Deep Task-specific Bottom Representation Network for Multi-Task Recommendation

Qi Liu*
University of Science and Technology of China
qiliu67@mail.ustc.edu.cn

Zhilong Zhou
Alibaba Group
zhilong.zhou1996@gmail.com

Gangwei Jiang
University of Science and Technology of China
gwjiang@mail.ustc.edu.cn

Tiezheng Ge†
Alibaba Group
tiezheng.gtz@alibaba-inc.com

Defu Lian
University of Science and Technology of China
liandefu@ustc.edu.cn

ABSTRACT

Neural-based multi-task learning (MTL) has gained significant improvement, and it has been successfully applied to recommendation system (RS). Recent deep MTL methods for RS (e.g. MMoE, PLE) focus on designing soft gating-based parameter-sharing networks that implicitly learn a generalized representation for each task. However, MTL methods may suffer from performance degeneration when dealing with conflicting tasks, as negative transfer effects can occur on the task-shared bottom representation. This can result in a reduced capacity for MTL methods to capture task-specific characteristics, ultimately impeding their effectiveness and hindering the ability to generalize well on all tasks. In this paper, we focus on the bottom representation learning of MTL in RS and propose the Deep Task-specific Bottom Representation Network (DTRN) to alleviate the negative transfer problem. DTRN obtains task-specific bottom representation explicitly by making each task have its own representation learning network in the bottom representation modeling stage. Specifically, it extracts the user's interests from multiple types of behavior sequences for each task through the parameter-efficient hypernetwork. To further obtain the dedicated representation for each task, DTRN refines the representation of each feature by employing a SENet-like network for each task. The two proposed modules can achieve the purpose of getting task-specific bottom representation to relieve tasks' mutual interference. Moreover, the proposed DTRN is flexible to combine with existing MTL methods. Experiments on one public dataset and one industrial dataset demonstrate the effectiveness of the proposed DTRN.

CCS CONCEPTS

• **Information systems** → **Information retrieval**.

*This work was done when the author Qi Liu was at Alibaba Group for intern.
†Corresponding author: Tiezheng Ge

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom.

ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3614837

KEYWORDS

multi-task recommendation, hypernetwork, task-specific representation, behavior sequence modeling

ACM Reference Format:
Qi Liu, Zhilong Zhou, Gangwei Jiang, Tiezheng Ge, and Defu Lian. 2023. Deep Task-specific Bottom Representation Network for Multi-Task Recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3583780.3614837

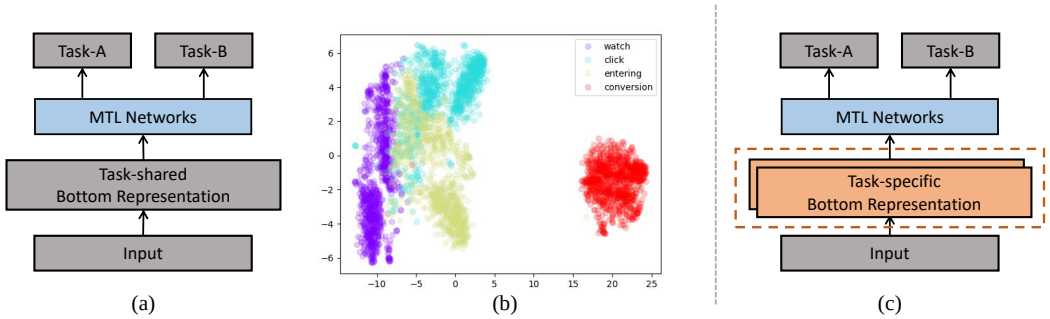
1 INTRODUCTION

Recommendation system (RS), which aims to provide preferred candidates (e.g., items, ads, videos, news, etc) upon user properties, has played a vital role in web applications. In the industrial scenario, RS needs to satisfy multiple objectives, such as click rate, conversion rate, and recommendation diversity. Multi-task learning (MTL) which trains a single network to produce multiple predictions for each objective meets the requirements. Thus MTL has attracted lots of attention in recommendation system [2, 4, 8, 10, 14, 15, 17, 20, 23, 25, 32–35, 41] and make a success in lots of applications.

Table 1: The average times of target item appearing in each behavior sequence for each task.

Method	watch	click	entering	conversion
watch_seq	1.24	1.01	0.83	0.69
non_watch_seq	0.60	0.70	0.61	0.42
click_seq	0.07	0.13	0.09	0.07
non_click_seq	1.76	1.49	1.30	1.06
entering_seq	1.56	2.00	2.57	1.30

The architecture of MTL methods in RS usually follows the paradigm as shown in Figure 1(a). These methods take the task-shared bottom representation as input and extract the task-specific representation through the gating-based parameter-sharing networks. Then, task-specific tower networks predict probabilities of actions (e.g. watch, click, and entering) based on task-specific representations. The benefits of the paradigm lie in two folds. First, it boosts multi-task performance by sharing information among related tasks. Second, it provides a highly compact and efficient mechanism of modeling for training and severing multi-task prediction.



(a) (b) (c)

Figure 1: Existing MTL methods focus on the network structure (blue rectangles in (a)) with their weakness in obtaining task-specific representation (b). We focus on the neglected bottom representation learning (orange rectangles in (c)).

However, MTL in RS still faces the challenge of performance degradation caused by negative transfer between tasks with conflicting correlations. During the optimization process, the conflicting gradients produced by multiple tasks will lead to compromised bottom representation. The compromised bottom representation will inevitably hurt the generalization of the whole network and cause a negative transfer. For example, gradients from the conversion task will emphasize the income-related feature embedding while gradients of the clicking task should mainly focus on the hobby-related feature embedding. At the same time, both tasks give gradients to tyrannize the other related feature embedding which leads to compromised representation. Hence, how to solve the potential conflict between multiple tasks and simultaneously boost all tasks' performance is the core problem in MTL research. There are two major research lines, including the design of optimization strategies and network architectures. Prior studies [5, 13, 16, 18, 30, 37] proposed to modify the gradient magnitude and directions adaptively to alleviate the conflict between multiple tasks' gradients. For network architecture, existing works [1, 4, 7, 8, 19, 20, 25, 31, 32, 35, 45] devoted much effort to the parameter-sharing mechanism, which aimed at solving the negative transfer problem through well-designed network structures being derived from either human's insight or Automated Machine Learning. However, these methods ignore the importance of the task-shared bottom representation. A generalized bottom representation is an essential factor in boosting MTL performance [29]. What's more, we observe that gating-based parameter-sharing methods have a limited ability in eliminating the compromise of representation. As shown in Figure 1(b), we visualize the task-specific representations which are the output of Progressive Layered Extraction (PLE) [25], and there exist overlaps between these representations, which would incur conflicting gradients for different tasks when learning task-specific knowledge. We are motivated to fuse the task information into the bottom representations to obtain task-specific bottom representation as shown in Figure 1(c).

To address the above challenges, we propose a novel MTL model, called Deep Task-specific Bottom Representation Network (DTRN). As shown in Figure 2, it consists of the embedding layer, the task-specific interest module (TIM), the task-specific representation refinement module (TRM), and MTL Networks. Observing that various types of behavior sequences reflect different interests [9], we are motivated to extract the task-specific interests with the multiple types of behaviors. As shown in Table 1, we compute the average occurrence times of the target item's category in the user's each behavior sequence for each task. The result shows the correlation difference between different tasks and behavior sequences. This demonstrates the fine-grained influence difference of different types of historical behaviors on the user's current behavior. Instead of employing an independent behavior sequence modeling module for each task and behavior sequence pair, we innovatively design TIM based on the hypernetwork [11]. In TIM, the number of behavior sequence modeling modules does not increase with the number of tasks and behavior sequences, so it is parameter-efficient and less prone to overfitting. Specifically, the hypernetwork takes the task and behavior embeddings as input to generate dynamic parameters, which will be injected into the base behavior sequence modeling module to produce the task-specific behavior sequence modeling module. Moreover, feature representation should be adaptive to different contexts and has various importance to different tasks. Additionally, to get the task-related context-aware representation namely task-specific bottom representation, we design a task-specific representation refinement module, consisting of multiple task-specific refinement networks to generate context-aware feature representation for each task. The task-specific bottom representation provides more choices for each task to synthesize gradients from other tasks and thus avoid the negative transfer.

The main contributions of this paper are summarized as follows:

- We make a first attempt to achieve task-specific bottom representation learning in MTL models. We propose the Deep Task-specific Representation Network (DTRN) to replace the task-shared bottom representations in the MTL recommendation, which efficiently learns the task-specific bottom representation, and thus alleviates the negative transfer issues.
- We design the Task-specific Interest Module (TIM) to extract the user interest from multiple types of behavior sequences for each task, and the Task-specific Representation Refinement Module (TRM) to obtain the context-aware representation of each task, with the aim of achieving task-specific bottom representations for better generalization.
- We conduct offline experiments on both industrial and public datasets to verify the effectiveness of the proposed method DTRN, which achieves remarkable improvement w.r.t multiple business metrics in the online A/B test experiment.

2 RELATED WORK

2.1 MTL in Recommendation System

Earlier factorization-based models [17, 23] learn task-specific user (or item) embeddings by sharing the other item (or user) embedding. Regularization methods are used to make the learned task-specific embeddings discriminative. Such methods have limited expressive ability and cannot fully unleash the power of MTL. Recently, deep learning-based multi-task models [10, 34, 35] have boosted recommendation performance significantly. Hard-sharing [1] methods exploit the shared bottom network to achieve knowledge transfer, which will cause negative transfer for less relevant tasks. Expert-sharing [20, 25] methods use multiple experts and task-specific gating networks to realize weighted parameter sharing. These methods try to strengthen the dependency and resolve conflicts between tasks. Defined-sharing [8, 21, 31, 32, 35] methods realize knowledge transfer through the human-designed attention mechanism. Learning-base sharing methods [4, 7, 19, 45] utilize the power of AutoML or sparsity to learn the parameter sharing at different granularities automatically. All these methods focus on the design of parameter sharing and ignore the representation learning problem. We think converting the task-agnostic representation to a task-specific representation benefits the MTR.

2.2 Behavior Sequence Modeling

Behavior sequence modeling is an integral part of state-of-the-art recommendation models. DIN [44] discovers that only part items in the behavior sequence are related to the target item, and uses an attention MLP to activate those related items. DIEN [43] utilizes the dynamic change of user interest hidden in the user behavior sequence and proposes a two-layer attention-GRU network to fit this property. SIM [24] adopts the approximate nearest neighbor search technology to process the user's ultra-long behavior sequence. Recently, inspired by the success of Transformer [28] in NLP and CV, [3, 10, 36, 42] try to exploit Self-Attention to model the items' relationship before performing local activation. DMT [10] proposes using separate Transformer encoder-decoder networks to model multiple behavior sequences. However, most existing methods focus on single-type behavior modeling, especially click behavior. In real-world recommendation systems, we can obtain multiple types of behavior sequences based on multiple types of user feedback. These diversified behavior sequences can portray the user character accurately, which is hard to achieve by only exploiting click behavior. Going one step further, we find that the correlations between tasks and behavior sequences diff. There are better solutions than simply extracting a shared interest representation from each behavior sequence for all tasks. The proposed DTRN can extract task-specific interest representations from each behavior sequence in the multi-task and multi-behavior scenarios.

3 METHOD

3.1 Overview

Suppose there are T tasks, N types of sparse ID features, and M types of user behavior sequences. We represent the instance by $\{\mathbf{x}, \mathbf{y}\}$, where $\mathbf{x} = [x_1^s, ..., x_N^s, x_{N+1}^b, ..., x_{N+M}^b]$, $\mathbf{y} = [y_1, y_2, ..., y_T]$. Here x_i^s denotes the sparse ID feature (e.g., user_id, age, item_id and etc), x_i^b denotes the user behavior sequence (e.g., the chronological items clicked by user), and $y_i \in \{0, 1\}$ is the label for each task. MTL in recommendation system aims at estimating the probability $P(y_i|\mathbf{x}), i \in \{1, ..., T\}$ for each instance. We formulate the MTL in RS as the following Eq. (1):

$$P(y_i|\mathbf{x}) = F(\mathbf{x}), i \in \{1, ..., T\}, \tag{1}$$

where F is the MTL model.

DTRN takes $\mathbf{x}$ as input and transforms it into dense vector through the embedding layer. The TIM takes task type embedding, behavior sequence type embedding, and embeddings of behavior sequences as input to extract task-specific interest. The task-specific interest together with sparse ID embeddings contains the original information namely the context of each instance. Based on the context, TRM performs representation refinement by applying Multi-Layer Perception network to adjust the importance of each feature for each task respectively. The task-specific bottom representation generated by TRM can combine with MTL networks (like PLE) to make predictions of multiple tasks.

3.2 Embedding Layer

Sparse ID. x_i^s is the sparse ID feature like user_id, age, item_id, it will be first converted to one-hot vector $\mathbf{t_i^s} \in R^{K_i}$, where K_i denotes the cardinality of feature i. $t_i^s[j] = 1$ if the value of x_i^s is assigned index j in all values. And then an embedding matrix $\mathbf{E_i^s} \in R^{K_i \times d}$ is used to get the embedding vector $\mathbf{e_i^s} = \mathbf{t_i^s E_i^s}$, where d is the embedding dimension.

Behavior Sequence. In RS, users have rich and diverse behaviors, such as clicking items, sharing items, and purchasing items. Such behaviors can be represented by a variable-length sequence of items $\mathbf{x_i^b} = \langle S_{b_1}, S_{b_2}, ..., S_{b_{N_i}} \rangle$, where N_i is the length of behavior sequence and S_{b_k} is the identifier of interacted items (like item_id). Firstly, $\mathbf{x_i^b}$ will be transformed to sparse matrix $\mathbf{t_i^b} \in R^{N_i \times K_i}$, where K_i denotes the cardinalty of feature i. $t_i^b[k, j] = 1$ if the value of S_{b_k} is assigned index j in all values. Embedding matrix $\mathbf{E_i^b} \in R^{K_i \times d}$ is used to get the behavior sequence embedding $\mathbf{e_i^b} = \mathbf{t_i^b E_i^b}$, where d is the embedding dimension.

Task and Behavior Sequence Type There are multiple tasks and multiple behavior sequences in our setting. We assign each task and each type of behavior sequence an embedding vector as their identification. There is a task type embedding matrix $\mathbf{E_T} \in R^{T \times d}$ and the embedding of task-i is $\mathbf{T_i} = E_T[i]$. The behavior sequence type embedding matrix is $\mathbf{E_B} \in R^{M \times d}$ and behavior sequence-b's type embedding is $\mathbf{BS_b} = E_B[b]$, where M is the number of behavior sequence types. Task type and behavior sequence type embedding will also sever as the input of DTRN.

3.3 TIM: Task-Specific Interest Module

As shown in Figure 3, the proposed TIM contains two major submodules, Hypernetwork, and Conditional Transformer. (1) Hypernetwork uses task type embedding and behavior sequence type embedding as inputs to dynamically generate conditional parameters. The generated parameters are supposed to capture the relatedness between corresponding task and behavior sequence. (2) Conditional Transformer contains one base Transformer network

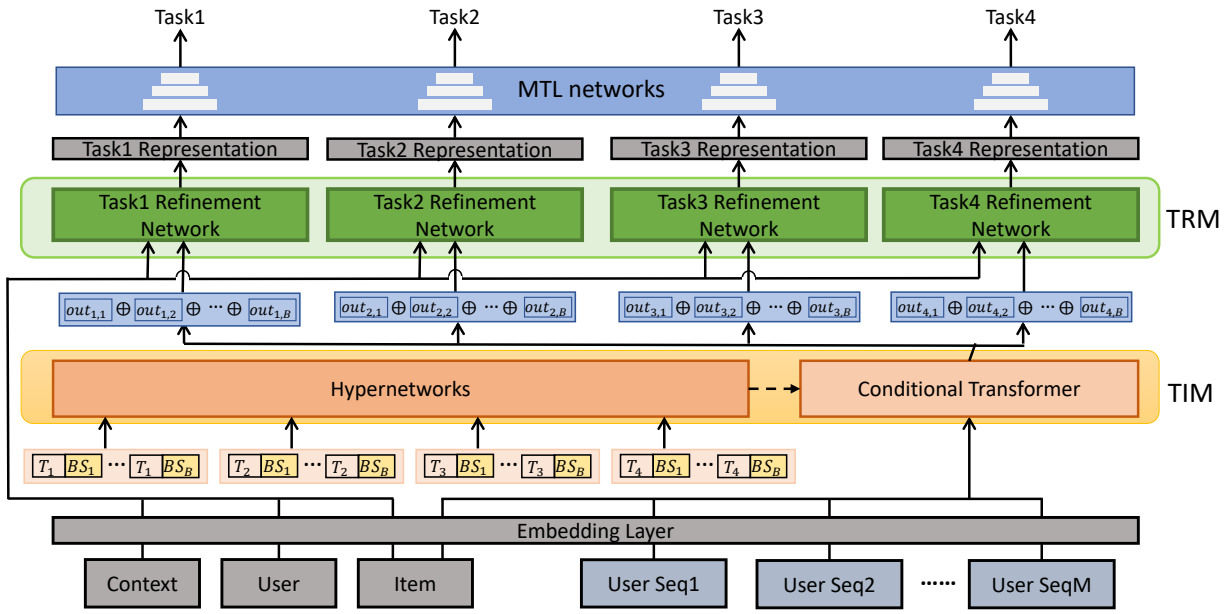


Figure 2: The overall framework of Deep Task-specific Representation Network(DTRN). DTRN consists of Task-specific Interest Module(TIM) and Task-specific Representation Refinement Module(TRM), which learns task-specific bottom representations.

which is applied for modeling of behavior sequence and is shared by all task and behavior sequence pairs. The conditional parameters generated by hypernetwork will be injected into the layer normalization layer of the base Transformer network to achieve the goal of task-specific interest extraction. TIM has the advantage of being parameter-efficient as the parameters only come from the base Transformer network and hypernetwork.

3.3.1 Hypernetwork. The key principle of TIM is to control the unified behavior sequence modeling network producing specific interest for each task and behavior pair. To achieve this, we apply hypernetwork to acquire the task and behavior sequence type embeddings and generate conditional parameters for the task and behavior sequence pair. These parameters will act as additional scale and shift parameters to the layer normalization layer in Transformer to generate the fine-grained user's interest hidden in the behavior sequence towards the specific task. Specifically, we adapt two hypernetworks to generate conditional scale and shift parameters respectively. We implement hypernetwork with a two-layer Multi-Layer Perceptron (MLP) and use ReLU as the hidden layer's activation function. The process is as the following Eq. (2):

$$\gamma^l_{t,b} = MLP_{\theta^l_\gamma}(\mathbf{T_i}, \mathbf{BS_b}), \beta^l_{t,b} = MLP_{\theta^l_\beta}(\mathbf{T_i}, \mathbf{BS_b}), \tag{2}$$

where l represents the position of layer normalization in Transformer. t and b are the type indicator of task and behavior sequence. θ is the parameter of MLP. γ^l and β^l without subscript are the original scale and shift parameters of the layer normalization.

3.3.2 Conditional Transformer. With the success of Transformer in various scenarios, we take it as our base network to extract interest from behavior sequence. However, previous Transformer models can only handle one single task at a time, which is not suitable for the MTL. To this end, we propose a new architecture called Conditional Transformer, which learns the correlation and difference among various behavior sequences for different tasks in one unified model with conditional layer normalization. The components of Conditional Transformer are as follows:

Multi-Head Self-Attention (MHSA): To capture the relationships among item pairs from different subspaces, we apply the multi-head self-attention. MHSA first projects the sequence embedding into the query Q, key K, and value V for h times with different projection matrices. Then, attention mechanism uses the Q and V to compute the relatedness scores between all item pairs in the sequence. The calculated relatedness scores are applied to make a weighted sum of the values V to get new sequence representation. Finally, the output of multiple heads will be concatenated together to get the final result through the projection matrix W^O. The MHSA can be expressed as follows:

$$MHSA(X) = concat(head_1, ..., head_h)W^O, \tag{3}$$

$$head_i = Softmax(\frac{XW^Q_i(XW^K_i)^T}{\sqrt{d'}})XW^V_i, \tag{4}$$

where h is the number of heads, $W^Q_i, W^K_i, W^V_i \in R^{d\times d'}$, $W^O \in R^{d\times d}$. And d and d' is the dimension of the input and weight vectors while $d' = \frac{d}{h}$.

Point-wise Feed-Forward Networks (FFN): To increase the representation ability of Transformer, FFN will be applied to perform nonlinear transform after the MHSA. FFN is a two-layer MLP with the ReLU activation in between, which is as Eq. 5.

$$FFN(X) = ReLU(XW_1 + b_1)W_2 + b_2, \tag{5}$$

where $W_1 \in R^{d \times d_f}$, $b_1 \in R^{d_f}$, $b_2 \in R^d$, $W_2 \in R^{d_f \times d}$ and d_f is the dimension of hidden state.

Conditional Layer Normalization (CLN): Inspired by [22, 26], to allow base Transformer to adapt to each task-behavior sequence pair, the generated conditional scale parameter $\gamma^l_{t,b}$ and conditional shift parameter $\beta^l_{t,b}$ derived from the hypernetwork based on the task and behavior sequence type embeddings will be integrated to the layer normalization, which can be described as follows:

$$CLN^l_{t,b}(X) = \gamma^l_{t,b} \cdot \gamma^l \cdot \frac{X - \mu}{\delta} + \beta^l_{t,b} + \beta^l \tag{6}$$

where l is the position of layer normalization, t is the task type, b is the behavior sequence type, μ, δ is the mean and standard deviation of the input. The CLN empowers Transformer to have the ability to manipulate hidden representations by scaling them up or down, shifting them left or right, shutting them off, etc based on different conditional information with minimal parameters cost.

Components Integration: As shown in Figure 3, Conditional Transformer follows the encoder-decoder structure. The encoder has the ability to capture the relatedness of items by swapping information between items from different latent spaces with the help of MHSA, which can strengthen the representation of each item. The decoder also exploits the power of MHA to finish the local activation to get interest by using the target item as query and the output of encoder as keys and values. MHA (Multi-head attention) is similar to MHSA, except that query comes from target item. Specifically, the encoder and decoder layer are both made up of the MH(S)A, the FFN, and the CLN. They both exploit residual connections around each sublayer. The integrated components can be expressed as Eq. (7):

$$\begin{aligned}
out^0_{t,b} &= CLN_{t,b}(X + MHSA(X)), \\
out^{enc}_{t,b} &= CLN_{t,b}(out^0_{t,b} + FFN(out^0_{t,b}), \\
out^3_{t,b} &= CLN_{t,b}(e_{item} + MHA(e_{item}, out^{enc}_{t,b})), \\
out^{dec}_{t,b} &= CLN_{t,b}(out^3_{t,b} + FFN(out^3_{t,b})),
\end{aligned} \tag{7}$$

where $X = e^s_i, i \in [N+1, N+M]$ is the embedding of behavior sequence-i, e_{item} is the embedding of target item. The final output $out^{dec}_{t,b}$ is the interest extracted from behavior sequence-i for task-t. As we have T tasks and M types of behavior sequences, we will get $T \times M$ such interests.

Task-Specific Interest: After getting $T \times M$ interests for each task and behavior sequence pair, we will concatenate interests grouped by task to obtain the task-specific interest as Eq. (8):

$$interest_i = concat(out^{dec}_{i,1}, out^{dec}_{i,2}, ..., out^{dec}_{i,M}), i \in \{1, ..., T\}, \tag{8}$$

where $interest_i$ is the task-i's task-specific interest.

3.4 TRM: Task-specific Representation Refinement Module

The TRM takes sparse ID embeddings with task-specific interests as input and aims at refining the representation of each feature. The refinement has two advantages. First, it empowers feature representation to be adaptive under different contexts. For example, the feature *food*'s representation should be related to fast food on Wednesday while should contain information about restaurants

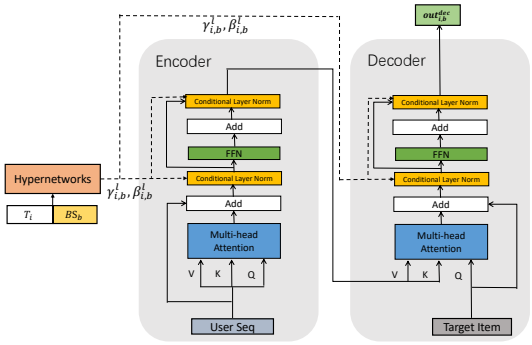


Figure 3: The network structure of TIM.

on Sunday. Second, it meets the requirements that different features have various importance for different tasks [12]. We develop a SENet-like network to achieve the above advantages by assigning importance vector on the raw dense vector of all features for each task based on the context. First, we concatenate sparse ID embeddings together with task-specific interest to get raw task-specific bottom representation $raw_i = [e^s_1, e^s_2, ..., e^s_N, interest_i]$. The raw task-specific bottom representation contains the original information namely context about the instance and serves as input of each task's SENet-like network. The output of SENet-like network acts as refinement vector and will do element-wise multiply with raw task-specific bottom representation to perform task-related context-aware representation refinement to get the task-specific bottom representation. We implement SENet-like network through a two-layer MLP with ReLU as the hidden layer's activation and Sigmoid as the output layer's activation function. The whole process is as Eq. (9):

$$r_i = raw_i \odot refine_i,\ refine_i = MLP_i(raw_i), \tag{9}$$

where $r_i, i \in \{1, ..., T\}$ is the task-specific bottom representation.

3.5 Multi-Task Learning

There are usually multiple objectives in an industrial recommendation system, such as click, like, and conversion. The ranking system should be able to learn and estimate multiple objectives and combines these estimations to compute a final preference ranking score. The proposed DTRN can combine with MTL methods in RS for multi-task learning. With the help of TIM and TRM, we can get T different task-specific bottom representation r_i. Individual r_i will be fed into the MTL model parallelly. Then, the MTL model output the multi-task logits $[o_1, o_2, .., o_T]$, which is illustrated as Eq. (10):

$$[o_1, o_2, ..., o_T] = MTL\_model(r_1, r_2, ..., r_T), \tag{10}$$

The loss function for the task-t is L_t. The total loss L is the sum of losses in multi-task:

$$L = \sum_{t=1}^{T} L_t(\hat{y}_t, y_t), \hat{y}_t = \sigma(o_t), \tag{11}$$

where y_t is the ground truth label, $\hat{y}_t$ is the prediction probability, and σ denotes the sigmoid function. In our experiments, we set all L_t binary cross entropy loss.

Table 2: The statistic of the E-commerce dataset.

#Sample	#Feature	#User	#Item
376,197,451	116	59,677,310	41,343
#Watch	#Click	#Entering	#Conversion
218,732,912	5,648,867	2,747,893	45,972

Sequence	Watch	Non Watch	Click	Non Click	Entering
avg length	9.98	7.34	0.65	16.96	22.80
max length	50	50	50	50	100

Table 3: The statistic of the Tenrec dataset.

#Sample	#Feature	#User	#Item
117,551,246	16	993,554	821,744
#Click	#Follow	#Like	#Share
28,358,705	176,496	2,227,004	243,310

Sequence	Click	Follow	Like	Share	Negative
avg length	11.62	0.12	0.87	0.21	9.8
max length	50	10	20	10	50

4 EXPERIMENTS SETUP

4.1 Datasets

E-commerce Dataset: We collect traffic logs from the display advertising system in Alibaba's micro-video e-commerce platform. One-month samples are used for training and samples of the following day are for testing. The size of the training and testing set is about 361 million and 15 million respectively. We need to predict four tasks: *watch, click, entering, and conversion*. Task *watch* means whether the user will watch the recommended micro-video for more than three seconds. Task *click* measures the probability that user touches the "like" button during watching. Task *entering* indicates that user goes to the detailed page of the merchandise during watching. Task *conversion* expresses that the user purchases the merchandise finally. There are five behavior sequences that are aggregated based on different types of user behavior. Detailed statistics about the E-commerce dataset are shown in Table 2.

Tenrec Dataset:* Tenrec dataset is also the collected user logs from the industrial platform. We conduct experiments on a subset named **QK-Video-1M** which contains one million users' behavior logs. We also need to predict four tasks in this dataset: *click, follow, like, and share. click* means the user clicks the displayed video and begins watching. *follow* indicates the user follows the video's uploader. *like* means the user touches the "like" button, and *share* expresses the user shares the video with his/her friends on social media. There are also five types of behavior sequences. The negative behavior sequence consists of videos that are skipped by the user. Statistics of the Tenrec dataset are shown in Table 3.

*https://github.com/yuangh-x/2022-NIPS-Tenrec

4.2 Baselines

We choose baselines from three aspects. First, we choose methods focusing on behavior sequence modeling, including **YouTubeNet** [6], **DIN** [44], **DIEN** [43], **BST** [3], **ATRank** [42], **DeepFeed** [36], **DMT** [10] which extract user's interest vector from a single type of behavior sequence or multiple types of behavior sequences. Second, we compare with existing MTL methods in RS, including **MMoE** [20], **PLE** [25], **M2M** [38], **AITM** [35]. Third, **Samll-Heads** [29] which concentrates on improving the generalization of the task-shared bottom representation.

4.3 Evaluation Metric

Two widely used metrics AUC, and LogLoss are chosen. The AUC (Area Under the ROC Curve) measures the ranking accuracy. A higher AUC indicates better performance. The LogLoss measures the accuracy of the estimated probability depending on the ground-truth label. Even a slight improvement is considered a significant boost for the industry recommendation task, as it leads to a significant increase in revenue.

4.4 Implementation Details

For the details of experiments, there is a slight difference between the two datasets. For the E-commerce Dataset, there are causation relationships among tasks, which urges us to select AITM [35] as the MTL model combined with DTRN. One line of causality is *watch->click*. Another line of causality is the *watch->entering->conversion*. Specifically, the causality of *watch->entering->conversion* means that the user will first watch the micro-video for more than three seconds, then goes to the detailed page of the merchandise and purchase the merchandise finally. As the tasks in the Tenrec dataset have no apparent causal relationship, we apply the MMoE [20] as the MTL model combined with DTRN. For baselines of focusing on a single type of behavior sequence, we choose the watch behavior sequence for E-commerce related experiments and the click behavior sequence for the Tenrec dataset. Notice, we still do multi-task learning for a single type of behavior sequence baselines. For the other baselines, we use all five types of behavior sequences in both two datasets. For the batch size, we set it to $2,048$ for both datasets and use Adam as the optimizer. The learning rate is $1e-4$ and $1e-3$ for E-commerce and Tenrec respectively, and one-epoch training is applied due to the one-epoch phenomenon [40]. We run all experiments through XDL2[39].

5 EXPERIMENT RESULTS

5.1 Performance Comparison

Table 4 shows the results of all methods. DTRN achieves the best metrics on all tasks in both the E-commerce and Tenrec datasets, which verifies the effectiveness of DTRN. There are some insightful findings from the results. (1) The proposed DTRN beats all baselines on all tasks. Compared with methods of behavior sequence modeling, DTRN achieves more fine-grained interest representation for each task and behavior sequence pair through the Transformer

Table 4: Performance comparison of baselines on two datasets. The best result is in boldface and the second best is underlined. ** indicates that difference to the best baseline is statistically significant at 0.01 level, and * represents 0.05 level.

Dataset	Method	Watch		Click		Entering		Conversion	
		AUC	LogLoss	AUC	LogLoss	AUC	LogLoss	AUC	LogLoss
E-commerce	YouTubeNet	0.7238	0.606855	0.7113	0.077608	0.8171	0.043184	0.8647	0.001475
	DIN	0.7245	0.606214	0.7087	0.077748	0.8171	0.043144	0.8636	0.001488
	DIEN	0.7257	0.605454	0.7110	0.077597	0.8173	0.043134	0.8650	0.001478
	ATRANK	0.7316	0.600047	0.7186	0.077087	0.8158	0.043219	0.8484	0.001539
	DFN	0.7318	0.600256	0.7230	0.076919	0.8232	0.042808	0.8727	0.001466
	DMT (MMoE)	0.7338	0.598775	0.7246	0.076762	0.8237	0.042764	0.8726	0.001466
	PLE	0.7340	0.598761	0.7237	0.076778	0.8231	0.042772	0.8693	0.001472
	AITM	0.7337	0.598964	0.7245	0.076755	0.8240	0.042739	0.8691	0.001475
	M2M	0.7343	0.598345	0.7245	0.076732	0.8238	0.042740	0.8674	0.001478
	SmallHeads	0.7342	0.598699	0.7246	0.076709	0.8240	0.042726	0.8673	0.001471
	DTRN	**0.7346****	**0.59816****	**0.7271****	**0.076584****	**0.8283****	**0.042496****	**0.8828****	**0.001456****

Dataset	Method	Click		Follow		Like		Share	
		AUC	LogLoss	AUC	LogLoss	AUC	LogLoss	AUC	LogLoss
Tenrec	YouTubeNet	0.9458	0.189366	0.8860	0.002852	0.9289	0.024425	0.9091	0.002369
	DIN	0.9460	0.189854	0.8863	0.002897	0.9271	0.024079	0.9032	0.002375
	DIEN	0.9466	0.188511	0.8820	0.002927	0.9308	0.023904	0.9074	0.002408
	ATRANK	0.9467	0.188962	0.8954	0.002879	0.9362	0.023156	0.9198	0.002412
	DFN	0.9504	0.181401	0.9063	0.002768	0.9489	0.021467	0.9244	0.002301
	DMT (MMoE)	0.9506	0.180974	0.9152	0.002724	0.9501	0.021243	0.9279	0.002280
	PLE	0.9508	0.180515	0.9149	0.002684	0.9504	0.021063	0.9283	0.002262
	AITM	0.9507	0.180837	0.9102	0.002711	0.9501	0.021218	0.9252	0.002305
	M2M	0.9507	0.180715	0.9128	0.002645	0.9506	0.021042	0.9289	0.002247
	SmallHeads	0.9510	0.180377	0.9150	0.002660	0.9508	0.021304	0.9274	0.002252
	DTRN	**0.9516***	**0.179113***	**0.9201***	**0.002577***	**0.9519***	**0.020913***	**0.9331***	**0.002220***

network and hypernetwork. When combined with existing MTL methods in RS, DTRN can boost performance significantly. Both the proposed TIM and TRM module work and ease the learning of the following MTL model. This demonstrates that task-specific bottom representation can alleviate the negative transfer to some extent. (2) The performance can be significantly improved by using methods that model multiple types of behavior sequences. For example, methods of modeling multiple types of behavior sequences ATRANK, DFN, and DMT obtain improvement over the methods of modeling the single type of behavior sequence YouTubenet, DIN, DIEN. It reveals that the single type of behavior sequence is not sufficient for describing the user's interest. Exploiting multiple types of behavior sequences can comprehensively capture the user's various interests from a different perspective. (3) The negative transfer effects on the shared bottom representation can limit the effectiveness of gating-based parameter-sharing mechanisms. This is evident in the performance of MTL methods in RS, such as MMoE, PLE, AITM, and M2M, which exhibit similar and relatively poor results. SmallHeads which focuses on improving the generalization of the task-shared bottom representation by introducing extra self-auxiliary losses gains improvements over those MTL methods. The result shows the improvement on the bottom representation is reasonable and can boost the MTL performance. (4) Among methods that take multiple types of behavior sequences, the interest representation extracted by DMT gains the best performance. We think the reason is that DMT allocates individual Transformer encoder-decoder for each behavior sequence respectively, which achieves behavior sequence-specific interest modeling. On the contrary, ATRank directly learns the relatedness of the stitched multiple types of behavior sequences and has the lowest performance. Such results demonstrate the necessity of fine-grained interest representation.

5.2 Ablation Studies

We conduct ablation studies on the E-commerce dataset to analyze the effectiveness of DTRN's components.

5.2.1 Effectiveness of Components in DTRN. Firstly, we investigate how the TIM and TRM module influence the performance of DTRN and demonstrate the results in Table 5. The ablation experiments are conducted based on the baseline DMT. We integrate TIM with DMT by concatenating the extracting task-specific interest with sparse ID embeddings as partial task-specific bottom representation for each task. When combining TRM with DMT, we set different task-specific representation refinement networks for each task and make the task-shared interest together with sparse ID embeddings serve as input.

From Table 5, we can find that integrating DMT either with TIM or TRM can outperform the DMT on all tasks, which indicates that full or even partial task-specific bottom representation benefits the MTL. The designed TIM module can capture the fine-grained correlation difference between different tasks and different behavior sequences, reflected as task-specific interest, and gains performance promotion. For the TRM, it achieves the task-related context-aware

representation refinement by applying separate refinement network for each task and also increases the metrics. The result of DTRN shows that task-specific interest is an important information of context and the vital refinement operation should act on the task-specific raw bottom representation.

Table 5: AUC of TRM and TIM.

Method	watch	click	entering	conversion
DMT	0.7337	0.7245	0.8240	0.8691
+TIM	0.7339	0.7255	0.8264	0.8786
+TRM	0.7339	0.7257	0.8266	0.8724
DTRN	**0.7346**	**0.7271**	**0.8283**	**0.8828**

5.2.2 The Way of Injecting Conditional Information into Transformer. In this section, we do experiments to explore where should the conditional information generated by hypernetwork be injected into. According to the architecture of Transformer, we divide it into QKV, FFN-1, FFN-2, and LN four components. The QKV component generates the "query", "key", and "value" through three separate matrices W^Q, W^K, W^V. We inject the conditional information into QKV through the residual mechanism. Specifically, $W^Q_{t,b} = W^Q \odot (1.0 + W_{t,b})$, where $W_{t,b}$ is reshaped from the generated parameters by the hypernetwork. $W^Q_{t,b}$ is the actual matrix for generating "query". W^K, W^V hold the same procedure with respective hypernetwork. FFN-1, and FFN-2 represent the first and second layer of the Position-wise Feed-Forward Network. The conditional information will be injected into the FFN's parameters W_1, b_1 and W_2, b_2. The injection way is the same as QKV through the residual mechanism. LN indicates the layer normalization in Transformer. LN injected with conditional information is our proposed CLN.

The experiment results are shown in Table 6. We can see that all the ways have the ability to extract task-specific interest, alleviate compromised task-shared bottom representation, and gain improvement over DMT. Concretely, the best result of LN (DTRN) compared with variants QKV, FFN-1, and FFN-2 highlight the critical role of layer normalization in Transformer, which is consistent with the conclusion in NLP [22]. Intuitively and mathematically, layer normalization is easier to achieve the representation specificity by scaling, shifting, and ignoring the middle hidden representation.

Table 6: AUC of implementing Conditional Transformer.

Method	watch	click	entering	conversion
DMT	0.7337	0.7245	0.8240	0.8691
QKV	0.7348	0.7251	0.8265	0.8786
FFN-1	**0.7352**	0.7266	0.8280	0.8707
FFN-2	0.7348	0.7251	0.8271	0.8715
LN (DTRN)	0.7346	**0.7271**	**0.8283**	**0.8828**

5.2.3 Combining DTRN with Different MTL Models. Since we focus on the task-specific bottom representation which is orthogonal to existing multi-task learning models. We show its effectiveness by combining DTRN with different MTL models in RS. The result in Table 7 shows the task-specific bottom representation output by DTRN gives each task stronger ability to acquire its desired representation, alleviating the conflict among tasks, and boosting the performance of all MTL models. What's more, AITM outperforms ShareBottom, MMoE, and PLE when combined with DTRN. This indicates that appropriate network architecture is also an important factor in MTL and can reinforce its own advantage with task-specific bottom representation.

Table 7: AUC of combining DTRN with different MTL models.

Method	watch	click	entering	conversion
ShareBottom	0.7338	0.7242	0.8235	0.8662
ShareBottom+DTRN	**0.7348**	**0.7270**	**0.8274**	**0.8713**
MMoE	0.7338	0.7246	0.8237	0.8726
MMoE+DTRN	**0.7348**	**0.7267**	**0.8270**	**0.8751**
PLE	0.7340	0.7237	0.8231	0.8692
PLE+DTRN	**0.7348**	**0.7268**	**0.8273**	**0.8743**
AITM	0.7337	0.7245	0.8240	0.8691
AITM+DTRN	**0.7346**	**0.7271**	**0.8283**	**0.8828**

5.2.4 Effect of Different Tasks. In this subsection, we perform ablation study to observe the influence of different tasks. We still use the DMT for comparison. We study the influence of each task by removing each task in turn and present the result in Table 8. We have the following observations. (1) DTRN outperforms DMT on all tasks in all ablation experiments, which indicates DTRN is robust to different task combinations. (2) From the first two row of Table 8, we find the *watch* task benefits the other task greatly. As the performance of *click*, *entering*, *conversion* drops severely when removing *watch* task. The reason is that the *watch* action is the foundation of the other three tasks because you proceed to the next action only when you have watched the micro-video for more than three seconds and know what the displayed micro-video is. Thus, the *watch* task which contains more positive samples provides more meaningful supervisory signals for the other three tasks. (3) The promotion between *watch* and *click* task is unidirectional. Removing *click* task does not affect the performance of *watch* task. (4) The *entering* task is important for the *conversion* task. The reason is that the *conversion* action happens only after the *entering* task. Directly removing the causal relationship between two tasks leads to performance degradation. (5) The influence among tasks is asymmetrical. The gradients from *entering* task benefit the *conversion* task, but the gradients from *conversion* task has no or negative effect on the *entering* task, the same as *watch* and *click* task. (6) Above all, result of Table 8 shows the complex relationship among tasks. Task-specific bottom representation is necessary and effective for handling the complexity.

5.2.5 Hyperparamter: Embedding Dimension. For hyperparameter, we study the effect of embedding dimension. As Figure 4 shows, DTRN always gains promotion over DMT regardless of the embedding dimension. Another observation is that the sparser the task (i.e. less positive samples), the more promotion. We think the reason is that the magnitude of the sparser task's gradient is smaller and will be influenced by the other tasks heavily. We think that such influence can enhance its performance but the task-shared bottom representation gives a limited promotion upper bound. As above analysis, influence among tasks is not always positive, task-specific

Table 8: AUC of removing each task in turn.

Method		watch	click	entering	conversion
-watch	DMT	-	0.7153	0.8196	0.8631
	DTRN	-	0.7170	0.8221	0.8780
-click	DMT	0.7331	-	0.8236	0.8649
	DTRN	0.7349	-	0.8282	0.8811
-entering	DMT	0.7327	0.7239	-	0.8652
	DTRN	0.7343	0.7252	-	0.8703
-conversion	DMT	0.7327	0.7246	0.8242	-
	DTRN	0.7343	0.7270	0.8284	-

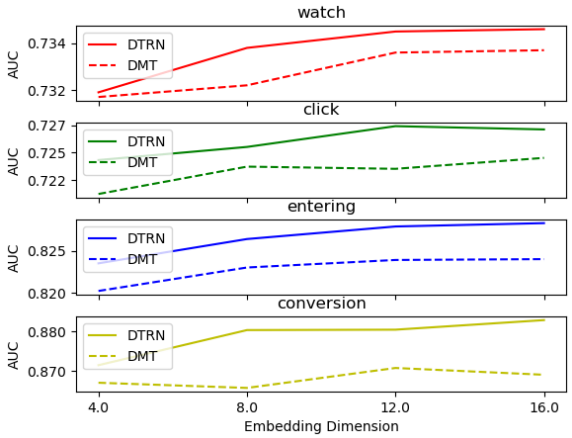


Figure 4: AUC of different embedding dimensions.

bottom representation makes each task have the chance to customize its own representation and hence boosts the performance. Especially, customized representation benefits the sparser tasks that are difficult to train notoriously more.

5.3 Case Study

5.3.1 Task-specific Interest. For understanding the TIM module, we visualize the task-specific interest $interest_i$ in Eq. (8). We select the watch_seq and click_seq for visualization. For both behavior sequences, we first randomly select 1000 instances that satisfy the length of the corresponding behavior sequence is more than 40. Then, we feed the 1000 instances into the DTRN to get 4×1000 task-specific interest vectors for each behavior sequence respectively. Finally, we visualize these task-specific interest vectors with t-SNE [27] in Figure 5, and we can find that task-specific interest is clearly distinguished. The result shows that the TIM module based on hypernetwork can capture correlation difference between tasks and behavior sequences, and then finish the extraction of the fine-grained interest.

5.3.2 Task-specific Bottom Representation. Finally, we visualize the task-specific bottom representation, which is the output of the TRM module (i.e. r_i in Eq. (9).). We randomly select 1000 instances from the test dataset and feed them into the DTRN to get the task-specific bottom representation. The result is shown in Figure 6, we can see that the representation of each task can be completely separated.

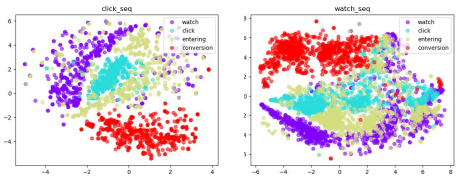


Figure 5: Visualization of Task-Specific Interest using t-SNE.

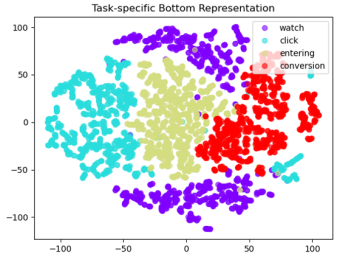


Figure 6: Visualization the task-specific representation.

This result shows that both our designed DTRN can introduce the task prior knowledge into the bottom representation and obtain the task-specific bottom representation for each task.

5.4 Online Results

We conduct A/B test in the online display advertising system to measure the benefits of DTRN compared with the online baseline DMT(MMoE). Both methods are allocated with 10% serving traffic for 30 days. Table 9 shows the relative promotion of four corresponding objectives. This is a significant improvement in an online display advertising system and proves the effectiveness of DTRN.

Table 9: A/B Test of DTRN compared to DMT.

Indicators	Accumulated Gains
Watch Rate	+0.63%
Click Rate	+1.00%
Entering Rate	+2.23%
Revenue Per Mille	+1.00%

6 CONCLUSION

In this paper, we propose the Deep Task-Specific Bottom Representation Network for the MTL in RS. DTRN which contains TIM and TRM modules aims at generating task-specific bottom representation. We conduct offline/online experiments to verify the effectiveness of the DTRN. Finally, we do some ablation studies and visualization to show the correctness of DTRN's component.

7 ACKNOWLEDGMENTS

The work was supported by Alibaba Group through Alibaba Innovative Research Program. Defu Lian was supported by grants from the National Natural Science Foundation of China (No. 62022077 and 61976198).

REFERENCES

[1] Rich Caruana. 1997. Multitask learning. *Machine learning* 28, 1 (1997), 41–75.

[2] Jin Chen, Defu Lian, Binbin Jin, Xu Huang, Kai Zheng, and Enhong Chen. 2022. Fast variational autoencoder with inverted multi-index for collaborative filtering. In *Proceedings of the ACM Web Conference 2022*. 1944–1954.

[3] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[4] Xiaokai Chen, Xiaoguang Gu, and Libo Fu. 2021. Boosting share routing for multi-task learning. In *Companion Proceedings of the Web Conference 2021*. 372–379.

[5] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*. PMLR, 794–803.

[6] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.

[7] Ke Ding, Xin Dong, Yong He, Lei Cheng, Chilin Fu, Zhaoxin Huan, Hai Li, Tan Yan, Liang Zhang, Xiaolu Zhang, et al. 2021. MSSM: a multiple-level sparse sharing model for efficient multi-task learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2237–2241.

[8] Chen Gao, Xiangnan He, Dahua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, Lina Yao, Yang Song, and Depeng Jin. 2019. Learning to recommend with multiple cascading behaviors. *IEEE transactions on knowledge and data engineering* 33, 6 (2019), 2588–2601.

[9] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time personalization using embeddings for search ranking at airbnb. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 311–320.

[10] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, Lixin Zou, Yiding Liu, and Dawei Yin. 2020. Deep multifaceted transformers for multi-objective ranking in large-scale e-commerce recommender systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2493–2500.

[11] David Ha, Andrew Dai, and Quoc V Le. 2016. Hypernetworks. *arXiv preprint arXiv:1609.09106* (2016).

[12] Tongwen Huang, Zhiqi Zhang, and Junlin Zhang. 2019. FiBiNET: combining feature importance and bilinear feature interaction for click-through rate prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 169–177.

[13] Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7482–7491.

[14] Defu Lian, Qi Liu, and Enhong Chen. 2020. Personalized ranking with importance sampling. In *Proceedings of The Web Conference 2020*. 1093–1103.

[15] Defu Lian, Haoyu Wang, Zheng Liu, Jianxun Lian, Enhong Chen, and Xing Xie. 2020. Lightrec: A memory and search-efficient recommender system. In *Proceedings of The Web Conference 2020*. 695–705.

[16] Xiao Lin, Hongjie Chen, Changhua Pei, Fei Sun, Xuanji Xiao, Hanxiao Sun, Yongfeng Zhang, Wenwu Ou, and Peng Jiang. 2019. A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In *Proceedings of the 13th ACM Conference on recommender systems*. 20–28.

[17] Nathan N Liu, Evan W Xiang, Min Zhao, and Qiang Yang. 2010. Unifying explicit and implicit feedback for collaborative filtering. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. 1445–1448.

[18] Shikun Liu, Edward Johns, and Andrew J Davison. 2019. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 1871–1880.

[19] Jiaqi Ma, Zhe Zhao, Jilin Chen, Ang Li, Lichan Hong, and Ed H Chi. 2019. Snr: Sub-network routing for flexible parameter sharing in multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 216–223.

[20] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1930–1939.

[21] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire space multi-task model: An effective approach for estimating post-click conversion rate. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 1137–1140.

[22] Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. *arXiv preprint arXiv:2106.04489* (2021).

[23] Weike Pan, Shanchuan Xia, Zhuode Liu, Xiaogang Peng, and Zhong Ming. 2016. Mixed factorization for collaborative recommendation with heterogeneous explicit feedbacks. *Information Sciences* 332 (2016), 84–93.

[24] Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2685–2692.

[25] Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. 2020. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *Fourteenth ACM Conference on Recommender Systems*. 269–278.

[26] Yi Tay, Zhe Zhao, Dara Bahri, Don Metzler, and Da-Cheng Juan. 2021. Hypergrid transformers: Towards a single model for multiple tasks. (2021).

[27] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[29] Yuyan Wang, Zhe Zhao, Bo Dai, Christopher Fifty, Dong Lin, Lichan Hong, Li Wei, and Ed H Chi. 2022. Can Small Heads Help? Understanding and Improving Multi-Task Generalization. In *Proceedings of the ACM Web Conference 2022*. 3009–3019.

[30] Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2020. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *arXiv preprint arXiv:2010.05874* (2020).

[31] Hong Wen, Jing Zhang, Fuyu Lv, Wentian Bao, Tianyi Wang, and Zulong Chen. 2021. Hierarchically modeling micro and macro behaviors via multi-task learning for conversion rate prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2187–2191.

[32] Hong Wen, Jing Zhang, Yuan Wang, Fuyu Lv, Wentian Bao, Quan Lin, and Keping Yang. 2020. Entire space multi-task modeling via post-click behavior decomposition for conversion rate prediction. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 2377–2386.

[33] Chenwang Wu, Defu Lian, Yong Ge, Zhihao Zhu, and Enhong Chen. 2023. Influence-Driven Data Poisoning for Robust Recommender Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[34] Xuyang Wu, Alessandro Magnani, Suthee Chaidaroon, Ajit Puthenputhussery, Ciya Liao, and Yi Fang. 2022. A Multi-task Learning Framework for Product Ranking with BERT. In *Proceedings of the ACM Web Conference 2022*. 493–501.

[35] Dongbo Xi, Zhen Chen, Peng Yan, Yinger Zhang, Yongchun Zhu, Fuzhen Zhuang, and Yu Chen. 2021. Modeling the sequential dependence among audience multi-step conversions with multi-task learning in targeted display advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 3745–3755.

[36] Ruobing Xie, Cheng Ling, Yalong Wang, Rui Wang, Feng Xia, and Leyu Lin. 2021. Deep feedback network for recommendation. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*. 2519–2525.

[37] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems* 33 (2020), 5824–5836.

[38] Qianqian Zhang, Xinru Liao, Quan Liu, Jian Xu, and Bo Zheng. 2022. Leaving No One Behind: A Multi-Scenario Multi-Task Meta Learning Approach for Advertiser Modeling. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1368–1376.

[39] Yuanxing Zhang, Langshi Chen, Siran Yang, Man Yuan, Huimin Yi, et al. 2022. PICASSO: Unleashing the Potential of GPU-centric Training for Wide-and-deep Recommender Systems. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE.

[40] Zhao-Yu Zhang, Xiang-Rong Sheng, Yujing Zhang, Biye Jiang, Shuguang Han, Hongbo Deng, and Bo Zheng. 2022. Towards Understanding the Overfitting Phenomenon of Deep Click-Through Rate Prediction Models. *arXiv preprint arXiv:2209.06053* (2022).

[41] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed Chi. 2019. Recommending what video to watch next: a multitask ranking system. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 43–51.

[42] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. Atrank: An attention-based user behavior modeling framework for recommendation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[43] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

[44] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1059–1068.

[45] Xinyu Zou, Zhi Hu, Yiming Zhao, Xuchu Ding, Zhongyi Liu, Chenliang Li, and Aixin Sun. 2022. Automatic Expert Selection for Multi-Scenario and Multi-Task Search. *arXiv preprint arXiv:2205.14321* (2022).