Multi-Faceted Hierarchical Multi-Task Learning for Recommender Systems

Junning Liu
Tencent PCG
Shenzhen, China
korchinliu@tencent.com

Xinjian Li
Tencent PCG
Shenzhen, China
xinjianli@tencent.com

Bo An
Nanyang Technological University
Singapore
boan@ntu.edu.sg

Zijie Xia
Tencent WXG
Shenzhen, China
zijiexia@tencent.com

Xu Wang
Tencent WXG
Shenzhen, China
caelanwang@tencent.com

ABSTRACT

There have been many studies on improving the efficiency of shared learning in Multi-Task Learning (MTL). Previous works focused on the "micro" sharing perspective for a small number of tasks, while in Recommender Systems (RS) and many other AI applications, we often need to model a large number of tasks. For example, when using MTL to model various user behaviors in RS, if we differentiate new users and new items from old ones, the number of tasks will increase exponentially with multidimensional relations. This work proposes a Multi-Faceted Hierarchical MTL model (MFH) that exploits the multidimensional task relations in large scale MTLs with a nested hierarchical tree structure. MFH maximizes the shared learning through multi-facets of sharing and improves the performance with heterogeneous task tower design. For the first time, MFH addresses the "macro" perspective of shared learning and defines a "switcher" structure to conceptualize the structures of macro shared learning. We evaluate MFH and SOTA models in a large industry video platform of 10 billion samples and hundreds of millions of monthly active users. Results show that MFH outperforms SOTA MTL models significantly in both offline and online evaluations across all user groups, especially remarkable for new users with an online increase of 9.1% in app time per user and 1.85% in next-day retention rate. MFH currently has been deployed in WeSee, Tencent News, QQ Little World and Tencent Video, several products of Tencent. MFH is especially beneficial to the cold-start problems in RS where new users and new items often suffer from a "local overfitting" phenomenon that we first formalize in this paper.

KEYWORDS

Recommendation Systems, Multi-task Learning

ACM Reference Format:

Junning Liu, Xinjian Li, Bo An, Zijie Xia, and Xu Wang. 2022. Multi-Faceted Hierarchical Multi-Task Learning for Recommender Systems. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557140

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557140

1 INTRODUCTION

Recommender systems (RS) are widely applied in online applications[1, 2, 5, 8, 28]. To better model user preference and increase satisfaction, there has been an increasing trend of using Multi-Task Learning (MTL) to simultaneously predict various user feedbacks. Compared to Single Task Learning (STL), MTL learns multiple tasks simultaneously in one model instead of one model per task. With the forthcoming era of immersive short videos such as TikTok, Reels and Triller, this trend has been accelerated as in full screen mode there are more parallel user feedbacks of more comparable importance to user experience (an example short-video App is shown in Fig. 1) whereas in the list page era there is a dominant impression-click behavior thread. The mainstream RS ranking scheme applies MTL to predict various specific user behaviors and predefined task labels as accurately as possible[17, 19, 20, 32], followed by a fusion model that aims to characterize the overall user satisfaction based on the outputs of the MTL model.

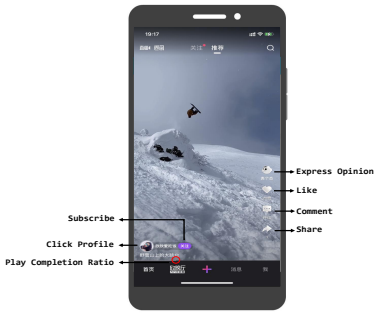


Figure 1: Immersive Short-Video App WeSee. Parallel user feedbacks such as subscribe, click profile, like, comment, share and play completion ratio are indicated on the screen.

Previous research on MTL focused on improving the shared learning with a small group of tasks [3, 19, 21, 25, 27]. These efforts span from the simplest hard parameter sharing [3] to sophisticated structures such as MMOE [19] and PLE [27]. On the other hand, there are often many more tasks beneficial to be modeled together in real world applications. For example in immersive short-video applications, users generate richer and more subtle feedbacks such

as time spent, likes, subscription, comments, sharing, and many predefined labels such as 3s skip (swipe away within 3 seconds) and 80% completion ratio. Joint learning these tasks with MTL has the benefit of better representation learning with more sufficient supervision and more shared learning among tasks. Furthermore, in RS and many other applications, large scale MTL is also frequently demanded when we have different value ranges of certain properties exhibiting very different correlation patterns with the predicted label. Despite that these feature regions may lack data samples, it is often important to improve the model's accuracy in those regions as they can be regions of higher business value, or regions important for the long-term value such as growth (e.g., new users, new content facing the cold-start situation) and content ecosystem. Normally, these property values are simply treated as sample features to the model. However, this often results in local overfitting in certain feature regions, especially when we have imbalanced and insufficient data samples. While we can split the different user/item/context regions into independent tasks, the number of tasks will exponentially increase as these are orthogonal dimensions that independently divide the samples.

Current MTL technologies do not scale well with more tasks, which explains why we observe the common industrial practice of employing 2 or even 3 MTL models in the ranking service to handle a total of 10-20 tasks where each MTL model normally handles 2-6 tasks. However, this practical approach is essentially a compromise between MTL and STL. Existing MTL learning structures such as MMOE and PLE handle well with two or a small number of tasks. With a large number of tasks, we will show later that simply plugging in more tasks with a flat branching connection using these structures can hardly yield significant improvement.

In this work, we propose a novel Multi-Faceted Hierarchical multi-task learning model (MFH) that aims at providing efficient multi-task learning for a large number of tasks through scalable cooperative learning design. The main contributions include:

- A new MTL model MFH is proposed to better address the challenge of scalable efficient multi-task learning with three major characteristics: Multi-Faceted, Hierarchical and Heterogeneous. The multifaceted and hierarchical design combined together introduces multidimensional implicit induction biases and results in much more efficient shared learning, thus greatly alleviating the local overfitting and the data scarcity issue for tasks with few samples. In addition, the MFH network is more heterogeneity-friendly and provides great flexibility for the model to better customize the tasks and generate further improvement.
- Extensive offline experiments are carried out to evaluate the effectiveness of MFH on industrial and public benchmark datasets. Online A/B test results in WeSee APP with tens of millions of DAU (Daily Active Users) also demonstrate the significant improvement of MFH over SOTA MTL models in real-world applications, with 2.14% increase in app-time per user and 0.19% increase in retention rate. The improvement is much more significant with new users as 9.10% increase in app-time per user and 1.85% increase in retention rate. Currently MFH has already been deployed in the online search and recommendation systems of several major products of Tencent: WeSee, Tencent News, QQ Little World and Tencent Video. The main source codes and additional technical results can be found in the Appendix.[1]
- *Micro* and *macro* perspectives of network design for cooperative learning in MTL are brought to attention for better understanding of the previous efforts on MTL and what is demanded for better scalability. Also a *local overfitting* phenomenon is introduced. These concepts are important to understand the core issues of scalable MTL.

2 RELATED WORK

In this section, we briefly review MTL models, applications of MTL and major related work on improving the shared learning efficiency in MTL. Multi-Task Learning [3, 31] is a general learning framework that improves the model generalization through cooperative learning between tasks. It explores the commonalities and differences between different tasks to facilitate the joint learning. MTL has been successfully applied to a wide range of applications, from RS [2, 8, 11, 27] to NLP [6, 26], and CV [10, 15, 22].

There are many studies on improving the shared learning efficiency in MTL. Shared-bottom [3] is the first simple structure for task sharing which cannot handle task conflicts well. Cross-Stitch Network [21] and Sluice Network [25] both learn static weights of linear combinations to fuse representations for different tasks selectively. MOE [14] first introduces expert modules and uses a gating network to fuse the expert outputs for upper task towers. MMOE [19] extends MOE to utilize task-specific gates to provide customized fusion for each task. M3oE [30] extends the customized gates in MMOE with multi-head gates. PLE [27] further improves the shared learning efficiency by differentiating task-specific experts and shared experts, and adopting a progressive routing mechanism. On top of PLE, MSSM [9] trains a field-level sparse connection to provide more flexible feature combination for different tasks, and replaces the gates with a matrix multiplied by a mask vector. Also focusing on the feature processing part of MTL, [12] uses multiple transformers to handle different user behavior sequences and applies MMOE on top, the "facet" there means type of user behaviors, different from aspect of task correlation as in this paper.

All works mentioned above focus on small groups of tasks and are often evaluated with 2 tasks. There still lacks model design exploiting the structure of the task relation graph for a large number of tasks. [16] tries to balance the training of task objectives through a Pareto efficient framework that dynamically adjust the loss weights of the tasks. ESMM [20] constructs a joint loss of CTCVR based on CTR and CVR's task relations. Recently ESMM is generalized in [29] to model the task relations with a Bayesian Graph and construct joint losses as in ESMM along the paths in the graph. However, many tasks cannot be modeled in a Bayesian graph, e.g., in many applications with a large number of mutually exclusive tasks splitting the user groups or item groups, modeling these tasks with a Bayesian graph is not an option. More importantly, the major difference is that [16], [29] and ESMM exploit task relations in loss design, instead of the learning network design as in this work, thus the benefit is on sample efficiency instead of shared learning efficiency. To the best of our knowledge, our work

[1] https://github.com/xinjianli6/MFH

is the first to address large groups of tasks and exploit semantic task relation structure in network design for MTL in RS.

Another thread of research applies neural architecture search (NAS) [33] and other AutoML methods to learn efficient MTL architectures [4, 18, 24]. However, as the searching cost is expensive, instead of general scope network optimization simplified assumptions are imposed on the search scope of network structures.

3 LARGE SCALE MTL MODEL EFFICIENCY

In this section, we first introduce the preliminary background and illustrate the problem setup, then we describe the baseline models, and lastly explain the challenges for large scale MTL.

3.1 MTL Ranking in an Industrial RS

We first introduce a real-world recommender system in WeSee, a short-video playing APP of Tencent, which serves tens of millions of users every day for immersive video watching experience. For a user request, the recommender system generates a recommended list of videos from a ten-million-scale candidate pool, then present the recommendations on the user's mobile screen one video at a time as shown in Fig. 1. Each recommended video will start autoplay and the user may take various actions such as keep watching, swipe, like, comment and share. In particular, if the user swipes up the current video, the system will show the next video from the list. When all videos of the recommended list are presented, a new request will be triggered to generate another list. The goal of the system is to recommend favorable videos that maximize user satisfaction which is normally quantified by total app time.

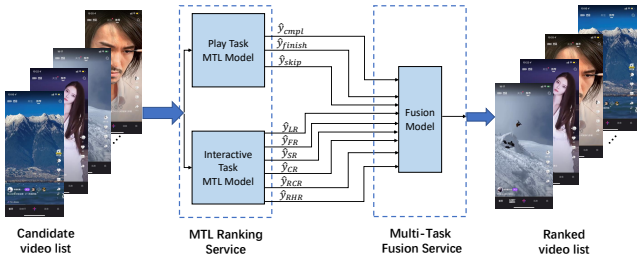


Figure 2: The MTL Ranking Framework

As in most industrial recommender systems, the recommender system adopts a two-stage design that contains two core processes [7], candidate recall and ranking. The ranking system aims to rank thousands of recalled candidate videos and select the top ones for presentation and is composed of two parts [27]: a deep MTL model and an Evolution Strategy (ES)/ Reinforcement Learning (RL) fusion model. The MTL model jointly outputs the likelihood of various user behaviors and the goal is to predict the concrete signals as accurately as possible. The fusion model then synthetically characterizes the overall user satisfaction score based on the concrete signals that the MTL model predicts. In our practice, there are two MTL models, a *Play Task Group MTL* and an *Interactive Task Group MTL* each dealing with a different group of tasks. The overall MTL ranking framework is shown in Fig. 2. We use the play task group as an example to study the MTL models.

The play task group MTL focuses on three important tasks that are highly related to the video watch time, i.e., *Play Completion Ratio* prediction, *Play Finish Rate* prediction, and *Play Skip Rate* prediction. For simplicity, we denote the three tasks by *Cmpl*, *Finish* and *Skip* respectively. Specifically, the Cmpl task is a regression task that predicts the completion ratio of a video view, defined as:

$$y_{cmpl} = \frac{watch\ time}{video\ length} \tag{1}$$

The watch time of a video view may often exceed the video length due to the rewatch and auto-replay nature of immersive short-video APPs, thus $y_{cmpl} \in [0, \infty)$. To handle exceptional cases, we truncate y_{cmpl} to ensure it is below a certain threshold.

The Finish task is a binary classification task that predicts the probability of watching a video to the end.

$$y_{finish} = \begin{cases} 1, & \text{if } watch\ time \geq video\ length \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The Skip task is a binary classification task that predicts the probability of quick skipping a video within a short time.

$$y_{skip} = \begin{cases} 1, & \text{if } watch\ time \leq c \text{ seconds} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where c is a small constant number.

The three tasks model users' watching behaviors from slightly different perspectives. The Cmpl task reflects a user's commitment to a video continuously, while the Finish and Skip tasks focus on modeling users' positive and negative viewing experience, respectively. The three tasks are jointly learned within one MTL model.

There are three different groups of users: *new users*, *low-activity users* and *high-activity users*. The activity level is determined according to total video watch time of the user. High-activity users are users with more than 60 min watch time. Naturally there are far fewer data samples for new/low-activity users than for high-activity users. New users have higher business value weights.

In general, we can easily have an MTL problem where certain properties of the samples support orthogonal splitting of the original tasks. For example, different user groups, different item groups, and different contexts such as time, location, and moving patterns all support dividing the original tasks into many more tasks.

3.1.1 Local Overfitting. In our practice a *local overfitting* phenomenon is observed. By local overfitting, we mean that the model is overfitted in part of the input space or the degree of overfitting in part of the input space is much more serious than other regions. In the MTL ranking of our system, existing models exhibit overfitting on all tasks for new/low-activity users, in contrast to high-activity users. Intuitively, this is because the training samples of new/low-activity users are much fewer than that of high-activity users and the model parameters trained with such imbalanced data are dominated by the data pattern of high-activity users. Thus the predictions on new/low-activity users are negatively affected by high-activity users, leading to unsatisfactory recommendation results and user experience. However, cold-start, i.e., addressing the recommendation for new users or new items, is a problem of great business value for retention and growth. Models are often developed separately for cold-start in industrial practice. This is far from an ideal approach as the cold-start model still suffers from few data samples, additional training/serving cost, and more importantly

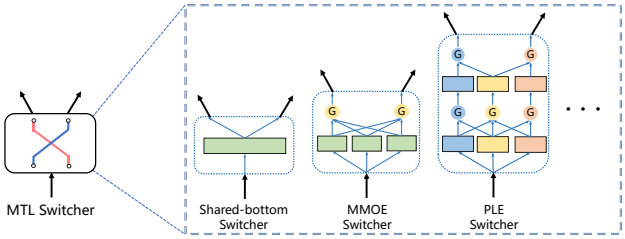


Figure 3: The MTL Switcher and Example Instantiations

the common patterns shared between different user groups are not transferred to benefit the minority groups.

Furthermore, this local overfitting phenomenon is not just limited to the cold-start problem of new users. In general, any feature regions that exhibit different patterns from the main regions and have much less data samples may suffer from local overfitting, e.g., new items, new users and minority contexts. Also local overfitting overlaps with the fairness issue in RS as broadly studied in [23] in such a way that local overfitting can be one of the technical issues that caused fairness problems across different group of users.

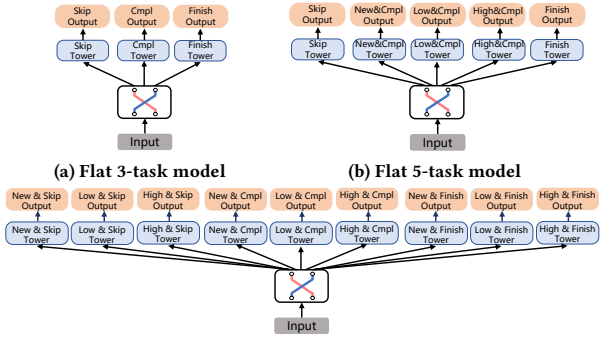
3.2 Micro Perspective Cooperative Learning

In the past years, most MTL works in RS focus on task groups of small size. Previous MTL models mainly study the micro perspective of shared learning and do not address large scale MTL. Next, we formally introduce the micro level cooperative learning with a switcher concept, then describe the baseline models using previous SOTA methods in our problem setup.

3.2.1 Micro level cooperative learning structure - MTL Switcher. To illustrate the micro level coordinated learning structures in MTL, we introduce a concept called *switcher* in deep learning models. As depicted in Fig. 3, a switcher is a neural architecture that takes one input and branching out multiple ($\geq$ 2) latent outputs. The input of the switcher can be any type of features, embeddings or intermediate latent representations in the network, and its outputs are some latent representations that will be fed into upper-level networks such as any hidden layer in the network or particularly specific task tower in deep MTL models. As discussed in [27], MTL needs to jointly address representation learning and information routing. Abstractly, switchers can be used to deal with the micro level cooperative learning of diverting one input to multiple intermediate latent outputs. As we see in Fig. 3, MTL structures such as shared-bottom, MMOE, PLE are merely switchers of different types.[2] Previous MTL research has been focusing on switcher innovations improving the micro level cooperative learning efficiency and does not work on MTLs with a large number of tasks.

3.2.2 Baseline Models. There is a dilemma for our problem: the minority user groups of new users and low-activity users share the same task tower with high-activity users. Thus the network parameters will be mostly under the influence of high-activity users given the imbalanced samples, yielding unsatisfying performance for new/low-activity users. On the other hand, if we separate different user groups into different models, we will lose the opportunity of transferring the knowledge shared between different user groups. A natural idea to address the cold start and local overfitting issues is to split original task's feature regions into independent tasks with the overfitting regions as separated tasks. For example, we can split the Cmpl task into three new tasks: New&Cmpl, Low&Cmpl, High&Cmpl, representing Cmpl for new users, Cmpl for low-activity users, and Cmpl for high-activity users respectively. This will give the overfitting regions more customized optimization independently without loss of accuracy on the major regions. We present a few task splitting variations in the baseline models.

Baseline 3-task Model: Figure 4a illustrates a baseline flat 3-task model with one switcher and three towers that correspond to the three tasks Skip, Cmpl, and Finish, respectively. Note that the baseline 3-task model may have different versions, depending on the specific switcher architecture it adopts.[3] In our case when the switcher upgrades from simple to more complex SOTA ones, i.e., Shared-bottom → MMOE → PLE, corresponding performance improvements are observed. Thus the baseline 3-task model uses PLE as the switcher structure.



(a) Flat 3-task model **(b) Flat 5-task model**

(c) Flat 9-task model

Figure 4: Baseline MTL models

Flat 5-task and 9-task Models: To improve the performance on the new user and low-activity user groups, we attempt to further divide a prediction task into three sub-tasks according to the division of user groups, i.e., prediction on new users, prediction on low-activity users, and prediction on high-activity users. We implement a partially-divided baseline flat 5-task model (Fig. 4b) by dividing only the Cmpl task into three sub tasks New&Cmpl, Low&Cmpl and High&Cmpl, and a fully divided baseline flat 9-task model (Fig. 4c) by dividing all tasks in the same way.

3.3 Challenges in Large Scale MTL

Despite the different SOTA networks we tried in the baseline models, the cold start and local overfitting problems still largely remain unsolved as shown in later experiments. In this subsection we discuss the necessity of macro-perspective network design for shared learning in large scale MTLs and the main challenges.

3.3.1 Shared Learning efficiency in Large Scale MTL. With a large number of unbalanced tasks that have multi-aspects of correlations, is the traditional MTL structure well prepared to scale efficiently? This is a natural question to ask.

With more tasks, the traditional MTL models are incapable of providing sufficient shared learning efficiency. Offline evaluation

[2]Note that in the shared-bottom case the bottom layer could downgrade to zero layer as the hard sharing scheme which simply shares the input directly to the branches.

[3]In the rest of this paper, we will describe other MTL models using abstract switchers in the same way.

results of Table 1 in Section 5 show that the flat 5-task and 9-task models make limited improvement compared to baseline 3-task model in terms of prediction accuracy. In other words, simply splitting the original 3 tasks into 5 tasks or 9 tasks and plugging in a flat sharing structure with the SOTA MTL models of PLE or MMOE can hardly generate significant performance gains, while in the case of small group such as 1 or 2 tasks, modeling more tasks normally generates performance gains. This shows that the majority dominance over minority is beyond the scope of micro level switchers and the performance gain of modeling more tasks in MTL with traditional methods is diminishing with more tasks.

3.3.2 Macro Level Cooperative Learning Structure. The macro learning structure concerns with the macro scale information sharing among tasks. One straightforward macro learning structure will be a flat branching structure as in all baseline models shown in Fig. 4, an opposite alternative can be a chain structure like the asymmetric sharing [27]. For the same macro level structure, any micro level switcher can be used for local diverting, i.e., differences in switchers do not make any difference on macro learning structures.

Large scale MTLs raise the importance of macro level cooperative learning structures. There are no common baselines for macro level network design. And the simple flat macro structure does not perform so well in large scale MTLs for several reasons. First, there are a large number of tasks with multi-aspects of correlations, whereas traditional methods do not exploit the multi-aspects of correlation in the macro level structure design to maximize the extent of shared learning; Second, for large scale MTLs, traditional networks with the simple flat sharing structure have difficulty in directly extracting out all tasks' latent information demands without any hierarchy. We need new macro structures that can increase the depth of shared learning; Thirdly, in large scale MTLs, the tasks are often highly unbalanced in terms of both data samples and the complexity of feature-label patterns. Traditional MTLs with homogeneous task tower structures are not efficient in such scenarios.

Thus, large scale MTLs demands better innovative macro structure design for efficient shared learning. This work focuses on the macro perspective of shared learning. To the best of our knowledge, we are the first to differentiate the micro and macro perspective of shared learning and the first to study the macro perspective network design for shared learning in MTL.

4 MULTI-FACETED HIERARCHICAL MTL

This section presents our solution model - MFH. We first introduce the notion of "facet" for tasks, then present a hierarchical MTL model and the MFH model. For the sake of readability, we employ the 9-task play task group MTL problem as an example to describe our models and later generalize to a general N-faceted MTL.

4.1 Multi-Faceted Hierarchical MTL (MFH)

4.1.1 Facets of Tasks. We introduce a concept of *facet* for tasks. Facets are orthogonal dimensions that every task has. There are several partitions for each facet that can divide tasks into groups. For example in the 9-task problem setup, each task simultaneously has two *facets*, i.e., *user behavior facet* = {Cmpl, Finish, Skip} and *user group facet* = {New, Low-activity, High-activity}, where each facet contains three *partitions*.

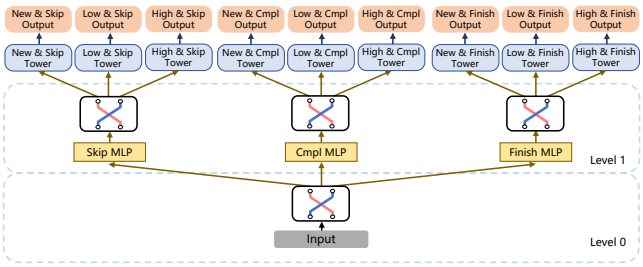


Figure 5: Hierarchical MTL (H-MTL) Model

The combination of any two partitions from the two facets defines a specific task. Facets provide prior structures for the correlation between tasks. There are correlations along each facet's aspect and tasks sharing common facet partitions have stronger correlations. For example, in the 9-task MTL task New&Cmpl shares the same user group facet partition of new users with tasks New&Finish and New&Skip, and at the same time shares the same user behaviour facet of play completion ratio with tasks Low&Cmpl and High&Cmpl. In general, the number of facets could be three or even more, e.g., the popularity of the video or the video length can each be another new facet. The number of tasks exponentially increases as more facets are introduced. Thus, not only limited to the cold-start scenarios, this type of multifaceted multi-task problem is actually commonly seen in industrial practice and poses new challenges on how to scale the cooperative learning in the multifacet MTL setup from the macro perspective.

4.1.2 Hierarchical MTL. In this subsection, we introduce a Hierarchical MTL (H-MTL) model for the 9-task problem. As depicted in Fig. 5, H-MTL utilizes a two-level tree architecture to model the task relationships in both facets and share the facet latent representations between tasks in a hierarchical fashion. At level 0, a switcher is adopted to learn the task relationship in the user behavior facet based on the input features, which connects to three MLPs (Multilayer Perceptrons) at level 1 that correspond to three partitions of the user behavior facet: Skip, Cmpl and Finish, respectively. Each MLP outputs a hidden representation to feed an MTL switcher which learns the task relationship in the user group facet conditioned on a particular partition of the user behavior facet, and connects to three task tower networks that correspond to the combination of this user behavior and one of the three user groups (i.e., new, low-activity and high-activity). Each task tower network concentrates on the corresponding task and predicts the final score for that task. Formally, the output of a specific task, e.g., Task New&Skip, can be abstractly formulated as:

$$Output_{New\&Skip} = Tower_{New\&Skip}(Switcher_{Skip}^{New\&Skip}(MLP_{Skip}(Switcher_{Input}^{Skip}(Input)))) \quad (4)$$

where $Switcher_X^Y$ indicates the corresponding output of $Switcher_X$ for (hidden) Task Y, and in general the MLP can downgrade to zero layer in which case the lower level switcher will feed directly to the upper level switchers.

From the macro cooperative learning perspective, H-MTL avoids to branch out directly from the input to all tasks as the baseline flat 9-task model does. Instead, it adopts a hierarchical structure with multi-level tree sharing among tasks. At each level, switchers

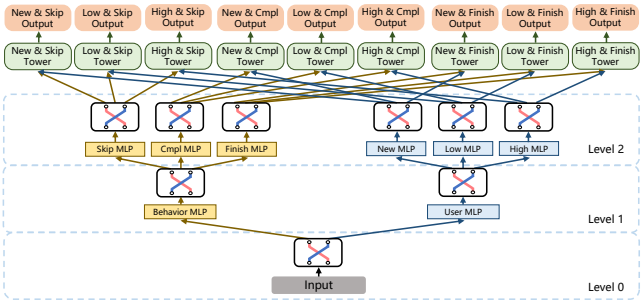


Figure 6: Multi-Faceted Hierarchical MTL Model(MFH)

are used to branch out semantic representations for the upper level sub-trees. In general, the tree starts from level 0 to level $k \leq N-1$ when we have N facets. Different permutations of the facets form different trees. For example, we can divide by user groups first then further divide by user behaviours.

4.1.3 Multi-Faceted Hierarchical MTL (MFH). Although the H-MTL model captures the hierarchical task relationships, we have to choose a particular tree corresponding to one specific permutation of the facets. However, facets are important explicit dimensions that reflect the task correlations independently. The tasks have different correlation groups along each facet simultaneously.

To further improve the adequacy of information sharing and cooperative learning among the tasks, we propose a more comprehensive model, named Multi-Faceted Hierarchical MTL (MFH). MFH is essentially composed of multiple H-MTL trees that are nested together. With the 9 task problem as an example, as shown in Fig. 6, at level 0, the switcher network learns the inter-facet task relationship between two facets, and branches out to the two facets' MLPs at level 1. The upper-level structures of MFH can be simply regarded as the combination of two variants of the H-MTL model. In particular, each tower network combines the hidden outputs from two different paths connected to the input, and outputs the predicted score for a specific task. For example, the output of Task New&Skip can be abstractly formulated as:

$$
\begin{aligned}
Output_{New\&Skip} = Tower_{New\&Skip}(\\
Switcher^{New\&Skip}_{Skip}(MLP_{Skip}(Switcher^{Skip}_{Behavior}(\\
MLP_{Behavior}(Switcher^{Behavior}_{Input}(Input))))) + \\
Switcher^{New\&Skip}_{New}(MLP_{New}(Switcher^{New}_{Group}(\\
MLP_{Group}(Switcher^{Group}_{Input}(Input)))))) \qquad (5)
\end{aligned}
$$

where $Switcher^{Y}_{X}$ indicates the corresponding output of $Switcher_X$ for (hidden) Task Y, and + denotes a generic combination operation that can be linear combination, concatenation or even applying gate/attention net to fuse, we use linear combination in this work.

Compared to the H-MTL model, MFH further improves the learning efficiency by firstly modeling three-fold task relationship by extending to three levels of MTL switchers, i.e., the inter-facet relationship, the first-order intra-facet relationship, and the second-order intra-facet relationship in the context of a particular partition of another facet. Second, it enables each task to share semantic information simultaneously with multiple sets of strong correlated tasks according to the shared facet partitions. With MFH, we can

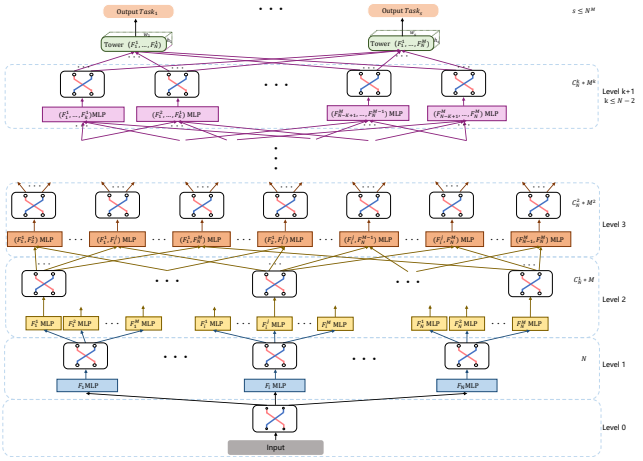


Figure 7: Generalized Version of the MFH Model

expand to any number of orthogonal trees of different facet permutations and nest the intermediate and leaf nodes as they cross. This maximizes the representation learning through multi-dimensional intersecting paths that support more shared learning.

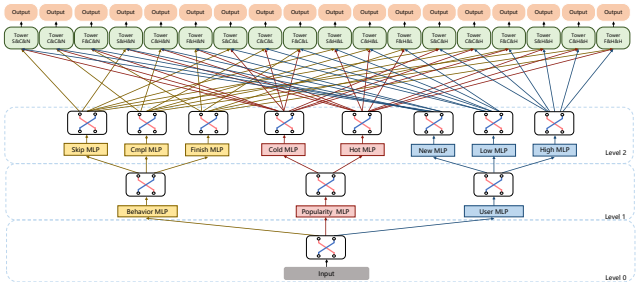


Figure 8: 3 Facets MFH Model

4.2 Generalize to N-faceted Multi-Task Setting

In this subsection, we generalize the MFH model to a generic N-faceted multi-task problem setup. Let F_i denote the i-th facet, and F_i^j denote the j-th partition of facet F_i. Facet F_i contains $M_i > 1$ partitions. For simplicity of presentation, we assume that all facets have M partitions. Then, there would be in total $s \leq M^N$ tasks in this multi-task setting. Each task is associated with one element of the Cartesian product of the N facets, denoted by a N-tuple, $(F_1^{j_1}, ..., F_N^{j_N})$, where $j_1, ..., j_N \in \{1, ..., M\}$. Given these definitions, we illustrate the generalized version of MFH model in Fig. 7.

Specifically, the level 0 switcher (i.e., the root node) expands the N facets MLPs F_i MLP, i from 1 to N. The level 1 switchers expand each facet to its M partition MLPs. In general, for any $k < N-1$, level k contains $C_N^{k-1} \times M^{k-1}$ switchers and they expand to $C_N^k \times M^k$ MLPs and switchers at level $k+1$. Each MLP at level $k+1$ has a unique code formed by a combination of k unique facets each with a specific partition. Upper MLPs and lower MLPs are connected through the lower level switchers if the upper level MLP's code contains the lower level MLP's code as a subcode. In general, we can choose to expand the multi-facet network to any level of k, $1 < k < N-1$, then connecting directly to the $s \leq M^N$ towers for output tasks. We can have at most M^N tasks but do not necessarily

need to split all tasks if there is not high business value or special pattern for the considered input regions.

Fig. 8 shows a 3-facet MFH of 18 tasks for the Play Task Group with an additional facet of video popularity, differentiating new items and old ones. This jointly models the complete cold-start problem for both new users and new items.

4.3 Heterogeneity of MFH

MFH is more heterogeneity-friendly than flat MTL. In Fig. 7, MFH's task towers and MLPs can all be designed heterogeneously. The size of the task towers can be customized to better fit a special task. For example, we can use smaller size MLP layers for task towers with less training samples. We can do this for the intermediate MLPs that correspond to tasks of less data samples as well. MFH is more flexible on heterogeneity as the common shared root is thinner and there are various granularities of sharing that can be customized to be heterogeneous. In addition to the structure, different tasks can also have customized input features only available for themselves.

Paired up with a generic fusion algorithm, MFH's heterogeneity can serve as a component of a universal ranking that unifies the ranking of heterogeneous candidates in a broader context, e.g., mixed ranking with video contents, ads and live broadcast.

5 EXPERIMENTS

Offline and online experiments are performed on both a large-scale industrial recommender system and a public benchmark dataset to evaluate the effectiveness of the proposed models.

Table 1: Performance on Play Tasks. The improvements of *MFH* over *baseline 3-task* are shown in brackets.

Models	PCR MSE	PFR AUC	PSR AUC
New User Group			
baseline 3-task	.5167	.7784	.7968
flat 5-task	.5184	.7770	.7973
flat 9-task	.5162	.7799	.7965
H-MTL 9-task	.5156	.7801	.7989
MFH 9-task	**.5138**$_{(-0.56\%)}$	**.7813**$_{(+0.37\%)}$	**.8003**$_{(+0.44\%)}$
Low-Activity User Group			
baseline 3-task	.4807	.7977	.8179
flat 5-task	.4832	.7929	.8174
flat 9-task	.4796	.7990	.8178
H-MTL 9-task	.4792	.8001	.8186
MFH 9-task	**.4776**$_{(-0.64\%)}$	**.8006**$_{(+0.36\%)}$	**.8198**$_{(+0.23\%)}$
High-Activity User Group			
baseline 3-task	.4085	.8177	.8548
flat 5-task	.4087	.8170	.8552
flat 9-task	.4079	.8181	.8536
H-MTL 9-task	.4073	.8183	.8549
MFH 9-task	**.4070**$_{(-0.37\%)}$	**.8199**$_{(+0.27\%)}$	**.8563** $_{(+0.18\%)}$

5.1 Real World Deployment and Evaluation

The proposed models are evaluated on Tencent's large-scale short video App WeSee which serves hundreds of millions monthly active users for immersive video watching.

5.1.1 Real World Deployment. We first deploy the MFH model in the RS ranking system of WeSee. An MFH 9-task model for the play task MTL and an MFH 12-task model for the interactive task MTL are deployed in the MTL ranking service of the ranking framework as shown in Fig. 2. The two MFH models jointly predict users' 9 behaviors, then a reinforcement learning model fuses the MTL outputs into one score for the final ranking. After validation, MFH has currently been deployed to Tencent News RS, QQ Little World RS and Tencent Video Search, generating extensive business value.

5.1.2 Offline Evaluations. **Dataset**: We collect an industrial dataset through sampling user logs from WeSee APP during a few consecutive days. There are 10 billion samples in the dataset. In addition to labels PCR (Play Completion Ratio), PFR (PLAY Finish Rate), PSR (Play Skip Rate) as mentioned before, there are also explicit user feedback labels LR (Like Rate), FR (Follow Rate), CMR (Comment Rate), SR (Share Rate), RCR (Read Comment Rate), RHR (Read Homepage Rate). The sample distribution on the user groups is 1.57% of new user group, 11.46% of low-activity user group and 86.97% of high-activity user group.

Learning Tasks: There are two MTL learning models to serve the online ranking. A play task group MTL that jointly predicts PCR, PFR and PSR, 3 tasks mostly related to the video playing process. Another interactive task group MTL that jointly predicts LR, FR, CMR, SR, RCR and RHR, the explicit user feedback behaviours.

MTL Models: For the play tasks MTL, we evaluate the baseline 3-task MTL (Fig. 4a), flat 5-task MTL (Fig. 4b), flat 9-task MTL (Fig. 4c), H-MTL 9-task (Fig. 5) and MFH 9-task MTL (Fig. 6). For the interactive tasks MTL, we evaluate the baseline 6-task, H-MTL 12-task and MFH 12-task models.

Since the focus in this work is the macro level task coordination learning structures, for each MTL model, we tried different MTL switchers for the micro level shared learning such as hard sharing, MMOE, CGC [27] and PLE and regard the performance of the best switcher choice as the performance of the corresponding macro level MTL structure. As a result, we adopt shared-bottom for level 0 switcher, PLE for level 1 switchers, CGC for level 2 switchers in H-MTL and MFH, and adopt PLE for switchers in rest of the models.

SOTA Baselines: Note that for both the play tasks MTL and the interactive tasks MTL, the SOTA models to be compared with MFH are the baseline flat models. As we explained earlier there are two aspects of the model design of our problem: for the micro level design, we adopt the SOTA switcher network PLE for all the baseline models, including the flat 9-task MTL and the flat 6-task MTL; for the macro level network design, there are few previous research on this aspect and the flat branching structure has been used without an explicit recognition as a macro design choice, thus the flat structure is in fact the current SOTA for macro level choice.

Experiment Setup: In the experiment, PCR prediction is a regression task trained with MSE loss (Mean Squared Error) and evaluated with MSE; tasks modeling other actions are all binary classification tasks trained with cross-entropy loss and evaluated with AUC. Samples in the first 14 days are used for training and the rest of samples for testing. For the task towers, we adopt a two-layer MLP network with RELU activation and heterogeneous hidden layer sizes: (128,64) for high-activity user tasks and (64, 32) for new and low-activity user tasks. Since the new user's user id does not contain much meaningful information, we remove the id feature for new users. The experts in the switcher are implemented with a multi-layer MLP and tuned on the following model-specific hyper-parameters: number of shared layers, number of experts.

Table 2: Performance on Interactive Tasks

User Group	Models	LR AUC	FR AUC	CR AUC	SR AUC	RCR AUC	RHR AUC
New & Low-Activity	baseline 6-task	.8821	.9453	.9495	.9167	.9113	.8723
	H-MTL 12-task	.8851$_{(+0.34\%)}$	.9455$_{(+0.02\%)}$	.9541$_{(+0.48\%)}$	.9208$_{(+0.45\%)}$	.9114$_{(+0.01\%)}$	.8800$_{(+0.88\%)}$
	MFH 12-task	**.8866$_{(+0.51\%)}$**	**.9461$_{(+0.08\%)}$**	**.9558$_{(+0.66\%)}$**	**.9239$_{(+0.79\%)}$**	**.9116$_{(+0.03\%)}$**	**.8882$_{(+1.82\%)}$**
High-Activity	baseline 6-task	.9195	.9484	.9722	.9521	.9417	.9243
	H-MTL 12-task	.9211$_{(+0.17\%)}$	.9515$_{(+0.33\%)}$	.9732$_{(+0.10\%)}$	.9560$_{(+0.41\%)}$	.9425$_{(+0.08\%)}$	.9305$_{(+0.67\%)}$
	MFH 12-task	**.9221$_{(+0.28\%)}$**	**.9531$_{(+0.50\%)}$**	**.9758$_{(+0.37\%)}$**	**.9587$_{(+0.70\%)}$**	**.9434$_{(+0.18\%)}$**	**.9327$_{(+0.91\%)}$**

Table 3: Online Experiment Evaluation

Models	apptime	2nd-day retention
Baseline Flat MTLs	-	-
MFH	+2.14%	+0.19%
MFH on new users	+9.1%	+1.85%

Table 4: Performance of 3-facets MFH *vs.* 2-facets MFH on Play Tasks for high-activity users. The relative improvements are shown in brackets.

Cold Start Items

Models	PCR MSE	PFR AUC	PSR AUC
MFH 9-task (2facets)	.2387	.8471	.8060
MFH 12-task (3facets)	**.2363$_{(-1.01\%)}$**	**.8506$_{(+0.41\%)}$**	**.8091$_{(+0.38\%)}$**

Non-Cold Start Items

Models	PCR MSE	PFR AUC	PSR AUC
MFH 9-task (2facets)	.3896	.8188	.8473
MFH-12task (3facets)	**.3892$_{(-0.10\%)}$**	**.8199$_{(+0.13\%)}$**	**.8475$_{(+0.02\%)}$**

Evaluation with Play Task Models: Table 1 illustrates the experiment results and we mark the best performance in bold. It is shown that H-MTL significantly outperforms all flat task models (*baseline 3-task*, *flat 5-task* and *flat 9-task*) in all tasks and all user divisions. With the flat task shared learning structure, introducing more tasks produces slight improvement, but much insufficient compared to the improvement H-MTL and MFH generate. MFH further significantly outperforms H-MTL in all tasks and user divisions. Of the improvement H-MTL and MFH generate, it is much more significant on the new users than on active users.

We also compare the performance between MFH 9-task and flat 9-task in Table 1, as both models have the same 9 tasks and the only difference is MFH *vs.* flat on macro shared learning structure. MFH 9-task outperforms flat 9-task on all tasks: -0.46% MSE on New&Cmpl, -0.42% MSE on Low&Cmpl, -0.22% MSE on High&Cmpl; +0.18% AUC on New&Finish, +0.2% AUC on Low&Finish, +0.22% AUC on High&Finish; +0.48% AUC on New&Skip, +0.24% AUC on Low&Skip, +0.32% AUC on High&Skip. This shows MFH manifests significant performance improvement over baseline flat sharing.

Evaluation with Interactive Task Models: The interactive tasks include LR (Like Rate), FR (Follow Rate), CR (Comment Rate), SR (Share Rate), RCR (Read Comment Rate) and RHR (Read Homepage Rate) 6 tasks. For interactive task MTL, we merge the new user group and the low-activity user group into one group thus have 12 tasks in total. As in Table 2, H-MTL and MFH achieve significant improvement over the baseline model on all tasks of all user groups.

5.1.3 Online Evaluation. Online experiments are also conducted for three weeks. The baseline uses the flat 3-task model for the play tasks MTL and a flat 6-task model for the interactive tasks MTL, the experiment group adopts an MFH 9-task model and an MFH 12-task model accordingly. Both the experiment and the control group have the same RL fusion model adapted to the corresponding MTL ranking models. Table 3 shows the significant improvement of MFH and it is worth noting that MFH achieves a remarkable increase of **+9.1%** apptime per user on new users.

5.1.4 Lessons Learned. To achieve the full potential of a new algorithm in real world systems, we need to consider the whole system's coordination and tune the subtle details patiently. For example, to avoid a sudden performance gap for new users or low-activity users when they migrate groups, we include new user samples in all groups' task training and low-activity user samples in high-activity user tasks training while in serving only do inference through the corresponding user group tasks. Thus the performance is smoother when users change groups, e.g., new users become low-activity users. As new users and low-activity users are far less than high-activity users, the larger user groups' training is not negatively affected. Another example is that the MTL ranking service and the MTF service need to work together to finish the final ranking, we need to patiently adapt the RL MTF model to the MTL model change to achieve the full benefit of the MFH model.

5.1.5 Ablation Study. Extensive ablation experiments are conducted according to the three major innovative designs of Multi-Faceted, hierarchical and heterogeneity. We first evaluate the ablation effect of removing Multi-Faceted design, as well as removing both Multi-faceted and hierarchy design. Comparing H-MTL 9-task model with MFH 9-task model when we remove Multi-Faceted design, significant performance decreases are observed for all tasks unanimously. Comparing flat 9-task model with MFH 9-task model when we remove both Multi-Faceted and hierarchical design, even more significant performance decreases are observed for all tasks unanimously. Please refer to Table 7 of Appendix for details.

Next we experiment on the ablation of the hierarchical design. As MFH can not remove the hierarchical design independent of the Multi-Faceted design, we observe the performance gap between flat 9-task model and H-MTL 9-task model. Performance decrease of the ablation of hierarchical design is observed with details in Table 8 of the Appendix.

We also conduct experiments ablating the heterogeneity of MFH. The results are shown in Table 9 of the Appendix where heter-abl stands for the ablation experiment where we remove all heterogeneous designs and adopt a task tower network of two-layer MLP of size (128,64) as the high-activity user group tasks' settings for all user groups, instead of heterogeneous task towers. Compared to the baseline model, the ablation of heterogeneity decreases the performance of the tasks to varying degrees as shown in Table 9.

5.1.6 More Tasks and Facets Evaluation: 3-facets MFH. We conduct offline experiment to evaluate the 3-facets MFH model as shown in Fig. 8 of Section 4.2. As in practice new items are normally not

Table 5: Performance on Ali-CCP Dataset

Micro Structure (Switchers)	Macro Structure	New User Group		Low-Activity User Group		High-Activity User Group	
		ctr AUC	cvr AUC	ctr AUC	cvr AUC	ctr AUC	cvr AUC
MMOE	Flat (2-task)	.6127	.6462	.6130	.6355	.6119	.6421
	Flat (6-task)	.6133$_{(+0.10\%)}$	.6462$_{(+0.00\%)}$	.6135$_{(+0.08\%)}$	.6417$_{(+0.98\%)}$	.6131$_{(+0.20\%)}$	.6438$_{(+0.26\%)}$
	H-MTL (6-task)	.6140$_{(+0.21\%)}$	.6472$_{(+0.15\%)}$	.6137$_{(+0.11\%)}$	.6440$_{(+1.34\%)}$	.6133$_{(+0.23\%)}$	.6446$_{(+0.39\%)}$
	MFH-MTL (6-task)	**.6148$_{(+0.34\%)}$**	**.6496$_{(+0.53\%)}$**	**.6144$_{(+0.23\%)}$**	**.6482$_{(+2.00\%)}$**	**.6138$_{(+0.31\%)}$**	**.6457$_{(+0.56\%)}$**
PLE	Flat (2-task)	.6139	.6457	.6151	.6408	.6120	.6439
	Flat (6-task)	.6140$_{(+0.02\%)}$	.6466$_{(+0.14\%)}$	.6160$_{(+0.15\%)}$	.6461$_{(+0.83\%)}$	.6135$_{(+0.25\%)}$	.6446$_{(+0.11\%)}$
	H-MTL (6-task)	.6144$_{(+0.08\%)}$	.6488$_{(+0.48\%)}$	.6165$_{(+0.23\%)}$	.6517$_{(+1.70\%)}$	.6137$_{(+0.28\%)}$	.6450$_{(+0.17\%)}$
	MFH-MTL (6-task)	**.6154$_{(+0.24\%)}$**	**.6494$_{(+0.57\%)}$**	**.6179$_{(+0.46\%)}$**	**.6544$_{(+2.12\%)}$**	**.6144$_{(+0.39\%)}$**	**.6457$_{(+0.28\%)}$**

Table 6: Performance on MovieLens Dataset

Micro Structure (Switchers)	Macro Structure	New User Group		Low-Activity User Group		High-Activity User Group	
		ctr AUC	rate MSE	ctr AUC	rate MSE	ctr AUC	rate MSE
MMOE	Flat (2-task)	.7035	.9670	.7194	.9145	.7373	.8082
	Flat (6-task)	.7044$_{(+0.12\%)}$	.9627$_{(-0.44\%)}$	.7194$_{(+0.01\%)}$	.9113$_{(-0.35\%)}$	.7370$_{(-0.03\%)}$	.8040$_{(-0.53\%)}$
	H-MTL (6-task)	.7050$_{(+0.21\%)}$	.9537$_{(-1.37\%)}$	.7205$_{(+0.15\%)}$	.9018$_{(-1.39\%)}$	.7415$_{(+0.57\%)}$	.8068$_{(-0.18\%)}$
	MFH-MTL (6-task)	**.7072$_{(+0.52\%)}$**	**.9517$_{(-1.58\%)}$**	**.7232$_{(+0.53\%)}$**	**.8988$_{(-1.72\%)}$**	**.7422$_{(+0.67\%)}$**	**.7960$_{(-1.52\%)}$**
PLE	Flat (2-task)	.7049	.9604	.7209	.9116	.7390	.7957
	Flat (6-task)	.7065$_{(+0.24\%)}$	.9581$_{(-0.24\%)}$	.7213$_{(+0.06\%)}$	.9102$_{(-0.16\%)}$	.7412$_{(+0.30\%)}$	.7958$_{(+0.02\%)}$
	H-MTL (6-task)	.7091$_{(+0.60\%)}$	.9576$_{(-0.29\%)}$	.7245$_{(+0.50\%)}$	.9047$_{(-0.76\%)}$	.7440$_{(+0.68\%)}$	.7925$_{(-0.39\%)}$
	MFH-MTL (6-task)	**.7098$_{(+0.71\%)}$**	**.9476$_{(-1.33\%)}$**	**.7260$_{(+0.72\%)}$**	**.8926$_{(-2.09\%)}$**	**.7468$_{(+1.06\%)}$**	**.7898$_{(-0.74\%)}$**

served to new users and low-activity users for a double blind cold start with concerns on user experience, we only serve cold start new items to high-activity usersThus instead of an MFH 18-task as shown in Fig 8, we have an MFH 12-task of 3 facets, with the high-activity user group's 3 tasks divided into 6 tasks, 3 for cold start content items and 3 for non-cold start items.

The same dataset as above is used to evaluate the offline performance of MFH 12-task *vs.* the 2 facets MFH 9 tasks. 3 facets MFH 12-task model achieves similar performance on new users and low-activity users compared to the 2 facets MFH 9-task model and achieves significant improvement on high-activity user group where the 3rd facet is applied. As shown in Table 4, the 3 facets MFH 12-task model outperforms the 2 facets MFH 9-task model on all tasks for the high-activity users, for both the non-cold start items and the cold start items. A great reduction of **-1.01%** on *Cmpl* task's MSE loss are observed for cold start new items, which is a remarkably significant improvement for content cold start as normally a 0.1% increase of AUC or MSE already generates online metric improvement significant enough to be observed in A/B testing.

5.2 Evaluation on Public Dataset

5.2.1 Ali-CCP Data. Ali-CCP (Alibaba Click and Conversion Prediction) Dataset[4] is a public dataset extracted from Taobao's Recommender System. The dataset includes 84 million data samples equally divided into training set and testing set, which contain 3.4 million clicks and 18 thousand conversions. CTR (Click Through Rate) and CVR (Conversion Rate) are two tasks modeling the user actions of click and purchase in the dataset. For users in the dataset, we divide the users into three groups: new users with 0-15 video views (vv), low-activity users with 15-50 vv, and high-activity users with more than 50vv, containing 22%, 18% and 60% users respectively. Since there are a CTR task and a CVR task for every group, we have a total of 6 tasks.

[4]https://tianchi.aliyun.com/dataset/dataDetail?dataId=408

5.2.2 MovieLens Data. MovieLens 20M dataset[13] contains around 20 million ratings of 27,278 movies by 138,493 users.We divide the users into three groups: new users with 20-36 vv, low-activity users with 37-99 vv, and high-activity users with more than 99vv, containing 27.12%, 34.9% and 37.98% users respectively. We mock a CTR task as treating the ratings ≥ 4 star as positive samples and ratings lower than 4 stars as negative samples. Another task is a regression task to predict the actual ratings.

5.2.3 Experiment Setup. For both datasets, we try both MMOE and PLE for the SOTA micro switcher structures. For the macro structures, a flat 2-task and a flat 6-task Model are used as the baseline models, compared with an H-MTL 6-task model and an MFH 6-task Model. For each task in both models, we adopt a two-layer MLP network of size (64, 32). Similar to Section 5.1.2, the SOTA models to compare are the MMOE flat and PLE flat models.

5.2.4 Experiment Results. As shown in Table 5 and Table 6, for both datasets, MFH significantly improves the performance of all six tasks compared to the baseline flat(2-task) and flat(6-task) Models, for both the MMOE and PLE micro switcher structures. For the same macro structures, micro structure of PLE switcher performs slightly better than MMOE switcher. The flat(6-task) model only performs slightly better than the flat(2-task) model compared to the significant improvement MFH-MTL(6-task) brings, this demonstrates the task scaling benefit MFH generates.

6 CONCLUSION

In this paper we propose a novel Multi-Faceted Hierarchical multi-task learning model (MFH), which uses a multi-faceted hierarchical tree structure to improve the MTL efficiency and scalability from the macro perspective of task sharing. Offline and online experiment results on the industrial and public datasets show significant and consistent improvements of MFH over baseline SOTA models. Researching the possibilities of applying Meta Learning to the MFH tree structures will be the focus of future work.

REFERENCES

[1] Charu C Aggarwal et al. 2016. *Recommender Systems*. Vol. 1. Springer.

[2] Trapit Bansal, David Belanger, and Andrew McCallum. 2016. Ask the GRU: Multi-task Learning for Deep Text Recommendations. In *Proceedings of the 10th RecSys*. 107–114.

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

[4] Xiaokai Chen, Xiaoguang Gu, and Libo Fu. 2021. Boosting Share Routing for Multi-task Learning. In *Companion Proceedings of the Web Conference 2021*.

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 7–10.

[6] Ronan Collobert and Jason Weston. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning*. 160–167.

[7] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for Youtube Recommendations. In *Proceedings of the 10th ACM RecSys*. 191–198.

[8] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. 2010. The YouTube Video Recommendation System. In *Proceedings of the 4th ACM RecSys*. 293–296.

[9] 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 ACM SIGIR*. 2237–2241.

[10] Jianping Fan, Tianyi Zhao, Zhenzhong Kuang, Yu Zheng, Ji Zhang, Jun Yu, and Jinye Peng. 2017. HD-MTL: Hierarchical Deep Multi-task Learning for Large-scale Visual Recognition. *IEEE Transactions on Image Processing* 26, 4 (2017), 1923–1938.

[11] Alexander Felfernig, Ludovico Boratto, Martin Stettinger, and Marko Tkalčič. 2018. *Group Recommender Systems: An Introduction*. Springer.

[12] 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 CIKM*. 2493–2500.

[13] F Maxwell Harper and Joseph A Konstan. 2015. The Movielens Datasets: History and Context. *Acm Transactions on Interactive Intelligent Systems* 5, 4 (2015), 1–19.

[14] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive Mixtures of Local Experts. *Neural Computation* 3, 1 (1991), 79–87.

[15] 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.

[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 RecSys*. 20–28.

[17] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Why I like it: Multi-task Learning for Recommendation and Explanation. In *Proceedings of the 12th ACM RecSys*. 4–12.

[18] 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.

[19] 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*. 1930–1939.

[20] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire Dpace Multi-task Model: An Effective Approach for Estimating Post-click Conversion Rate. In *Proceedings of the 41st ACM SIGIR*. 1137–1140.

[21] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch Networks for Multi-task Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3994–4003.

[22] Duy-Kien Nguyen and Takayuki Okatani. 2019. Multi-task Learning of Hierarchical Vision-language Representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 10492–10501.

[23] Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. 2021. Fairness in Rankings and Recommendations: An Overview. *The VLDB Journal* (2021), 1–28.

[24] Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. 2017. Routing Networks: Adaptive Selection of Non-linear Functions for Multi-task Learning. *arXiv preprint arXiv:1711.01239* (2017).

[25] Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Sluice Networks: Learning What to Share Between Loosely Related Tasks. *arXiv preprint arXiv:1705.08142* 2 (2017).

[26] Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A Hierarchical Multi-task Approach for Learning Embeddings from Semantic Tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6949–6956.

[27] 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 *Proceedings of the 14th ACM RecSys*. 269–278.

[28] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *Proceedings of the 21th ACM SIGKDD*. 1235–1244.

[29] 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 ACM SIGIR*. 2187–2191.

[30] Ruobing Xie, Rui Wang, Shaoliang Zhang, Zhihong Yang, Feng Xia, and Leyu Lin. 2021. Real-time Relevant Recommendation Suggestion. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 112–120.

[31] Yu Zhang and Qiang Yang. 2021. A Survey on Multi-Task Learning. *IEEE Transactions on Knowledge and Data Engineering* (2021), 1–1.

[32] 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 RecSys*. 43–51.

[33] Barret Zoph and Quoc V Le. 2016. Neural Architecture Search with Reinforcement Learning. *arXiv preprint arXiv:1611.01578* (2016).