Adversarial Feature Translation for Multi-domain Recommendation

Xiaobo Hao*
WeChat, Tencent
Beijing, China
rolyhao@tencent.com

Yudan Liu*
WeChat, Tencent
Beijing, China
danydliu@tencent.com

Ruobing Xie*
WeChat, Tencent
Beijing, China
ruobingxie@tencent.com

Kaikai Ge
WeChat, Tencent
Beijing, China
kavinge@tencent.com

Linyao Tang
WeChat, Tencent
Beijing, China
linyaotang@tencent.com

Xu Zhang
WeChat, Tencent
Beijing, China
xuonezhang@tencent.com

Leyu Lin
WeChat, Tencent
Beijing, China
goshawklin@tencent.com

ABSTRACT

Real-world super platforms such as Google and WeChat usually have different recommendation scenarios to provide heterogeneous items for users' diverse demands. Multi-domain recommendation (MDR) is proposed to improve all recommendation domains simultaneously, where the key point is to capture informative domain-specific features from all domains. To address this problem, we propose a novel Adversarial feature translation (AFT) model for MDR, which learns the feature translations between different domains under a generative adversarial network framework. Precisely, in the multi-domain generator, we propose a domain-specific masked encoder to highlight inter-domain feature interactions, and then aggregate these features via a transformer and a domain-specific attention. In the multi-domain discriminator, we explicitly model the relationships between item, domain and users' general/domain-specific representations with a two-step feature translation inspired by the knowledge representation learning. In experiments, we evaluate AFT on a public and an industrial MDR datasets and achieve significant improvements. We also conduct an online evaluation on a real-world MDR system. We further give detailed ablation tests and model analyses to verify the effectiveness of different components. Currently, we have deployed AFT on WeChat Top Stories. The source code is in https://github.com/xiaobocser/AFT.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

*All authors contributed equally to this research. Ruobing Xie is the corresponding author (ruobingxie@tencent.com)

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467176

KEYWORDS

recommender system; multi-domain recommendation; GAN

ACM Reference Format:
Xiaobo Hao, Yudan Liu, Ruobing Xie, Kaikai Ge, Linyao Tang, Xu Zhang, and Leyu Lin. 2021. Adversarial Feature Translation for Multi-domain Recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3447548.3467176

1 INTRODUCTION

Recommendation systems have been widely used in various real-world scenarios for users to get information and entertainment [14]. In recent years, the Matthew effect has spawned some influential super platforms such as Google, Twitter and WeChat. These platforms often have various recommendation scenarios to provide heterogeneous items. For example, the WeChat platform contains multiple recommendation functions on different domains including videos, articles and mini-programs. User behaviors in different recommendation domains could be linked via users' shared accounts after user approval. These multi-domain behaviors should be jointly considered to learn both user's general and domain-specific preferences, which are beneficial for recommendation in all domains.

Multi-domain recommendation (MDR) aims to improve all recommendation domains simultaneously [17]. The key challenge in MDR is how to capture informative domain-specific features from all domains for the target domain. An intuitive method is to jointly combine multi-domain behaviors as input features, while it cares less about the inter-domain feature interactions. Recently, [10, 32] jointly consider multiple user behaviors with attention and graph convolutional networks. Multi-task learning (MTL) is also an effective method to jointly make full use of multi-domain information in different domains, where the recommendation in each domain is treated as a task [15, 16]. A good MDR algorithm should learn the implicit relationships between heterogeneous features and user behaviors in different domains. However, MDR severely suffers from its inherent sparsity issues, which derive from (1) the sparsity of

user-item pair-wise click behaviors, and (2) the sparsity of feature interactions within and between different domains.

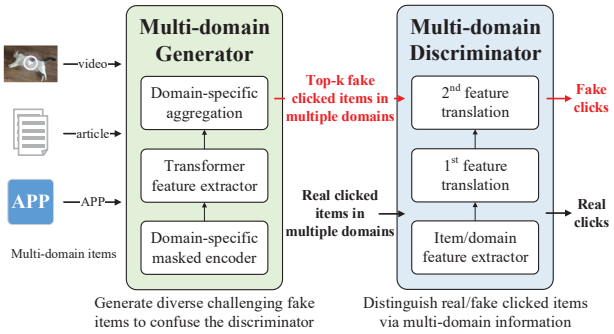

Figure 1: The overall framework of AFT in MDR.

To address these issues and improve all recommendation domains, we propose a novel **Adversarial feature translation (AFT)** framework for MDR. As shown in Fig. 1, AFT highlights useful feature interactions between different domains via a domain-specific masked encoder and a two-step feature translation under a generative adversarial network (GAN) framework. Specifically, in the generator, we first design a domain-specific masked encoder, which merely masks behaviors in the target domain. It forces AFT to learn from informative features in other domains. Next, a transformer extractor and a domain-specific aggregation are conducted to learn user aggregated preferences for the target domain. These domain-specific preferences are then used to generate the fake clicked items which compete with the real clicked items in the discriminator. A reinforcement learning (RL) method is conducted to update the generator. In the discriminator, inspired by the triplet-based training in knowledge representation learning (KRL) [5], we learn item, domain and users' general/domain-specific representations in the same semantic space via two feature translations. These translations explicitly model the interpretable relations among *(item, domain, user)* and *(user general preference, domain information, user domain-specific preference)*. Precisely, we first combine user preferences on items and domains to form the user representation, which reflects the user's general preferences in all domains. Next, we conduct the second feature translation from user's general preferences to user's domain-specific preferences through the target domain's information. The advantages of AFT mainly locate in three aspects: (1) the GAN with masking provides sufficient high-quality negative samples, which alleviate the sparsity and overfitting issues. (2) The domain-specific masked encoder forces AFT to focus more on features in other domains, which is beneficial for MDR. (3) The two explicit feature translations capture interpretable relations between item, domain and users' general/domain-specific representations, which provide a deeper understanding of MDR.

In experiments, we evaluate AFT with competitive baselines on both large-scale private and public datasets in MDR, achieving significant improvements on all domains. We also conduct an online evaluation to verify AFT in online scenarios. Ablation tests and model analyses are also conducted for a better understanding. Currently, AFT has been deployed on multiple recommendation domains in matching of WeChat, which is widely used by millions of users. The main contributions are concluded as follows:

- We propose a novel AFT framework for multi-domain recommendation. To the best of our knowledge, we are the first to combine GAN with feature translation in MDR.
- We first propose a novel domain-specific masked encoder in a GAN training framework to facilitate information flows between different domains in MDR.
- We design a two-step feature translation to highlight feature interaction and transfer among item, domain and user.
- AFT achieves the best performances in both online and offline experiments on all domains. We have also deployed AFT on several real-world recommendation domains.

2 RELATED WORKS

2.1 Recommendation System

Factorization machine (FM) [20] is a classical model for recommendation. With the thrive of deep learning, NFM [8], AFM [26] and AutoFIS [14] combine FM with DNN and attention layers. Autoint [22], BERT4Rec [23], DFN [27], and HRL-Rec [29] further bring in self-attention for feature and behavior level interactions. TDM [34] introduces a tree-based structure for all items, which combine matching and ranking in a single model. Recently, adversarial training has been verified in recommendation [25]. Transfer learning is also adopted to model different items and scenarios [13, 19]. [7] proposes a translation-based recommendation over personalized sequential behaviors. Inspired by these models, AFT combines self-attention layers [22] and ConvE [5] under a GAN framework.

Cross-domain recommendation (CDR) is a related task of MDR. Most CDR works concentrate on improve the target domain's performance with the help of the auxiliary source domain [18]. Recently, embedding and mapping approaches such as EMCDR [18] and SSCDR [12] have been well explored in CDR. CATN [31] also utilizes a transfer network for knowledge transfer. Differing from CDR that focuses on source-target domain adaptation, our MDR task aims to jointly improve *all results* in *multiple domains*.

2.2 Multi-domain Recommendation

Multi-domain recommendation aims to jointly optimize all domains, which can be roughly viewed as a sort of combination of CDR and MTL. MCF [30] considers multiple collaborative filtering (CF) tasks in different domains simultaneously and exploits the relationships between domains. [17] facilitates cross-media content information in MDR. ICAN [28] also focuses on cross-domain field interactions. There are some works that focus on user's multiple behaviors (e.g., click, unclick, purchase). ATRank [32] models multiple types of behaviors via self-attention. MBGCN [10] brings in graph convolutional networks. These models can be used in MDR with the behaviors replaced by the multi-domain behaviors. Multi-task learning is also a classical method for MDR. MDR can be regarded as a special example of MTL, where each domain is viewed as a task. For general MDR, MMoE [16] models the tradeoffs between domain-specific objectives and inter-domain relationships with a new Multi-gate expert strategy, which is more effective when domains are less related. SNR [15] improves MMoE with sub-networks divided from the Shared bottom model, which could better model the high-order

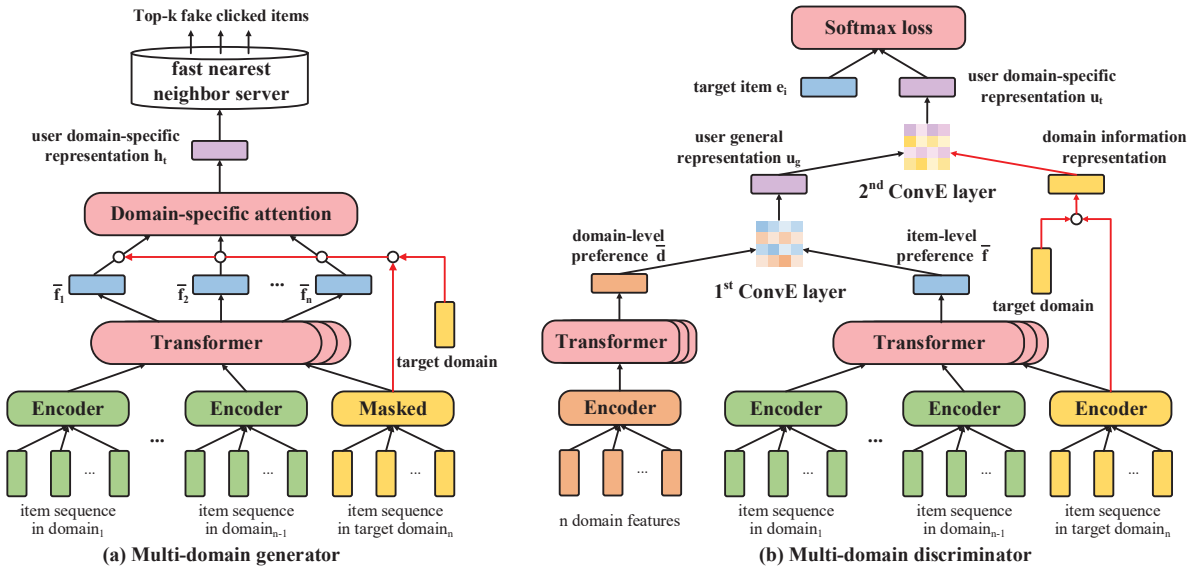

Figure 2: The overall architecture of AFT, which consists of (a) Multi-domain generator and (b) Multi-domain discriminator.

domain-specific interactions. In this work, we implement several competitive multi-task and multi-domain recommendation models as baselines. To the best of our knowledge, we are the first to introduce feature translation with domain-specific masked encoder under the generative adversarial network framework for MDR.

3 MODEL FRAMEWORK

AFT attempts to jointly improve the performances in all domains. In this section, we will first give a brief introduction to the overall architecture of AFT (in Sec. 3.1). Next, we will give detailed discussions on the multi-domain generator and discriminator (in Sec. 3.2 and Sec. 3.3). Finally, we will show its multiple optimization objectives (in Sec. 3.4).

3.1 Overall Architecture

Figure 2 shows the overall architecture of AFT. Precisely, AFT conducts a GAN framework to learn feature relations between different domains. It mainly consists of two modules, namely the multi-domain generator and the multi-domain discriminator. In multi-domain generator, AFT first conducts a domain-specific masked encoder to highlight useful target domain related features from other domains. Second, we use a transformer to model inter-domain feature interactions. Finally, we jointly combine these interacted features via a domain-specific attention to generate the top-k fake clicked items for the target domain. In the multi-domain discriminator, AFT first builds both user's domain-level and item-level preferences from user behaviors with the help of transformers. Second, inspired by the triplet-based feature interaction in KRL [5], we conduct a two-step feature translation to learn user general and domain-specific representations sequentially. Finally, the user domain-specific representation is used to retrieve appropriate items. The discriminator is trained under a multi-domain CTR-oriented objective, while the generator is updated via RL with click-based rewards. We design different models in generator and discriminator to enhance feature interactions for model training.

3.2 Multi-domain Generator

The Multi-domain generator aims to generate item candidates in all domains that users may click. These items are viewed as the "fake" clicked items used in the discriminator. The inputs mainly contain users' click behavior sequences in all domains. Precisely, we set $X = \{X_1, \cdots, X_n\}$ as the multi-domain behavior matrix, where X_i indicates the click sequence of the i-th domain d_i. The *target domain* is the t-th domain that AFT currently recommends for, which is represented as d_t. Fig. 2 (a) shows the multi-domain generator.

3.2.1 Domain-specific Masked Encoder. In this encoder, we attempt to extract useful features from each domain. Inspired by the masked language model in NLP [3], we propose a **domain-specific masked strategy** on the target domain. Precisely, for the target domain d_t, we randomly sample a position set $\texttt{pos}_t$ in the behavior sequence X_t to mask out user behaviors with a certain mask ratio. The masked behaviors in $\texttt{pos}_t$ are replaced with a [MASK] behavior (i.e., zero-padding). Note that we only conduct the masking operation on the target domain, since we want to highlight inter-domain information interactions by decreasing the information in d_t. We generate the masked behavior sequence X_t^{mask} in d_t as:

$$X_t^{\text{mask}} = \text{REPLACE}(X_t, \texttt{pos}_t, [\texttt{MASK}]). \quad (1)$$

The advantages of the masked strategy locate in two aspects: (1) it forces AFT to focus more on other domains when recommending for the target domain, which can enrich inter-domain information interactions between the click behavior in the target domain and the features in other domains. (2) The masked strategy brings in additional variances to user behavior learning in the target domain. It may take more time for the multi-domain generator to converge, but it can also prevent overfitting and improve both robustness and

generalization ability in multi-domain preference modeling. The aggregated feature in the i-th domain $\boldsymbol{f}_i$ is then calculated as:

$$\boldsymbol{f}_i = \text{Average\_pooling}(\boldsymbol{X}'_i), \tag{2}$$

in which $\boldsymbol{X}'_i = \boldsymbol{X}_i^{\text{mask}}$ when the i-th domain is the target domain, and otherwise $\boldsymbol{X}'_i = \boldsymbol{X}_i$. We choose the simple average pooling as the behavior aggregator, making a compromise between efficiency and effectiveness. More complicated encoders are also welcomed.

3.2.2 Multi-domain Feature Aggregator. This module aims to calculate the feature interactions between different domains to get the final users' aggregated preferences for d_t. First, we follow the Transformer layer [24] and take all domains' aggregated features $\{\boldsymbol{f}_1, \cdots, \boldsymbol{f}_n\}$ as inputs to form the interacted features. The i-th interacted feature $\bar{\boldsymbol{f}}_i$ is defined as follows:

$$\bar{\boldsymbol{f}}_i = \text{Transformer}(\boldsymbol{f}_i). \tag{3}$$

Next, we utilize an attention to highlight the target domain information in all interacted features from various domains. The user aggregated preference $\boldsymbol{h}_t$ for domain d_t is formalized as follows:

$$\boldsymbol{h}_t = \text{MLP}(\sum_{i=1}^{n} \alpha_i \bar{\boldsymbol{f}}_i), \quad \alpha_i = \text{Softmax}(\bar{\boldsymbol{f}}_i^\top (\boldsymbol{f}_t + \boldsymbol{d}_t)), \tag{4}$$

where $\boldsymbol{d}_t$ is the target domain's feature, and $\boldsymbol{f}_t$ is the item feature of d_t in Eq. (2). We conduct a two-layer MLP after the domain-specific attention to get $\boldsymbol{h}_t$. This domain-specific attention can be viewed as an amplifier for the target domain d_t. The user aggregated preference $\boldsymbol{h}_t$ is then regarded as the user domain-specific representation in the generator, which derives from all domains' features.

Finally, with the user domain-specific representation $\boldsymbol{h}_t$, the click probability $p_g(e_i|u)$ of item $e_i \in E$ and user u in d_t is defined as:

$$p_g(e_i|u) = \text{Sigmoid}(\boldsymbol{h}_t^\top \boldsymbol{e}_i). \tag{5}$$

The user domain-specific representation $\boldsymbol{h}_t$ is fed into a fast nearest neighbor server such as FAISS [11] to retrieve top-k item candidates of the t-th domain according to $p_g(e_i|u)$. These top-k candidates in different domains are used as "fake" clicked instances to train the discriminator. Following [25, 33], we utilize the top-k nearest items instead of directly regarding the virtual embedding $\boldsymbol{h}_t$ as the "fake" clicked items, for the realistic item embeddings are more challenging for the discriminator to distinguish. Sec. 4.2 gives detailed analyses of the top-k RL-based optimization.

3.3 Multi-domain Discriminator

The Multi-domain discriminator aims to distinguish user's real clicked items from the fake clicked items provided by the generator. We propose a novel two-step feature translation to explicitly model the interpretable relations between user, item and domain. Besides the multi-domain behaviors $\boldsymbol{X}$, we also use the weighted domain sequence $\boldsymbol{D}' = \{\boldsymbol{d}'_1, \cdots, \boldsymbol{d}'_n\}$ as our inputs. Precisely, $\boldsymbol{d}'_i = \beta_i \boldsymbol{d}_i$ indicates the i-th domain feature, while $\beta_i \in [0, 1]$ is a weight that reflects how many items the user have clicked in d_i. $\boldsymbol{X}$ implies the fine-grained item-level user preferences, while $\boldsymbol{D}'$ indicates the coarse-grained domain-level user preferences. Fig. 2 (b) shows the detailed architecture of the multi-domain discriminator.

3.3.1 Item/Domain Encoders. Similar to the generator, we use the average pooling to combine clicked item features in each domain. We do not use the mask strategy in the discriminator, for we want to precisely capture user preferences. We build the aggregated item feature $\boldsymbol{f}_i$ in the i-th domain as follows:

$$\boldsymbol{f}_i = \text{Average\_pooling}(\boldsymbol{X}_i). \tag{6}$$

$\boldsymbol{f}_i$ can be regarded as the item-level user preference for the i-th domain. Next, we conduct two transformer layers for both item and domain feature interactions across domains. We have:

$$\hat{\boldsymbol{f}}_i = \text{Transformer}(\boldsymbol{f}_i), \quad \hat{\boldsymbol{d}}_i = \text{Transformer}(\boldsymbol{d}'_i). \tag{7}$$

These two transformers enable sufficient feature interactions across different domains. We then concatenate the interacted domain/item embeddings in all domains as follows:

$$\bar{\boldsymbol{f}} = \text{Concat}(\hat{\boldsymbol{f}}_1, \cdots, \hat{\boldsymbol{f}}_n), \quad \bar{\boldsymbol{d}} = \text{Concat}(\hat{\boldsymbol{d}}_1, \cdots, \hat{\boldsymbol{d}}_n). \tag{8}$$

Here, $\bar{\boldsymbol{f}}$ and $\bar{\boldsymbol{d}}$ are the final item and domain representations for user u, which indicate user preferences at item and domain levels.

3.3.2 Two-step Feature Translation. In knowledge representation learning, translation-based methods like TransE [2] and convolution-based methods like ConvE [5] transfer the *head entity* $\boldsymbol{e}_h$ into the *tail entity* $\boldsymbol{e}_t$ through the *relation* $\boldsymbol{r}$ to learn knowledge representations (e.g., in TransE we have $\boldsymbol{e}_h + \boldsymbol{r} \approx \boldsymbol{e}_t$). Inspired by the successes of triplet-based transformation, we propose a new two-step feature translation specially designed for MDR with an effective KRL model ConvE [5]. In ConvE, $\boldsymbol{e}_t$ can be translated from $\boldsymbol{e}_h$ via $\boldsymbol{r}$ as:

$$\boldsymbol{e}_t = \text{ConvE}(\boldsymbol{e}_h, \boldsymbol{r}) = \text{Flatten}(\text{CNN}(f(\boldsymbol{e}_h, \boldsymbol{r})))\boldsymbol{W}_c, \tag{9}$$

where $f(\boldsymbol{e}_h, \boldsymbol{r})$ is a 2D reshaping and concatenation function. It first converts the embeddings $\boldsymbol{e}_h, \boldsymbol{r} \in \mathbb{R}^k$ into 2D matrices $\bar{\boldsymbol{e}}_h, \bar{\boldsymbol{r}} \in \mathbb{R}^{k_w \times k_h}$, where $k = k_w k_h$. Next, $\bar{\boldsymbol{e}}_h$ and $\bar{\boldsymbol{r}}$ are concatenated row by row as the same alternating pattern in [5] to get a mixed feature matrix. This matrix is then fed into a 2D convolutional layer and reshape to a vector via the Flatten$(\cdot)$ function and a projection matrix $\boldsymbol{W}_c$. Compared to other 1D feature translation such as the addition in TransE, the 2D convolution enables more sufficient element-wise feature interactions between $\boldsymbol{e}_h$ and $\boldsymbol{r}$ when generating $\boldsymbol{e}_t$, which improves feature translation.

In AFT, we assume that: (1) the user general preference should consist of both item-level and domain-level user preferences that capture user preferences in different granularities. And (2) the user domain-specific preference could be translated from the user general preference via the target domain information. Therefore, similar to the $(\boldsymbol{e}_h, \boldsymbol{r}, \boldsymbol{e}_t)$ triple translation in KRL, we conduct two feature triples for translations noted as *(item-level preference, domain-level preference, user general preference)* and *(user general preference, domain information, user domain-specific preference)*. Formally, we first learn the user general representation $\boldsymbol{u}_g$ as follows:

$$\boldsymbol{u}_g = \text{ConvE}(\bar{\boldsymbol{f}}, \bar{\boldsymbol{d}}). \tag{10}$$

The user general representation consists of both fine-grained item-level preference $\bar{\boldsymbol{f}}$ and coarse-grained domain-level preference $\bar{\boldsymbol{d}}$. We then translate $\boldsymbol{u}_g$ into the user domain-specific representation $\boldsymbol{u}_t$

via the target domain information, which is built from domain-level feature $\boldsymbol{d}_t$ and item-level feature $\boldsymbol{f}_t$ as follows:

$$\boldsymbol{u}_t = \text{ConvE}(\boldsymbol{u}_g, \text{Concat}(\boldsymbol{d}_t, \boldsymbol{f}_t)\boldsymbol{W}_d). \tag{11}$$

Differing from $\boldsymbol{f}_t$ that simply considers user behaviors in the target domain, $\boldsymbol{u}_t$ combines different item-level and domain-level user preferences in all domains for the target domain d_t with two feature translations. Finally, $\boldsymbol{u}_t$ is used to generate a ranking score $p_d(e_i|u)$ with the item candidate e_i. We have:

$$p_d(e_i|u) = \text{Sigmoid}(\boldsymbol{u}_t^\top \boldsymbol{e}_i), \tag{12}$$

$p_d(e_i|u)$ indicates the predicted click probability of item e_i for u in d_t given by the discriminator. An item that is more similar to the user domain-specific representation $\boldsymbol{u}_t$ will get a higher score.

3.4 Optimization Objectives

We conduct a GAN framework to extract domain-specific preferences. The overall loss function of AFT mainly consists of three parts for the generator, discriminator, and dissimilarity.

Discriminator loss. The discriminator loss L_D is the widely-used CTR-oriented objective with the click probability $p_d(e_i|u)$ as:

$$L_D = -\frac{1}{N_{all}}\left(\sum_{S_p} \log p_d(e_i|u) + \sum_{S_n} \log(1 - p_d(e_i|u))\right). \tag{13}$$

S_p and S_n are the positive and negative sets of all domains. $(e_i, u) \in S_p$ indicates that e_i is clicked by u (viewed as a positive sample), while $(e_i, u) \in S_n$ means that e_i is not clicked or is given by the generator (viewed as a negative sample). S_g is the set whose instances are given by the generator. The L_D loss function implies that the discriminator aims to distinguish real clicked items from generated "fake" clicked items. Note that we remove all generated items in S_g which are actually positive instances clicked by the user.

Generator loss. For the generator, since we utilize the *discrete* top-k nearest items as the fake clicked items to train our discriminator, AFT cannot update the generator via the classical minimax game. Instead, we conduct a REINFORCE-based RL method also used in [25], which directly uses the click probability $p_d(e_i|u)$ in discriminator as the reward to train the generator. Precisely, for each action-state pair $(e_i, u) \in S_g$, we set its Q-value $Q(e_i, u)$ as $p_d(e_i|u)$. The loss function of the generator L_G is defined as:

$$L_G = -\frac{1}{N_g}\sum_{S_g} p_g(e_i|u)Q(e_i, u),\ Q(e_i, u) = p_d(e_i|u). \tag{14}$$

$p_g(e_i|u)$ can be regarded as the action probability of e_i given u. The loss function aims to maximize the expected reward of the click probability given by the discriminator.

Dissimilarity loss. Moreover, we further propose a dissimilarity loss L_M to broaden the gap between the generated "fake" clicked item set X_u^d and the real clicked item set Y_u^d for user $u \in U$ and domain $d \in D$. It encourages the generator to provide more diverse negative samples, which are beneficial for the discriminator training (especially essential in the matching module). We use the Maximum Mean Discrepancy (MMD) loss [6] to measure the dissimilarity between fake and real clicked item distributions, noted as:

$$L_{MMD} = -\sum_{u\in U}\sum_{d\in D} \sup_{f\in\mathcal{F}} ||\mathbb{E}[f(X_u^d)] - \mathbb{E}[f(Y_u^d)]||. \tag{15}$$

We use the Gaussian kernel $f \in \mathcal{F}$ in Eq. (15). We can find that the MMD loss L_{MMD} attempts to minimize the supremum of the expectation embeddings' L1 distances between the real and "fake" clicked items of u. It makes the generated items less similar to the real clicked items, which improves the diversity of the multi-domain generator. More discussions on the MMD loss are in Sec. 4.2.

Overall loss. The final loss function of AFT is formalized as:

$$L = \lambda_D L_D + \lambda_G L_G + \lambda_M L_{MMD}. \tag{16}$$

λ_D, λ_G and λ_M are empirically set as 1. We use top-5 fake items for each positive instance to balance L_D and L_G training.

4 ONLINE DEPLOYMENT

4.1 Online Serving

We have deployed AFT on a well-known platform named *WeChat*. WeChat is one of the most popular instant messaging software in China, which has more than 1.2 billion active users per mouth. The WeChat ecosystem contains various (recommendation) functions containing different domains, including *articles*, *long videos*, *short videos*, *programs* and *applications*.

The online architecture of all recommendation domains mainly consists of two modules, including a matching module and a ranking module widely used in industry [4]. Matching aims to fast retrieve hundreds of candidates from the entire item corpus (usually containing million-level items) efficiently, while ranking aims to provide specific item ranks accurately. Precisely, AFT is deployed on the *matching* module as an embedding-based retrieval, since both generator and discriminator will generate a user domain-specific representation that is suitable for fast retrieval. In online, the discriminator first takes all multi-domain behaviors as inputs and outputs the user domain-specific representation $\boldsymbol{u}_t$ as in Eq. (11). Since Eq. (12) implies that the nearest item of the user domain-specific representation has the highest predicted click probability, we can directly use a fast nearest neighbor retrieval tool like FAISS [11] to generate top-N candidates for the following ranking (we set $N = 200$ in our online system). The time complexity of online serving is $O(\log M + T)$, where $\log M$ is the fast retrieval time cost of M item candidates, and T is the computation cost of one forward step in the discriminator. It is acceptable for online serving.

4.2 Discussions on AFT Training Stability

The challenges of conducting a stable AFT training mainly locate in three aspects: (1) AFT uses GAN with RL for model training, which is powerful but not easy to converge [25]. It is essential to meticulously balance both generator and discriminator during training. (2) The inherent sparsity of MDR in both user-item click behaviors and feature interactions between different domains further multiplies the convergence difficulties. (3) Differing from classical GAN-based models in CV and NLP that usually have universal standards (e.g., whether an image indicates a dog), recommendation aims to predict whether a user will click an item, which can only be judged by the user himself. However, most fake clicked items predicted by the generator are not exposed to the user. Directly regarding them as negative samples for the discriminator may bring in false negative issues, since they are not real unclicked instances. On the contrary, users are likely to click similar items as their clicked ones, which

is the theoretical basis of collaborative filtering. Too much attention on distinguishing such "fake" items from real clicked ones will harm the performance, since it may reduce the diversity of negative samples and disturb the overall hyperplane of user preferences.

To solve these challenges and enable a fast and stable model convergence, we propose the following solutions: (1) we use top-5 nearest items instead of the generated virtual item embeddings for discriminator training. In this case, the generator can provide more realistic and diversified fake clicked items to enable an effective and stable GAN training, which also alleviates the sparsity issues in MDR. Moreover, the 5 fake clicked items for each positive instance can well balance the iterative training of the generator and discriminator. (2) We propose the masked encoder that forces the generator to focus more on useful features in other domains, which alleviates the sparsity in inter-domain feature interactions. (3) For the false negative phenomenon in recommendation, we propose the dissimilarity loss L_{MMD} as a penalty term when the fake clicked item is too similar to the real clicked one. It encourages the generator to provide high-quality and diversified negative items, but not exactly the same as positive instances (which are too challenging and may be false negative). It is perfectly suitable for matching, since matching cares whether good items are retrieved in hundreds of items, not the specific top 10 item ranks.

5 EXPERIMENTS

In experiments, we conduct both offline and online evaluations on real-world MDR datasets, aiming to answer the following research questions: (**RQ1**): How does AFT perform in MDR on large-scale public and industrial datasets (see Sec. 5.4)? (**RQ2**): How does AFT perform in real-world multiple online recommendations with various online metrics (see Sec. 5.5)? (**RQ3**): Are different components and losses essential in AFT (see Sec. 5.6)? (**RQ4**): What are the impacts of different parameters on AFT (see Sec. 5.7)?

5.1 Datasets

We evaluate AFT with baselines on both large-scale public and private multi-domain recommendation (MDR) datasets.

Netflix. Netflix Prize dataset is a classical dataset widely used in recommendation [1, 21]. It contains nearly 100 million rating instances of 480 thousand users. Inspired by [15], we regard the movie recommendations of different categories as different domains for MDR. Precisely, we select the *history* and *romance* categories, which contain 3 million rating instances for training and 1.6 million rating instances for evaluation. We consider the user-movie pairs with rating $\geqslant 3$ as the positive instances for MDR, and split these instances into a train and a test set with the chronological order.

MDR-5B. We build a new MDR-5B dataset with 5 heterogeneous recommendation domains in WeChat ecosystem. The recommendation domains include *article*, *long video*, *short video*, *program* and *application*. Heterogeneous items in different domains may have different features. We collect nearly 5 billion user behaviors from randomly selected 44 million users on 1.5 million items. All data are preprocessed via data masking to protect user privacy. The instances of all domains in the former few days are regarded as the train set, while the randomly selected 11.7 million instances in the latter few days are regarded as the test set.

Table 1: Statistics of two large-scale datasets.

Dataset	# domain	# user	# item	# instance
Netflix	2	480K	3.1K	4.6M
MDR-5B	5	44M	1.5M	5.0B

5.2 Competitors

We implement several competitive recommendation models as baselines for MDR, which are categorized into four classes.

Single-domain methods. Single-domain methods only consider user behaviors in the single target domain and learn each domain separately. We conduct three classical recommendation models including FM [20], AFM [26] and AutoInt [22] for this methods, which are widely verified in industry. Moreover, we implement a strong session-based recommendation model BERT4Rec [23]. We use (single) to represent models in their single-domain versions.

Multi-domain methods. These methods jointly consider user behaviors in all domains as features for training. We also conduct FM [20], AFM [26] and AutoInt [22] for multi-domain training. Moreover, we implement a classical multi-behavior model ATRank [32], which jointly aggregates user's multiple behaviors with attention. Note that BERT4Rec cannot be directly used with multiple behaviors. We use (multi) to represent all multi-domain methods.

Multi-task learning. We implement three classical MTL models including Shared bottom, MMoE [16] and SNR[15]. MMoE uses a multi-gate mixture-of-experts strategy, while SNR enhances the Shared bottom with sub-networks to learn inter-domain features.

Negative sampling. The final AFT model is represented as AFT (D+G) with both multi-domain discriminator and generator. To highlight the effectiveness of our multi-domain generator, we implement AFT (D) that only has the discriminator, which uses the unclicked items as negative samples like other baselines. For further comparisons, we replace the generator of AFT with a recent strong negative sampling method hard negative mining (HNM) proposed in Huang et al. (2020). Other ablation versions are in Sec. 5.6.

5.3 Experimental Settings

In AFT, the maximum length of all user historical behaviors is 50. The dimensions of input item and domain features are 64. We conduct a 4-layer 8-head transformer for generator and a 1-layer 16-head transformer for discriminator. The mask ratio is set as 15% in the target domain. The matrix dimensions of two ConvE layers are 16×64. In training, we use Adam with the batch size set as 256. In online, AFT provides top 200 items as the matching results. All parameters are optimized with the grid search. All models share the same settings. Please see Appendix A.1 for more details.

5.4 Offline CTR Prediction (RQ1)

5.4.1 Evaluation Protocol. We conduct the CTR prediction on two datasets. AFT is deployed in the **matching** module, which focuses more on ***whether good items are retrieved*** rather than their specific ranks [4]. Therefore, we use the classical Recall@N metric (r@N) [1, 21] widely used for evaluating matching models as our main metric. It is measured by the average recall of real clicked items given top-N items of users. To simulate the real-world scenario in

Table 2: Offline evaluation on MDR-5B with Recall@N.

Domain	article		long video		short video		application		program		average	
Recall@N	r@50	r@200	r@50	r@200	r@50	r@200	r@50	r@200	r@50	r@200	r@50	r@200
FM (single)	0.081	0.103	0.107	0.223	0.202	0.266	0.189	0.360	0.093	0.204	0.134	0.231
AFM (single)	0.125	0.173	0.224	0.396	0.245	0.346	0.182	0.431	0.131	0.220	0.181	0.313
AutoInt (single)	0.131	0.187	0.242	0.411	0.214	0.393	0.204	0.446	0.158	0.225	0.190	0.332
BERT4Rec (single)	0.093	0.193	0.396	0.613	0.385	0.436	0.215	0.532	0.199	0.351	0.258	0.425
FM (multi)	0.095	0.188	0.130	0.267	0.354	0.359	0.204	0.391	0.156	0.251	0.188	0.291
AFM (multi)	0.141	0.224	0.282	0.495	0.276	0.412	0.233	0.517	0.132	0.287	0.213	0.387
AutoInt (multi)	0.155	0.285	0.276	0.470	0.301	0.465	0.277	0.532	0.225	0.429	0.247	0.436
ATRank (multi)	0.201	0.332	0.382	0.499	0.314	0.545	0.301	0.581	0.275	0.479	0.295	0.487
Shared bottom	0.171	0.309	0.054	0.181	0.009	0.154	0.316	0.665	0.249	0.513	0.160	0.364
MMoE	0.194	0.342	0.208	0.393	0.032	0.381	0.305	0.653	0.255	0.514	0.199	0.456
SNR	0.135	0.211	0.315	0.487	0.152	0.344	0.286	0.575	0.288	0.529	0.235	0.429
AFT (D)	0.163	0.309	0.451	0.609	0.257	0.554	0.303	0.645	0.291	0.547	0.293	0.533
AFT (D+HNM [9])	0.255	0.405	0.496	0.621	0.388	0.584	0.531	0.746	0.463	0.675	0.427	0.606
AFT (D+G)	**0.363**	**0.523**	**0.584**	**0.745**	**0.467**	**0.709**	**0.540**	**0.786**	**0.527**	**0.715**	**0.496**	**0.696**

matching, we choose r@50/r@200 for MDR-5B and r@20/r@50 for Netflix. We also evaluate AFT on AUC as a supplement.

5.4.2 Recall@N Results. Table 2 shows the results on MDR-5B, and Table 3 shows the results on Netflix. We can observe that:

(1) AFT (D+G) achieves the best performances in different r@N of all domains on both Netflix and MDR-5B. These improvements are significant with the significance level $\alpha = 0.01$. It indicates that AFT has successfully learned the implicit laws of feature translations between item, domain and user under the minimax game, and thus can better learn user domain-specific preferences from all domains. AFT has more significant improvements in MDR-5B, since it has more item candidates and thus is more challenging in matching.

(2) The significant improvements derive from both generator and discriminator: (a) the domain-specific adversarial training enables sufficient explorations on all recommendation domains, which is the dominating reason for the huge improvements. The generated challenging and diverse fake clicked items can spur the discriminator to have a better overall understanding of user preferences, which is especially essential for matching. (b) The domain-specific masked encoder in generator also compulsorily enriches the inter-domain information interactions between the click behaviors in the target domain and the features in other domains by masking behaviors in the target domain. It can alleviate the serious inherent sparsity issue in under-trained domains. (c) The two-step feature translation in discriminator can better extract informative and interpretable translation rules among user, item and domain, which are beneficial for jointly using multi-domain information. We will give detailed ablation tests and analyses in Sec. 5.6.

(3) AFT (D+G) outperforms both AFT (D) and AFT (D+HNM) by a large margin in MDR-5B. It indicates that our generator with domain-specific masked encoder could provide more informative negative samples to polish the discriminator's training. The improvements over AFT (D+HNM) reflect the advantages of our domain-specific generator compared to strong negative sampling strategies [9] (note that HNM also benefits from the multi-domain modeling in AFT). Moreover, all AFT models perform better than other baselines in overall, which verifies the power of the two-step feature translation in discriminator. In general, MTL methods and multi-domain methods have better performances. Some baselines cannot receive consistent improvements on all domains. It indicates the significance of multiple behaviors in MDR matching.

Table 3: Offline evaluation on Netflix with Recall@N.

Domain	Romance		History	
Recall@N	r@20	r@50	r@20	r@50
FM (single)	0.338	0.432	0.588	0.656
AFM (single)	0.455	0.611	0.751	0.846
AutoInt (single)	0.473	0.610	0.711	0.833
BERT4Rec (single)	0.698	0.770	0.891	0.939
FM (multi)	0.393	0.484	0.609	0.689
AFM (multi)	0.487	0.624	0.756	0.852
AutoInt (multi)	0.589	0.683	0.788	0.920
ATRank (multi)	0.613	0.702	0.822	0.935
Shared bottom	0.749	0.837	0.878	0.929
MMoE	0.776	0.860	0.867	0.921
SNR	0.704	0.798	0.897	0.955
AFT (D)	0.755	0.843	0.922	0.971
AFT (D+HNM [9])	0.767	0.864	0.924	0.973
AFT (D+G)	**0.784**	**0.872**	**0.926**	**0.977**

5.4.3 AUC Results. In offline evaluation, Recall@N is viewed as the most essential metric, since we deploy AFT on matching. To verify AFT from different aspects, we also conduct an evaluation with AUC, which reflects the probability of clicked items ranking higher than unclicked items. Table 4 shows the AUC results of AFT and representative baselines on three domains of MDR-5B (for a better display). We can observe that: (1) AFT (D+G) still achieves the best

performances on all domains, significantly outperforms all baselines including AFT (D+HNM). The improvements on AUC indicate that AFT can well distinguish unclicked items from clicked ones, which can also be used in ranking. (2) AFT (D+HNM), ATRank and SNR achieve relatively good results among baselines, which shows that multiple behaviors are essential. In general, the conclusions of AUC and Recall@N are basically consistent.

Table 4: Offline evaluation on MDR-5B with AUC.

Domain	article	long	short
AFM (single)	0.636	0.677	0.652
AutoInt (single)	0.641	0.711	0.646
AFM (multi)	0.650	0.725	0.659
AutoInt (multi)	0.652	0.723	0.670
ATRank (multi)	0.667	0.741	0.678
SNR	0.645	0.735	0.621
AFT (D+HNM [9])	0.699	0.755	0.713
AFT (D+G)	**0.714**	**0.797**	**0.720**

5.5 Online Evaluation (RQ2)

5.5.1 Evaluation Protocol. We further conduct an online A/B test to evaluate AFT in real-world scenarios. We have deployed AFT on multiple recommendation scenarios in WeChat as stated in Sec. 4. In online evaluation, we give the results of two integrated recommendation feeds in *WeChat Top Stories* and *WeChat Reading* as examples. Both feeds contain three heterogeneous domains including *article*, *long video* and *short video*. Precisely, AFT is deployed as an additional retrieval channel in the matching module, with the ranking and other matching modules unchanged. The online matching baseline is an ensemble model containing multiple single-domain, multi-domain and rule-based methods. In the online A/B test, we focus on two metrics including the CTR and average click number per capita (ACN) to measure online accuracy. We conduct the A/B test for 7 days with nearly 50 million users involved.

5.5.2 Experimental Results. We report the improvement percentages of AFT in Table 5, from which we can know that:

(1) AFT achieves significant improvements on all CTR and ACN metrics in two scenarios with the significance level $\alpha = 0.01$. It confirms the power of the generator with domain-specific masked encoder and the two-step feature translation in online systems.

(2) In WeChat Top Stories, CTR and ACN of cold-start users have more significant improvements. It indicates that AFT could well translate user preferences from other domains to the target domain, which alleviates the sparsity. Therefore, AFT is especially helpful for cold-start users that have fewer historical behaviors.

(3) WeChat Reading is a new integrated recommendation feed, where all domains are viewed as cold-start domains. AFT can jointly learn user domain-specific preferences from user behaviors in mature domains, which benefits the cold-start domains.

5.5.3 Recommendation Trial. Real-world recommendation of CTR-oriented objectives usually struggles with the homogenization issue. To improve the diversity and discover user's potential interests, we add a recommendation trial channel in matching, which provides

Table 5: Online A/B tests on WeChat MDR systems.

Scenario	All users		Cold-start users	
	CTR	ACN	CTR	ACN
WeChat Top Stories	+0.51%	+0.48%	+1.20%	+1.08%

Scenario	Main feed		Push feed	
	CTR	ACN	CTR	ACN
WeChat Reading	+2.16%	+1.67%	+13.71%	+9.27%

new items not in users' existing interests. It is similar to the exploration in RL. Specifically, we utilize the fake clicked items provided by the generator as the recommended results. These items are recommended by the generator but different from the real clicked ones due to the dissimilarity loss L_M, which are both high-quality and diversified. In online evaluation, we find that AFT achieves 8.51% improvements on the impression percentage of items with new interests (noted by new taxonomies), and 168.45% astonishing CTR improvements in the recommendation trial channel compared with a simple diversified exploration baseline.

5.6 Ablation Tests (RQ3)

We aim to verify the effectiveness of different components in AFT via ablation tests. We implement several ablation versions of AFT without the generator, dissimilarity loss and feature translation (replaced with attention), and report Recall@200 of four domains on MDR-5B in Table 6. We can find that: (1) all components are indispensable in AFT, among which the generator is the dominating reason for improvements. The first two ablation settings verify the importance of our GAN and domain-specific masked encoder. It encourages AFT to consider more information from other domains to generate informative negative samples, which polishes the discriminator's training. (2) The feature translation also outperforms attention-based aggregation by a large margin, which confirms that our feature translation is very suitable for MDR. (3) The dissimilarity loss broadens the gaps between fake and real clicked items for MDR. It makes the generator more diversified (which is beneficial for the discriminator) and the training more robust.

Table 6: Ablation Tests on the MDR-5B dataset.

Domain	article	long	short	app
AFT	**0.523**	**0.745**	**0.709**	**0.786**
– generator	0.309	0.609	0.554	0.645
– domain-specific mask	0.371	0.634	0.628	0.753
– feature translation	0.359	0.673	0.524	0.581
– dissimilarity loss	0.388	0.741	0.597	0.687

5.7 Model Analyses (RQ4)

5.7.1 Analysis on the Mask Ratio. The mask ratio of the domain-specific masked encoder determines the degree to which the generator pays attention to other domains' features when predicting for the target domain. Fig. 3 (a) shows the r@200 on MDR-5B with

different mask ratios. We can find that AFT achieves the best performance when the mask ratio is set as 15%. A higher mask ratio wipes out more user behaviors in the target domain and encourages more concentrations on other domains, which improves the challenges as well as inter-domain interactions in the fake clicked items. Too low mask ratios may not make this module function well. On the contrary, too high mask ratios will prevent AFT to learn from the target domain. At last, we select 15% as our mask ratio.

5.7.2 Analysis on the Number of Fake Clicked Items. We also evaluate the impacts of different numbers of fake clicked items given by the generator. From Fig. 3 (b) we can find that: the r@200 first increases and then decreases as the number of generated negative samples increases from 1 to 20. AFT achieves the best performance when we use top-5 fake clicked items for the discriminator. It is because that too few fake clicked items will lead to an insufficient exploration of diversified negative samples, while too many fake clicked items will dilute the attention to positive instances. Both model analyses with different parameters also verify the stability of AFT in model convergence.

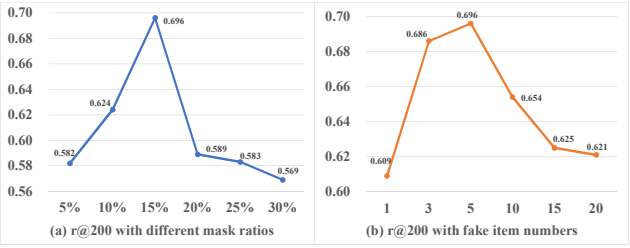

Figure 3: Results of model analyses: (a) AUC with mask ratios. (b) AUC with different fake clicked item numbers.

6 CONCLUSION AND FUTURE WORKS

We propose a novel AFT for MDR, which designs a domain-specific masked encoder and a two-step feature translation to highlight the inter-domain feature interactions under an adversarial training framework. Offline and online evaluations verify the effectiveness of AFT. AFT has been deployed online, affecting millions of users.

In the future, we will use more sophisticated models for explicit feature translation, and explore more fine-grained and interpretable domain interactions. We will also combine other domain adaptation and pre-training models with AFT for better performances.

REFERENCES

[1] Robert M Bell and Yehuda Koren. 2007. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Proceedings of ICDM*.
[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of NIPS*.
[3] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *Proceedings of ICLR*.
[4] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of RecSys*.
[5] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of AAAI*.
[6] Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. 2015. Training generative neural networks via maximum mean discrepancy optimization. *UAI* (2015).
[7] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based recommendation. In *Proceedings of RecSys*.
[8] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of SIGIR*.
[9] Jui-Ting Huang, Ashish Sharma, Shuying Sun, Li Xia, David Zhang, Philip Pronin, Janani Padmanabhan, Giuseppe Ottaviano, and Linjun Yang. 2020. Embedding-based retrieval in facebook search. In *Proceedings of KDD*.
[10] Bowen Jin, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Multi-behavior recommendation with graph convolutional networks. In *SIGIR*.
[11] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Trans. Big Data* (2019).
[12] SeongKu Kang, Junyoung Hwang, Dongha Lee, and Hwanjo Yu. 2019. Semi-supervised learning for cross-domain recommendation to cold-start users. In *Proceedings of CIKM*.
[13] Ying Li, Jia-Jie Xu, Peng-Peng Zhao, Jun-Hua Fang, Wei Chen, and Lei Zhao. 2020. ATLRec: An Attentional Adversarial Transfer Learning Network for Cross-Domain Recommendation. *JCST* (2020).
[14] Bin Liu, Chenxu Zhu, Guilin Li, Weinan Zhang, Jincai Lai, Ruiming Tang, Xiuqiang He, Zhenguo Li, and Yong Yu. 2020. AutoFIS: Automatic Feature Interaction Selection in Factorization Models for Click-Through Rate Prediction. In *Proceedings of KDD*.
[15] 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 AAAI*.
[16] 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 KDD*.
[17] Weizhi Ma, Min Zhang, Chenyang Wang, Cheng Luo, Yiqun Liu, and Shaoping Ma. 2018. Your Tweets Reveal What You Like: Introducing Cross-media Content Information into Multi-domain Recommendation.. In *Proceedings of IJCAI*.
[18] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-Domain Recommendation: An Embedding and Mapping Approach.. In *Proceedings of IJCAI*.
[19] Cong Phuoc Huynh, Arridhana Ciptadi, Ambrish Tyagi, and Amit Agrawal. 2018. Craft: Complementary recommendation by adversarial feature transform. In *Proceedings of ECCV*.
[20] Steffen Rendle. 2010. Factorization machines. In *Proceedings of ICDM*.
[21] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I Nikolenko. 2020. RecVAE: A New Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *Proceedings of WSDM*.
[22] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of CIKM*.
[23] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of CIKM*.
[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS*.
[25] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of SIGIR*.
[26] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *Proceedings of IJCAI*.
[27] Ruobing Xie, Cheng Ling, Yalong Wang, Rui Wang, Feng Xia, and Leyu Lin. 2020. Deep Feedback Network for Recommendation. In *Proceedings of IJCAI*.
[28] Ruobing Xie, Zhijie Qiu, Jun Rao, Yi Liu, Bo Zhang, and Leyu Lin. 2020. Internal and Contextual Attention Network for Cold-start Multi-channel Matching in Recommendation. In *Proceedings of IJCAI*.
[29] Ruobing Xie, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. 2021. Hierarchical Reinforcement Learning for Integrated Recommendation. In *Proceedings of AAAI*.
[30] Yu Zhang, Bin Cao, and Dit-Yan Yeung. 2010. Multi-domain collaborative filtering. In *Proceedings of UAI*.
[31] Cheng Zhao, Chenliang Li, Rong Xiao, Hongbo Deng, and Aixin Sun. 2020. CATN: Cross-domain recommendation for cold-start users via aspect transfer network. In *Proceedings of SIGIR*.
[32] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. ATRank: An attention-based user behavior modeling framework for recommendation. In *Proceedings of AAAI*.
[33] Fan Zhou, Ruiyang Yin, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Jin Wu. 2019. Adversarial point-of-interest recommendation. In *Proceedings of WWW*.
[34] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning Tree-based Deep Model for Recommender Systems. In *Proceedings of KDD*.

A APPENDIX

A.1 Implementation Details and Motivations

AFT mainly consists of a generator and a discriminator. We use transformer extractors in both generator and discriminator since they can effectively learn feature interactions between different domains. In the two-step feature translation, inspired by the knowledge representation learning method that learns with triples *(head entity, relation, tail entity)*, we also use similar feature translations via *(item-level preference, domain-level preference, user general preference)* and *(user general preference, domain information, user domain-specific preference)* to model interpretable relations between item, domain and user. We use ConvE since it is efficient and effective. Other KRL models could also be easily adopted in the two-step feature translation. All item and domain embeddings are randomly initialized. Note that the "translation" in Adversarial feature translation indicates that "the process of moving something from one place to another", not limited to mathematical translation with the strict linear transformation.

In Sec. 4.2, we bring in detailed discussions on the training challenges and solutions of AFT. The MMD loss aims to broaden the distance between the expectations of fake clicked and real clicked item embedding distributions, which has been widely used in other works. The key hyper-parameter to keep a fast and stable convergence is the number of fake clicked items provided by the generator. We combine these generated items with real unclicked items as negative samples. We have tested the fake clicked item numbers among $\{1, 3, 5, 10, 15, 20\}$, and select the best 5 in the final version. For model parameters, we introduce most of the essential hyper-parameters in Sec. 5.3, including embedding dimensions, learning parameters and weights. We use a grid search for parameter selection. Note that we cannot choose too large embedding dimensions for our model, considering the online memory and computation costs. The final parameters will be listed in the released version of the source codes. We implement AFT with multi-core GPU, using Tensorflow 1.4.1 with Linux. The amount of memory is 60G.

A.2 Evaluation Details

In offline evaluation, we conduct five runs and report the average results. We also conduct significance tests to verify that the improvements are significant. For the evaluation metric, we mainly concentrate on Recall@N since we deploy AFT on the matching module, which cares whether good items are retrieved in top N items (N is usually set as several hundred in real-world large-scale recommendation systems). We use Recall@50 and Recall@200 for WeChat MDR dataset, and Recall@20 and Recall@50 for Netflix. Similar evaluation with Recall@N on Netflix is also reported in Shenbin et al. [21]. We also consider AUC as a supplement to evaluate the overall ranking ability. We select several strong baselines of all four different method types and report their AUC results in three domains. We highlight that AFT achieves the best performances on all five domains of WeChat measured by Recall@N and AUC. In general, the conclusions of AUC and Recall@N are basically consistent.

In ablation study, *AFT - generator* indicates the AFT version that only uses the discriminator with real unclicked items as negative samples, which is equivalent to the AFT (D) version. *AFT - domain-specific mask* indicates that AFT keeps the generator but removes the domain-specific masked encoder. *AFT - feature translation* indicates that we replace the two-step feature translation with a domain-specific attention-based aggregation. *AFT - dissimilarity loss* indicates that we only use L_D and L_G in training. Other components remain unchanged in all ablation settings.