.; Jiao, Y.; Long, C.; Wang, Y.; Wang, X.; Yang, S.; Liu, J.; and Zhang, J. 2022. Prototype Feature Extraction for Multi-task Learning. In *Proceedings of the ACM Web Conference 2022*, 2472–2481.

Yang, C.; Pan, J.; Gao, X.; Jiang, T.; Liu, D.; and Chen, G. 2022. Cross-Task Knowledge Distillation in Multi-Task Recommendation. *AAAI*.

Yang, E.; Pan, J.; Wang, X.; Yu, H.; Shen, L.; Chen, X.; Xiao, L.; Jiang, J.; and Guo, G. 2023. Adatask: A task-aware adaptive learning rate approach to multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 10745–10753.

Yu, T.; Kumar, S.; Gupta, A.; Levine, S.; Hausman, K.; and Finn, C. 2020. Gradient Surgery for Multi-Task Learning. *NeurIPS*, 33: 5824–5836.

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. 2022. Tenrec: A Large-scale Multipurpose Benchmark Dataset for Recommender Systems. *CoRR*, abs/2210.10629.

Zhang, W.; Bao, W.; Liu, X.; Yang, K.; Lin, Q.; Wen, H.; and Ramezani, R. 2020. Large-scale Causal Approaches to Debiasing Post-click Conversion Rate Estimation with Multi-task Learning. In Huang, Y.; King, I.; Liu, T.; and van Steen, M., eds., *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, 2775–2781. ACM / IW3C2.

Zhu, J.; Dai, Q.; Su, L.; Ma, R.; Liu, J.; Cai, G.; Xiao, X.; and Zhang, R. 2022. BARS: Towards Open Benchmarking

for Recommender Systems. In *SIGIR '22: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Zhu, J.; Liu, J.; Yang, S.; Zhang, Q.; and He, X. 2021. Open Benchmarking for Click-Through Rate Prediction. In *Proceedings of the 30th ACM International Conference on Information Knowledge Management*, CIKM '21, 2759–2769. New York, NY, USA: Association for Computing Machinery. ISBN 9781450384469.