Curriculum Multi-Level Learning for Imbalanced Live-Stream Recommendation

Shuodian Yu[1] , **Junqi Jin**[2] , **Li Ma**[1] , **Xiaofeng Gao**[*1] , **Xiaopeng Wu**[2] , **Haiyang Xu**[2] and **Jian Xu**[2]
[1]MoE Key Lab of Artificial Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
[2]Alibaba Group, Beijing, China
{timplex233, mali-cs}@sjtu.edu.cn, gao-xf@cs.sjtu.edu.cn, {junqi.jjq, pengyi.wxp, shenzhou.xhy, xiyu.xj}@alibaba-inc.com

Abstract

In large-scale e-commerce live-stream recommendation, streamers are classified into different levels based on their popularity and other metrics for marketing. Several top streamers at the head level occupy a considerable amount of exposure, resulting in an unbalanced data distribution. A unified model for all levels without consideration of imbalance issue can be biased towards head streamers and neglect the conflicts between levels. The lack of inter-level streamer correlations and intra-level streamer characteristics modeling imposes obstacles to estimating the user behaviors. To tackle these challenges, we propose a curriculum multi-level learning framework for imbalanced recommendation. We separate model parameters into shared and level-specific ones to explore the generality among all levels and discrepancy for each level respectively. The level-aware gradient descent and a curriculum sampling scheduler are designed to capture the de-biased commonalities from all levels as the shared parameters. During the specific parameters training, the hardness-aware learning rate and an adaptor are proposed to dynamically balance the training process. Finally, shared and specific parameters are combined to be the final model weights and learned in a cooperative training framework. Extensive experiments on a live-stream production dataset demonstrate the superiority of the proposed framework.

1 Introduction

Nowadays, large e-commerce live-stream platforms have evolved into an important approach for advertisers, or rather streamers, to promote their products to prospective users. In e-commerce live-stream advertising systems, the estimations for user behavior feedback, i.e., click-through rate (CTR), conversion rate (CVR), and click-follow rate (CFR), are all essential values for advertisement ranking and recommendation. As different streamers have varied demands, it is crucial to precisely measure multiple user behaviors based on the multi-task model in order to calculate the overall rank score. In addition, the platform generally splits all streamers into distinct streamer level groups depending on their popularity, gross merchandise volume (GMV), and other indicators to facilitate the implementation of various marketing tactics. Accurate estimation for all streamers will improve the **efficiency** and **fairness** of the platform. It is not only conducive to the current market prosperity by maximizing the economic efficiency, but also essential to the long-term market prosperity by ensuring the healthy progression of tailed new streamers.

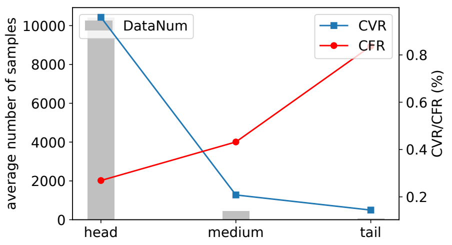


Figure 1: An example of a highly imbalanced distribution of streamers. The average exposure number of head streamers is significantly higher than the other two levels. There are also substantial disparities in user behaviors (i.e., CVR and CFR). For instance, users are more inclined to purchase items from trustworthy head streamers.

Despite the success in multi-task learning (MTL) [Zhang and Yang, 2021], the modeling for multi-level streamers is neglected in existing methods. Moreover, the long-tail effect in live-stream scenario is particularly noticeable. As illustrated in Figure 1, the distribution of user behaviors and the number of exposures differ between levels. A small number of head streamers occupy a large number of exposures. And the tail streamers not only have less exposure, but also have the lowest CVR. Predicting different levels of streamers with a unified model will inevitably lead to imbalanced prediction results since the model can be trained biased towards different levels. This inaccurate and unfair prediction results in streamers not being properly exposed, affecting the long-term ecological development of the live-stream platform.

Modeling multi-level streamers in a shared unified model turns out to be inadequate. In particular, it faces several challenges, including: (1) **Imbalanced samples in different levels:** Streamer level group that occupy a large number of

[*]Xiaofeng Gao is the corresponding author.

samples will affect others with less training samples, while training models for each level separately will lead to overfitting in tail level because of data sparsity. There are many works to solve the long-tail problems in the field of computer vision [Zhou *et al.*, 2020]. But they are hard to directly migrate to large-scale recommendation because of scenario particularity and complexity. How to mitigate the imbalance by transferring information from dominated levels to the tail levels is not trivial. (2) **Complex relationships between different levels:** The inner association between streamers at different levels is complicated, although all of the streamers are in the same feature space. There is both shared information that can migrate with each other (i.e., the representation of general features) and conflicting information that should be represented independently (i.e., the user behavior distribution). If we regard each level as a task, traditional multi-task models have difficulty modeling multiple levels beyond multiple tasks due to the quadratic complexity. The emerging multi-domain learning method (MDL) [Sheng *et al.*, 2021] points out a way to take each level as a domain and separate the parameters explicitly. Nevertheless, its simple DNN structure is not compatible with existing complex multi-task models. And it still has no consideration towards data imbalance among different levels.

Another naive solution is to train a base model with all samples and build a finetuned separate model for each level. However, it still suffers from complex relationships on multiple levels. It cannot explicitly express the process of information migration between levels, leading to degradation of prediction efficiency. What's more, for large-scale training, it is infeasible to determine what base model can reach the global optimal with following fine-tune procedure. The cooperation between the base model and the finetuning process is not established. How to capture the complex correlations among multiple levels of streamers from the imbalanced data for the large-scale scenario is all-important.

To address the aforementioned limitations, we propose the curriculum imbalanced multi-level learning framework (CMLIR) and exploit the generality and discrepancy among levels separately from the perspective of model parameters. Specifically, the common de-biased knowledge among all streamers is explicitly represented by shared parameters, while the conflicts across different levels are eliminated and separately modeled by level-specific parameters. The final model parameters at each level consist of a combination of shared parameters and their respective specific parameters. They are trained with a cooperative learning framework to reach the global optimal:

- Generality: it learns the shared parameters on the full dataset containing all streamers as the shared pattern. Considering the imbalanced level distribution, we draw on the curriculum learning strategy to prevent the shared knowledge dominated by head streamers, which enables the representation learning of general features and knowledge in a balanced manner.
- Discrepancy: it learns the level-specific parameters for each streamer level to extract the individuality, since streamers in different levels have distinct characteristics conflicting with the others. To alleviate the imbalanced sample distribution, a hardness-aware learning rate based on the specific gradient is set to make specific parameters of all levels converge simultaneously.
- Cooperative learning: it learns two parts of parameters in a unified framework. The combination of shared and level-specific parameters forms the final model weights for each level. We claim that a progressive learning approach that updating two parts iteratively is superior to training them individually. They work cooperatively with an adaptor to achieve the global optimal.

Our contributions can be summarized as follows:

- We claim that in live-stream scenario, different levels of streamers have distinct and imbalanced data distributions, which leads to inaccurate and unfair performance to predict them in a unified model. We propose a curriculum imbalanced multi-level learning framework to address the issue to learn both generality and discrepancy of different levels.
- We design a shared and level-specific parameters separation architecture and train them in a cooperative learning framework. The key idea is to alleviate data imbalance by explicitly modeling the balanced commonality with the shared part and the level characteristic with the specific one. Under a curriculum scheduler, the balanced shared and specific parameters are learned cooperatively, and combined to be the model weights for each level.
- We conduct extensive experiments on a real-world live-stream advertising dataset. Our learning framework can be integrated with existing models and consistently outperforms state-of-the-art multi-task and multi-domain methods. We prove the effectiveness of the proposed architecture and components.

2 Related Work

2.1 Multi-Task and Multi-Domain Learning

Multi-task learning [Caruana, 1997] aims to leverage shared knowledge across multiple related tasks to facilitate the learning of each task and improve overall performance. [Ma *et al.*, 2018b; Xi *et al.*, 2021] models the dependencies between multiple types of user behavior feedback in recommender system. Designing network architectures for parameters sharing across tasks [Thrun and Pratt, 2012] is also an important approach for more generalization. [Misra *et al.*, 2016] proposes cross-stitch units to model the combination of task-specific layers for each task. [Ma *et al.*, 2018a] proposes Multi-gate Mixture-of-Experts (MMoE) that shares several expert networks across all tasks and train a gating network to optimize each task. [Tang *et al.*, 2020] further introduces a shared-specific expert structure to avoid parameters conflict and progressively extract shared- and specific- features.

Despite of the success in multi-task modeling, multi-domain learning [Dredze *et al.*, 2010; Li *et al.*, 2020], which deals with data from multiple domains, has recently received attention. To facilitate knowledge transformation from multiple scenarios to each other, [Sheng *et al.*, 2021] proposes a star topology recommender that separates the model weights into shared and domain specific parts. [Luo *et al.*, 2022] addresses domain conflict and overfitting problem through learning method optimizing. [Zhang *et al.*, 2022a] leverages model-based meta-learning to capture inter-domain correlations. [Zou *et al.*, 2022] stacks multi-task layers over multi-scenario layers under MMoE architecture and propose an expert selection algorithm. However, to the best of our knowledge, these approaches barely take data imbalance between different domains into consideration explicitly. The biased shared parameters and limited samples in minor domains will lead to non-optimal prediction results as we have introduced.

Previous MTL and MDL methods mainly focus on the parameters sharing among tasks through the perspective of the model architecture. In contrast, our proposed method factors the parameters of a fixed backbone model into two parts to represent the generality and discrepancy among streamers. Moreover, our approach will focus on optimizing the training procedure of two parameter parts.

2.2 Imbalance Learning

Imbalance problem is especially obvious in online live-stream scenarios. This phenomenon leads to a large amount of observations for the head streamers and very little data for the tails. There are two common ways to solve it: re-sampling approaches to re-balance the dataset or modify the loss function to assign different weights for major/minor data. [Cui *et al.*, 2019] devise a re-weighting scheme that employs the effective number of samples for each class to re-balance the loss and produce a class-balanced loss. [Alshammari *et al.*, 2022] explores three techniques to balance weights and [Kang *et al.*, 2020] uses instance-balanced sampling based on the high-quality representations for long-tailed recognition. [Zhou *et al.*, 2020] proposes a two-stage training to get a balanced classifier. In recommendation, [Zhang *et al.*, 2021b] focus on knowledge migration in the long-tail distribution of items through meta-learning and curriculum transfer, but it simplifies the modeling of long-tail distribution by transferring knowledge solely between head and tail items.

3 Problem Formulation

In the live-stream e-commerce platform, the prediction of user feedback is one of typical key indicators for advertising. Estimating them precisely is essential, but it still has many obstacles in this emerging scenario. Given a user $u \in \mathcal{U}$ and a streamer $s \in \mathcal{S}$, the input vector $\mathbf{x}$ consists of features of the specific u and s, and other contextual features. The model $\hat{y} = f(\mathbf{x}; \Theta)$ predicts binary behavior $y = Pr(\text{behavior} = 1|\mathbf{x})$ based on $\mathbf{x}$ and learnt parameters Θ. Furthermore, the platform has divided all streamers into different streamer level group based on their popularity, GMV and other metrics for different marketing strategies, and s belongs to the specific level $l_s \in \{1, \dots, C\}$. In reality, minor streamers only have little probability to be exposed because of less budget and popularity. Hence, the number of training samples in each streamer level group $\mathcal{D}^c = \{(u,s)|l_s = c\}$ follows a long-tail distribution and the predicting performance for different groups is highly imbalanced.

Our goal is to improve predicting results $\hat{y}$ for different user-streamer pairs by optimizing model weights Θ. Specifically, predicting results can be quantified as two objectives through the metric Area under the ROC curve (AUC):

1. Efficiency: maximizing the AUC_c within samples of each level c, and the overall AUC for the whole test data.
2. Fairness: minimizing Gini coefficient [Allison, 1978], which is a widely used measure to calculate the economic inequalities: $G = \frac{\sum_{i=1}^{C}\sum_{j=1}^{C}|\text{AUC}_i - \text{AUC}_j|}{C\sum_{i=1}^{C}\text{AUC}_i}$.

4 Proposed Model

4.1 Overview

As we have discussed above, ignoring the level discrepancy and forecasting all with a single model will result in inferior performance due to conflicts and imbalance. Because of the inability to learn generic features, training models separately using only individual samples within each level also leads to poor performance. To this end, we propose a cooperative learning framework that optimizes the shared parameters for all levels and the level-specific parameters independently, in order to distill the general and contradictory information, respectively. The final model weights at each level are determined by combining the shared and specific parameters, as shown in Figure 2. The imbalance problem is resolved during the training of de-biased shared and level-specific parameters.

4.2 Generality: Shared Parameters Learning

To extract shared information among different levels, we utilize the training samples from all levels to learn a shared representation that can express commonalities between various streamer levels. Meta-learning based methods, e.g. MAML [Finn *et al.*, 2017] and Reptile [Nichol *et al.*, 2018], have been widely used to achieve an initialization of recommender which could fast adapt to specific tasks with few update steps. However, these meta learning approaches are predicated on the notion that the data is insufficient and rarely take the imbalance problem between tasks into account. In order to prevent the multi-level conflicts and capture a balanced shared representation, we propose the shared parameters learning process with two components: level-aware gradient descent and curriculum sampling scheduler.

The level-aware gradient descent component aims to aggregate common directions from each level, which corresponds to finding commonalities between different levels, as shown in Figure 2 (B). In one iteration, we first update model parameters $\tilde{\Theta}$ on each level with the sampled data $\mathbf{x}_c$ with several update steps and record the gradients respectively:

$$\tilde{\Theta}_c \leftarrow \tilde{\Theta} - \gamma \nabla_{\tilde{\Theta}} \tilde{\mathcal{L}}_c(\mathbf{x}_c; \tilde{\Theta}) \tag{1}$$

To aggregate the updated weights to obtain the shared parameters and prevent gradients in each level from negatively affecting each other, we seek to alter the gradients themselves and remove the conflicting portions. As most gradient-based multi-task learning methods [Yu *et al.*, 2020] do, the conflicts can be assessed by the negative inner product of the gradients that point away from each level. The average first-order gradients in [Nichol *et al.*, 2018] is adopted to update $\tilde{\Theta}$ as:

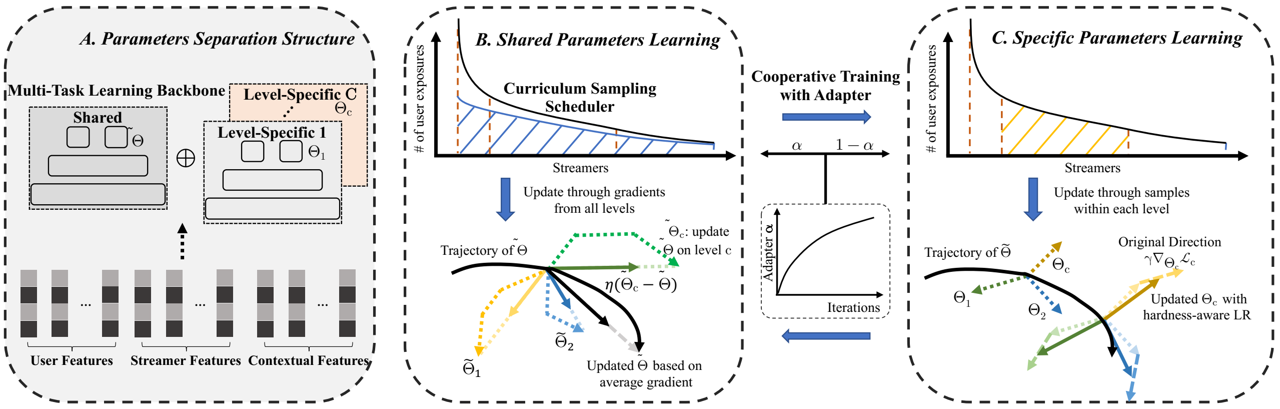

A. Parameters Separation Structure
Multi-Task Learning Backbone
Shared
Level-Specific C
Level-Specific 1
User Features
Streamer Features
Contextual Features
B. Shared Parameters Learning
Curriculum Sampling Scheduler
# of user exposures
Streamers
Update through gradients from all levels
Trajectory of Θ̃
Θ̃c: update Θ̃ on level c
η(Θ̃c − Θ̃)
Updated Θ̃ based on average gradient
Cooperative Training with Adapter
α
1 − α
Adapter α
Iterations
C. Specific Parameters Learning
Update through samples within each level
Original Direction γ∇Θc ℒc
Updated Θc with hardness-aware LR

Figure 2: An illustration of our proposed learning framework. In part A, the backbone model take the shared and independent level-specific parameters as the final weights for prediction in each level. The part B (Sec. 4.2) balances examples from all streamers with a curriculum sampling scheduler and aggregate the gradients from each level to update shared parameters $\tilde{\Theta}$ along the bold black trajectory. The part C stands for specific parameters learning (Sec. 4.3) through samples within each level. The two parts of parameters are trained cooperatively with an adapter to trade-off the learning progress.

$$\tilde{\Theta} \leftarrow \tilde{\Theta} + \frac{\alpha}{C}\sum_{c=1}^{C}(\tilde{\Theta}_c - \tilde{\Theta}), \tag{2}$$

where α is the step size parameter annealing during training.

In practice, data at different levels $|\mathcal{D}_c|$ can be severely unbalanced, as shown in Table 1. As a result, models trained on such an imbalanced distribution will be skewed towards the dominating proportion, resulting in performance degradation for the remaining streamers. Sampling is a widely used and effective technique to deal with imbalanced problem [Wang *et al.*, 2019]. [Chen *et al.*, 2021] set curriculum sampling based on the hardness of tasks. To address this problem, a curriculum sampling scheduler is proposed to enhance the tail learning and de-bias the shared parameters in the imbalance scenario. Explicitly, we define the curriculum training samples for level c as $\mathbf{x}_c(i)$, where i refers to the current iteration. We perform linear interpolation of the number of samples over iterations to re-balance the level distributions.

$$|\mathbf{x}_c(i)| = |\mathbf{x}_c| \times (\frac{|\mathcal{D}^{\text{tail}}|}{|\mathcal{D}^c|})^{i/N}, \tag{3}$$

where N is the total number of iterations, and $\mathcal{D}^{\text{tail}}$ is the tail level dataset. According to the definition of $\mathbf{x}_c(i)$, the data distribution of different levels progressively varies from the initial one at $i = 0$ to the balanced state on the final iteration. Therefore, the head streamers will be dynamically resampled over iterations to be close to the tail level, and the learned shared parameters will be de-biased to ensure generality modeling for all levels.

4.3 Discrepancy: Specific Parameters Learning

Since the shared parameters have depicted the balanced inter-level correlations among different levels, it is expected that the level-specific knowledge will be updated during the specific parameters learning phase using samples within its own level. The final model weights for each level c are composed of the learned shared parameters $\tilde{\Theta}$ and the specific parameters Θ_c for refined multi-task prediction, where the level discrepancy is explicitly represented by Θ_c.

Specifically, since we have divided the model weights of level c into $\tilde{\Theta}$ and Θ_c, we optimize each Θ_c in each iteration based on fixed $\tilde{\Theta}$ as shown Figure 2 (C). A straightforward approach refers to pretrain and fine-tune (FT). That is, tuning the shared parameters $\tilde{\Theta}$ on each level's particular samples to obtain the specific model. However, such FT technique cannot guarantee global optimal and is prone to overfitting minor streamers. Derived from inner loop phase in meta learning [Finn *et al.*, 2017], we take k gradient descent steps at each iteration and accumulate the direction points to the end parameters as the level-specific parameters. In other words, Θ_c can be regarded as the colored solid arrows shown in the lower right corner of Figure 2, which can be formulated as:

$$\Theta_c \leftarrow \Theta_c - \gamma \nabla_{\Theta_c} \mathcal{L}_c(\mathbf{x}'_c; \tilde{\Theta} + \Theta_c), \tag{4}$$

where γ denotes the inner learning rate, $\mathbf{x}'_c$ is the sampled set from $\mathcal{D}_c$, and the element-wise summation $\tilde{\Theta} + \Theta_c$ is the final weight for level c.

Nevertheless, such optimization method still has the following drawbacks: (1) Hard to determine when all levels converge, especially when the shared parameters are not fully converged. Differences in sampled examples can lead to fluctuations in parameters. (2) No consideration of the hardness and imbalance of different levels. This disparity in difficulty between different levels leads to under-fitting and over-fitting at different levels with unbalanced prediction results. To tackle these problems, different from shared parameters learning phase, we employ the gradient $\nabla_{\Theta_c}\mathcal{L}_c$ to measure the "hardness" and balance the convergence of different levels as [Chen *et al.*, 2018] does. Concretely, $\|\nabla_{\Theta_c}\mathcal{L}_c\|_2$ is used to measure the distance to the optimal parameters, and the model can self-judge the learning hardness of different levels. We normalized the norm of gradient from each level

Algorithm 1: Training Algorithm for CMLIR

```
   Input: Training dataset $\mathcal{D}$, learning rate $\gamma$.
1  Construct set $\{\mathcal{D}^c\}_{c=1}^{C}$ for each level;
2  Randomly Initialize $\tilde{\Theta}$;
3  Initialize $\{\Theta_c\}_{c=1}^{C}$ as zeros;
4  for $i \leftarrow 1$ to $N$ do
5      Shared parameters $\Theta$ optimization:
6      for $c \leftarrow 1$ to $C$ do
7          Sample $\mathbf{x}_c$ from $\mathcal{D}^c$ for each level as Eqn. 3;
8          Calculate $\tilde{\mathcal{L}}_c(\mathbf{x}_c; \tilde{\Theta})$ to update $\tilde{\Theta}_c$ as Eqn. 1;
9      Do a global step to update $\tilde{\Theta}$ as Eqn. 2
         $\tilde{\Theta} \leftarrow \tilde{\Theta} + \frac{\alpha}{C}\sum_{c=1}^{C}(\tilde{\Theta}_c - \tilde{\Theta})$;
10     Specific parameters $\{\Theta_c\}_{c=1}^{C}$ optimization:
11     Sample $\{\mathbf{x}'_c\}_{c=1}^{C}$ uniformly at random from each level;
12     for $c \leftarrow 1$ to $C$ do
13         Calculate $\mathcal{L}_c(\mathbf{x}'_c; \tilde{\Theta} + \Theta_c)$;
14         Get hardness-aware rate $\beta_c = \frac{\|\nabla_{\Theta_c}\mathcal{L}_c\|_2}{\sum_{c'=1}^{C}\|\nabla_{\Theta_{c'}}\mathcal{L}_{c'}\|_2}$;
15         Do a specific step as Eqn. 7
             $\Theta_c \leftarrow \Theta_c - (1-\alpha)\beta_c\gamma\nabla_{\Theta_c}\mathcal{L}_c$;
```

to get the hardness-aware learning rate:

$$\beta_c = \frac{\|\nabla_{\Theta_c}\mathcal{L}_c\|_2}{\sum_{c'=1}^{C}\|\nabla_{\Theta_{c'}}\mathcal{L}_{c'}\|_2} \tag{5}$$

To alleviate the fluctuation problem, we set another constraint α, which is automatically varied according to the iterations. We expect the model to be inclined to learn shared information for fast adaptation. While at the end of training that the shared part is relatively smooth, the learning rate of shared parameters increases to distinguish the personality of each level. As a result, a power growth α satisfies such demands and achieves a satisfactory performance intuitively:

$$\alpha = 1 - (\frac{i}{N})^{\eta}, \tag{6}$$

where i is the number of current training iteration, N is the maximum iteration, and η is a hyper-parameter coefficient. The α is also utilized in shared parameter learning as Eqn. 2, which can be viewed as an adaptor for dynamically balancing the training process of two parameter components. And the overall optimization of specific parameters is formulated as:

$$\Theta_c \leftarrow \Theta_c - (1-\alpha)\beta_c\gamma\nabla_{\Theta_c}\mathcal{L}_c(\mathbf{x}'_c; \tilde{\Theta} + \Theta_c) \tag{7}$$

4.4 Cooperative Model Training

To make the shared and specific parameters work cooperatively, we integrate the learning procedure into a unified framework. Algorithm 1 summarizes the whole training process of CMLIR. Final model weights for each level is determined by combination $\tilde{\Theta} + \Theta_c$. The training framework ensures the smooth transfer in both shared and specific parameters optimization as introduced above. The two parts of parameters are trained alternately to achieve the global optimum. In each iteration, we first update shared parameters $\tilde{\Theta}$ with the curriculum sampling scheduler to shift the learning focus from general feature representation to balanced learning. Then we optimize specific parameters Θ_c for each level individually based on the updated $\tilde{\Theta}$. The hardness-aware learning rate adjustment guarantees balanced and simultaneous updating between different levels.

Level	#streamer	#user	#interactions
Tail	2,334	50,075	129,676
Medium	5,219	118,615	2,343,869
Head	19	72,301	197,744
Overall	7,446	118,922	2,671,289

Table 1: Statistics of the collected dataset in three levels.

5 Experiments

In this section, we conduct experiments on a live-stream dataset. We seek to answer the following research questions: **RQ1**: How do the proposed CMLIR framework performs compared with the state-of-the-art MTL and MDL methods; **RQ2**: How do the proposed components in CMLIR perform individually and complement with each other for improving the performance; **RQ3**: How does our proposed shared and level-specific parameters separation architecture for different level ease the multi-level imbalance; **RQ4**: How do the hyper-parameters influence the performance?

5.1 Experimental Settings

Dataset. To exploit the effectiveness of CMLIR, we conduct our experiments on a dataset collected from Taobao Live, a popular live-stream e-commerce platform with hundreds of millions of users and tens of thousands of streamers. The data distribution among different streamers is extremely imbalanced compared to other MTL public dataset. We record a part of the advertising traffic logs with two types of user feedback, i.e., conversion rate (CVR) and click-follow rate (CFR), for six days as our dataset. These two behaviors are predicted with the MTL model. We take the first 5 days of data as the training set, and the last day is used for validation and testing. We divide the dataset into three parts according to the streamer level, which is pre-defined by the platform. The detailed statistics are shown in Table 1.

Compared methods. To verify the effectiveness of the proposed approach, we compare our method with several baselines that can be categorized into three aspects:

- Multi-task learning methods: Shared-Bottom [Caruana, 1997], MMoE [Ma *et al.*, 2018a], PLE [Tang *et al.*, 2020]. We apply MTL models to predict CVR and CFR simultaneously. They jointly learn a unified model for all levels without distinguishing parameters.
- Multi-domain method: STAR [Sheng *et al.*, 2021]. In our scenario, it treats different levels as different domains and learns individual parameters for each level. During implementation process. we extend its FCN into shared-bottom structure for multi-task prediction.

- Pretrain and Fine-tune (FT). We pretrain the MTL models on all the data, and then fine-tune on the each streamer level to achieve parameter separation. Since our method is model agnostic, we choose two best MTL competitors, i.e. MMoE and PLE, as backbones and compare our CMLIR with fine-tune strategy.

Implementation details. We train all models by Adam optimizer [Kingma and Ba, 2014]. The loss functions are Binary Cross Entropy following backbone model. The batch size is set to 512 and the learning rate is 0.001. We use ReLU [Nair and Hinton, 2010] as the activation function for all methods. The DNN dimension of experts in MMoE and PLE are set as $[128, 64]$ for all methods. For CMLIR, the iteration number is set as 25, the initial step size is 1.0, the adaptor coefficient η is 1.0, and the step number k in specific learning is set as 4. All the above models are trained with Tesla T4 GPU and implemented with Tensorflow 1.14. A part of experiments are conducted on XDL [Zhang *et al.*, 2022b].

Metrics. We use the most widely adopted metric AUC for evaluation. AUC measures the bipartite ranking between positive and negative samples. Concretely, we count the respective AUC of each level and overall AUC (mix prediction results from all streamers) for CVR and CFR. Moreover, to measure the fairness for different levels, we refer to the Gini Coefficient following [Jamal and Qi, 2019] as introduced in Sec. 3. All the results are based on the average of 3 runs.

5.2 Overall Performance (RQ1)

We show the overall experiment results in Table 2. Since the proposed learning framework can integrate with different backbone models, we apply CMLIR with two MTL methods, MMoE and PLE. Our proposed CMLIR consistently outperforms baselines on two tasks CVR and CFR from different stream levels, proving its effectiveness and universality. To validate the existence of conflicts between each level, we first report the AUC scores of MMoE and PLE that predicts in a unfied model compared with their FT versions. The FT strategy obviously helps all the base models to achieve preciser results in most of tasks and levels, suggesting that for different levels, learning individual parameters to obtain respective models will boost performance.

Second, among different strategies for parameters separation (e.g., FT, multi-domain, ours), CMLIR achieves the best performance on AUC metric in various tasks and levels. STAR outperforms SharedBottom in most of metrics and even achieves better performance in some tasks than MMoE and PLE that have more complicate structures. It demonstrates that the parameter separation for different levels is feasible. Nevertheless, we assume that the worst result in CVR task of tail level can be considered as a result of data imbalance. While the FT and STAR do not consider the imbalance distribution of different streamer levels, the outperforming of CMLIR illustrates the superiority of considering both the migration and sharing of information among different levels, as well as the need to extract the characteristic of the levels individually through the level-specific parameters.

Moreover, it is notable that the performance of medium level degrades when training the individual models which can be found through the comparison between medium CVR AUC of MTL methods and that of their FT version. Since this level occupies most of the training samples, the data imbalance leads to sufficient training when all streamers share a unified model and over-fitting during FT. CMLIR mitigates the performance degradation and even achieves better results than the unified model. It proves that we resolve the data imbalance between different levels to some extent.

Finally, since our approach accounts for the imbalance distribution, we achieve greater performance gains on head and tail levels with relatively fewer samples. Not only does it not harm the overall performance, but it is able to ultimately improve the fairness of the model at different levels by addressing the imbalance problem. The Gini coefficients (lower values indicate fairer results) in two tasks reveals that the inequity in predictive performance is mitigated through CMLIR framework. Additionally, we discover that FT also achieves a lower Gini coefficient than a unified model with the same backbone. The high Gini coefficients of STAR indicates that the current MDL approach does not address the imbalance problem between domains well.

5.3 Ablation Study

We design a series of ablation studies to better understand the performance of the proposed model in Table 3.

Impact of Curriculum Sampling Scheduler (RQ2, 3). Designing sampling strategies is a very common and effective approach to deal with imbalance distribution. To analyze the contribution of curriculum learning strategy, we design a series of variants toward sampling strategies [Zhang *et al.*, 2021a]:

- FT (Balanced): Train a base model with balanced data (samples distribute evenly) and finetune individually.
- MLIR: Remove the curriculum sampling of CMLIR, and sample according to original distribution.
- MLIR (Over-sampling): Sample from minor levels repeatedly to re-balance the sampled data
- MLIR (Under-sampling): Down-sample the head streamers for re-balancing in contrast to over-sampling.

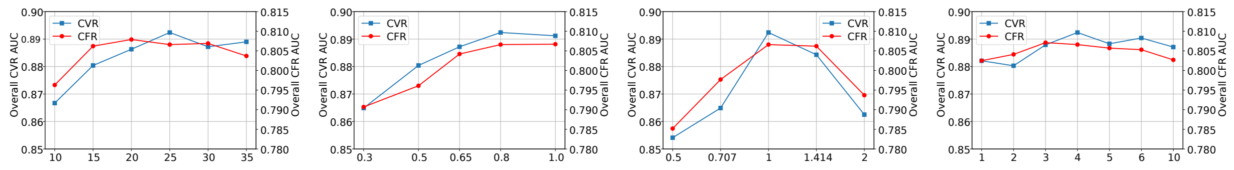
From the middle part of Table 3, we can see that FT (Balanced) has the worst performance, which indicates that directly use balanced data is not suitable to learn the generalization features. No matter what sampling strategy is used, MLIR with re-balanced samples still achieve bad results. It demonstrates that the shared parameters trained on balanced data are unsuitable for extracting commonality information that can fast adapt to optimal weights at each level.

However, we observe that the overall AUCs on two tasks of MLIR slightly drop compared to CMLIR, which shows the usefulness of curriculum sampling scheduler. Based on the above analysis, we can summarize that sampling is crucial for the improvement of the model. Inappropriate balancing strategies bring about a decline in performance, whereas curricular settings that are adequate can boost results.

Impact of Parameter Separation (RQ4). To discover the impact of our shared- and specific- parameters framework, we conduct ablation analysis by altering components:

Method	Tail AUC		Medium AUC		Head AUC		Overall AUC		Gini COEFF	
	CVR	CFR	CVR	CFR	CVR	CFR	CVR	CFR	CVR	CFR
SharedBottom	0.8183	0.7733	0.8633	0.7928	0.7828	0.7407	0.8884	0.8016	0.0436	0.0301
MMoE	0.8012	0.7789	0.8702	0.7965	0.7768	0.7513	0.8835	0.8035	0.0509	0.0259
MMoE+FT	0.8278	0.7846	0.8695	0.7993	0.7859	0.7568	0.8896	0.8063	0.0449	0.0242
PLE	0.8057	0.7801	0.8684	0.8020	0.7710	0.7734	0.8882	0.8073	0.0531	0.0162
PLE+FT	0.8134	0.7842	0.8645	**0.8037**	0.7810	0.7736	0.8869	0.8091	0.0453	0.0170
STAR	0.7653	0.7828	0.8658	0.8016	0.7706	0.7499	0.8891	0.8042	0.0558	0.0295
MMoE+CMLIR	**0.8323**	0.7778	**0.8705**	0.7975	**0.7894**	0.7693	**0.8924**	0.8066	**0.0434**	**0.0160**
PLE+CMLIR	0.8238	**0.7851**	0.8679	0.8031	0.7825	**0.7742**	0.8891	**0.8100**	0.0454	0.0163

Table 2: Results of different approaches on a live-stream production dataset. The bold value marks the best one in each column.



(a) Iteration Number N (b) Step size α (c) Adaptor Coefficient η (d) Step Number k

Figure 3: Overall AUC performance for CVR and CFR tasks w.r.t. different hyper-parameters.

Method	CVR AUC	CFR AUC
w/o Shared Param.	0.8302	0.7025
w/o Specific Param.	0.8814	0.7996
w/o Hardness-Aware LR	0.8863	0.8058
w/o Adaptor	0.8870	0.8044
FT (Balanced)	0.8692	0.7948
MLIR	0.8893	0.8034
MLIR (Over-Sampling)	0.8793	0.7967
MLIR (Under-Sampling)	0.8805	0.7938
CMLIR (Full Model)	**0.8924**	**0.8066**

Table 3: Ablation study of CMLIR. We present the overall AUC to indicate the performance.

- w/o Shared Param.: Remove the shared parameters learning and directly train models for each level using only samples from that level.
- w/o Specific Param.: Remove the specific parameters learning and inference with shared parameters directly.
- w/o Hardness-Aware LR: Remove the normalization term in Eqn. 5.
- w/o Adaptor: Remove the adaptor in Eqn. 6 and replace the step size for two parts with a constant 1.

Note that the w/o shared parameter version does not learn commonalities, and the models at different levels are wholly separate. The worst performance proves the common information plays a key role in recommendation. On the other extreme, removing the specific parameters will degrade the method to the single model, of which the results are also close to MMoE in Table 2. The performance of w/o Hardness-aware LR is poorer than CMLIR. It manifests the efficacy of normalization of learning rate which balanced convergence of different levels. The fact that the full model performs slightly worse with the adapter removed proves that the adaptor contributes to the overall performance to some degree.

5.4 Hyper-Parameters Sensitivity

This subsection studies the impact of key hyper-parameters (i.e. iteration number, step size, adaptor coefficient, and step numbers) while others unchanged. Figure 3(a) reveals that proper number of iterations can bring performance improvement. But two tasks tend to converge inconsistently. Figure 3(b) shows that annealing step size α from 1.0 can benefit in achieving the optimum, which reaches the same conclusion as [Nichol *et al.*, 2018]. Figure 3(c) shows the linear variation of the adaptor with the reported $\eta = 1$ yields the best results. Figure 3(d) indicate the low value of step number k will result in under-fitting of specific information, even degenerate into the version without level-specific parameters. A large number of steps results in excessive learning of specific characteristics, which is detrimental to model prediction.

6 Conclusion

In this paper, we explore and tackle the level-wise imbalance distribution problem in recommendation. We propose a curriculum imbalanced multi-level learning framework CMLIR for multi-task recommendation, which integrates shared and specific parameters learning to extract the generality and discrepancy among different levels and resolve the imbalance problem. Extensive experiments on a real-world dataset show the effectiveness of CMLIR. For future work, we are interested in more generalized imbalance problems in recommendation to fully exploit the potential of CMLIR.

Acknowledgements

This work was supported by National Key R&D Program of China [2020 YFB1707900], National Natural Science Foundation of China [62272302, 62172276], Shanghai Municipal Science and Technology Major Project [2021 SHZDZX0102], and Alibaba Group through Alibaba Innovative Research Program.

References

[Allison, 1978] Paul D Allison. Measures of inequality. *American sociological review*, pages 865–880, 1978.

[Alshammari *et al.*, 2022] Shaden Alshammari, Yu-Xiong Wang, Deva Ramanan, and Shu Kong. Long-tailed recognition via weight balancing. In *CVPR*, pages 6897–6907, 2022.

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

[Chen *et al.*, 2018] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, pages 794–803. PMLR, 2018.

[Chen *et al.*, 2021] Yudong Chen, Xin Wang, Miao Fan, Jizhou Huang, Shengwen Yang, and Wenwu Zhu. Curriculum meta-learning for next poi recommendation. In *SIGKDD*, pages 2692–2702, 2021.

[Cui *et al.*, 2019] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, pages 9268–9277, 2019.

[Dredze *et al.*, 2010] Mark Dredze, Alex Kulesza, and Koby Crammer. Multi-domain learning by confidence-weighted parameter combination. *Machine Learning*, 79(1):123–149, 2010.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[Jamal and Qi, 2019] Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *ICCV*, pages 11719–11727, 2019.

[Kang *et al.*, 2020] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, pages 1–16, 2020.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Li *et al.*, 2020] Pengcheng Li, Runze Li, Qing Da, An-Xiang Zeng, and Lijun Zhang. Improving multi-scenario learning to rank in e-commerce by exploiting task relationships in the label space. In *CIKM*, pages 2605–2612, 2020.

[Luo *et al.*, 2022] Linhao Luo, Yumeng Li, Buyu Gao, Shuai Tang, Sinan Wang, Jiancheng Li, Tanchao Zhu, Jiancai Liu, Zhao Li, Binqiang Zhao, et al. Mamdr: A model agnostic learning method for multi-domain recommendation. *arXiv preprint arXiv:2202.12524*, 2022.

[Ma *et al.*, 2018a] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *SIGKDD*, pages 1930–1939, 2018.

[Ma *et al.*, 2018b] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. Entire space multi-task model: An effective approach for estimating post-click conversion rate. In *SIGIR*, pages 1137–1140, 2018.

[Misra *et al.*, 2016] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *ICCV*, pages 3994–4003, 2016.

[Nair and Hinton, 2010] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[Nichol *et al.*, 2018] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[Sheng *et al.*, 2021] Xiang-Rong Sheng, Liqin Zhao, Guorui Zhou, Xinyao Ding, Binding Dai, Qiang Luo, Siran Yang, Jingshan Lv, Chi Zhang, Hongbo Deng, et al. One model to serve all: Star topology adaptive recommender for multi-domain ctr prediction. In *CIKM*, pages 4104–4113, 2021.

[Tang *et al.*, 2020] Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *RecSys*, pages 269–278, 2020.

[Thrun and Pratt, 2012] Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[Wang *et al.*, 2019] Yiru Wang, Weihao Gan, Jie Yang, Wei Wu, and Junjie Yan. Dynamic curriculum learning for imbalanced data classification. In *ICCV*, pages 5017–5026, 2019.

[Xi *et al.*, 2021] Dongbo Xi, Zhen Chen, Peng Yan, Yinger Zhang, Yongchun Zhu, Fuzhen Zhuang, and Yu Chen. Modeling the sequential dependence among audience multi-step conversions with multi-task learning in targeted display advertising. In *SIGKDD*, 2021.

[Yu *et al.*, 2020] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *NeurIPS*, 33:5824–5836, 2020.

[Zhang and Yang, 2021] Yu Zhang and Qiang Yang. A survey on multi-task learning. *TKDE*, 2021.

[Zhang *et al.*, 2021a] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. Deep long-tailed learning: A survey. *arXiv preprint arXiv:2110.04596*, 2021.

[Zhang *et al.*, 2021b] Yin Zhang, Derek Zhiyuan Cheng, Tiansheng Yao, Xinyang Yi, Lichan Hong, and Ed H Chi. A model of two tales: Dual transfer learning framework for improved long-tail item recommendation. In *WWW*, pages 2220–2231, 2021.

[Zhang *et al.*, 2022a] Qianqian Zhang, Xinru Liao, Quan Liu, Jian Xu, and Bo Zheng. Leaving no one behind: a multi-scenario multi-task meta learning approach for advertiser modeling. In *WSDM*, pages 1368–1376, 2022.

[Zhang *et al.*, 2022b] Yuanxing Zhang, Langshi Chen, Siran Yang, Man Yuan, Huimin Yi, Jie Zhang, Jiamang Wang, Jianbo Dong, Yunlong Xu, Yue Song, et al. Picasso: Unleashing the potential of gpu-centric training for wide-and-deep recommender systems. In *ICDE*, pages 3453–3466. IEEE, 2022.

[Zhou *et al.*, 2020] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *CVPR*, pages 9719–9728, 2020.

[Zou *et al.*, 2022] Xinyu Zou, Zhi Hu, Yiming Zhao, Xuchu Ding, Zhongyi Liu, Chenliang Li, and Aixin Sun. Automatic expert selection for multi-scenario and multi-task search. In *SIGIR*, page 1535–1544, 2022.