An Empirical Study of Selection Bias in Pinterest Ads Retrieval

Yuan Wang[*][†]
ywang4@scu.edu
Santa Clara University
Santa Clara, California, USA

Peifeng Yin[†]
pyin@pinterest.com
Pinterest
San Francisco, California, USA

Zhiqiang Tao
zhiqiang.tao@rit.edu
Rochester Institute of Technology
Rochester, New York, USA

Hari Venkatesan
hvenkatesan@pinterest.com
Pinterest
San Francisco, California, USA

Jin Lai
jinlai@pinterest.com
Pinterest
Seattle, Washington, USA

Yi Fang
yfang@scu.edu
Santa Clara University
Santa Clara, California, USA

PJ Xiao[‡]
pxiao@pinterest.com
Pinterest
San Francisco, California, USA

ABSTRACT

Data selection bias has been a long-lasting challenge in the machine learning domain, especially in multi-stage recommendation systems, where the distribution of labeled items for model training is very different from that of the actual candidates during inference time. This distribution shift is even more prominent in the context of online advertising where the user base is diverse and the platform contains a wide range of contents. In this paper, we first investigate the data selection bias in the upper funnel (Ads Retrieval) of Pinterest's multi-cascade ads ranking system. We then conduct comprehensive experiments to assess the performance of various state-of-the-art methods, including transfer learning, adversarial learning, and unsupervised domain adaptation. Moreover, we further introduce some modifications into the unsupervised domain adaptation and evaluate the performance of different variants of this modified method. Our online A/B experiments show that the modified version of unsupervised domain adaptation (MUDA) could provide the largest improvements to the performance of Pinterest's advertisement ranking system compared with other methods and the one used in current production.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; **Recommender systems**.

KEYWORDS

Selection Bias, Pre-Ranking System, Online Advertising

[*]The work was done during the contractorship at Pinterest.
[†]Both authors made equal contribution to the work.
[‡]PJ Xiao is the corresponding author.

KDD '23, August 6–10, 2023, Long Beach, CA, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0103-0/23/08.
https://doi.org/10.1145/3580305.3599771

ACM Reference Format:
Yuan Wang, Peifeng Yin, Zhiqiang Tao, Hari Venkatesan, Jin Lai, Yi Fang, and PJ Xiao. 2023. An Empirical Study of Selection Bias in Pinterest Ads Retrieval . In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3580305.3599771

1 INTRODUCTION

Pinterest is a visual discovery platform that allows users to discover and save ideas for various interests such as fashion, home decor, and travel. It has become a popular destination for users to search for and discover new products, ideas, and inspiration. As a result, it has also become an attractive advertising platform for businesses looking to reach and engage with their target audience. To support the growing demand for online advertising, Pinterest has developed a large-scale advertisement serving platform using the multi-cascade ranking system [12] to deliver the most relevant ads to users.

Like many other online advertising platforms, this multi-cascade recommendation system contains several stages to filter and rank ads based on various business logic and modeling signals. As shown in Figure 1, a typical ads serving system has four main stages: Ads Targeting, Ads Retrieval, Ads Ranking, and Ads Auction.

- Ads Targeting is the very first stage. At this stage, it only selects the ads that meet the targeting criterion preset by advertisers.
- Ads Retrieval is the second stage right after Ads Targeting. In this stage, various mechanisms including Retrieval models (the models used in the Retrieval stage) are used to select a smaller subset of ad candidates out of the millions of candidates received from the Targeting stage. Selected ad candidates are passed down to the Ads Ranking stage for more comprehensive scoring and ranking.
- In the Ads Ranking stage, a set of sophisticated models is developed to accurately score the specific objectives (i.e. CTR, CVR, Relevance etc.) of each ad candidate selected at the Retrieval stage. The model prediction in this stage will directly impact many key aspects, such as the quality of delivered ads. As a result, this stage is only able to score a very limited number of

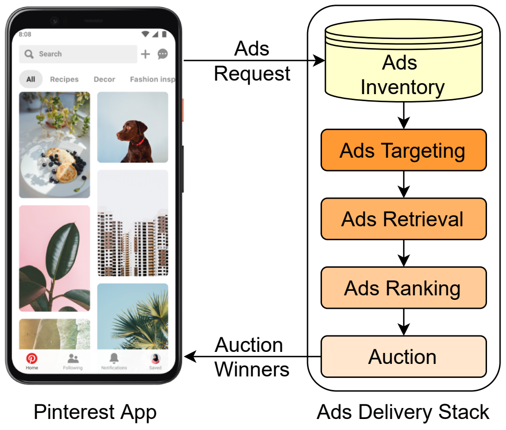


Figure 1: The life cycle of online ads delivery. At high level, an ads request is triggered when a user opens the Pinterest app or starts a new session, and the ads request will be sent to the ads delivery system to query for a dozen of ads. In the ads delivery backend, ad candidates in the inventory will flow through various stages like Targeting, Retrieval, Ranking, and Auction, which sends the auction winners back to the mobile app, where the selected ads will be visible to the user.

ad candidates. This is because it spends much more of the allotted time budget to score each ad candidate, using very complex and performant models, to ensure the prediction accuracy.

- Ads Auction is the last stage in the serving stack. The main objective here is to make the final decision of each auction candidate: 1) whether this candidate should be delivered to the user; 2) which position in the targeting surface should this candidate be inserted into. Afterwards, the winning candidates will be delivered to the user's device and inserted into the corresponding position, where the user will see the ads and respond to these ads with various user actions.

As discussed above, the Ads Retrieval is the second stage of the delivery system, and it is responsible for retrieving the most valuable ads from a large set of ad candidates for each query. The goal of this stage is to retrieve all relevant ads, while also minimizing the number of irrelevant or low-quality ones. This requires the use of machine learning models that can efficiently predict the relevance and quality of ads candidate based on a variety of features and signals. It has been a difficult problem for Retrieval stage to efficiently fulfill this mission due to several key challenges:

- The selected subset of candidates have to be of high quality to avoid wasting the capacity of expensive full ads ranking on the low quality ads;
- The size of selected candidates has to be small enough such that subsequent comprehensive ranking at Ads Ranking stage can handle these ad candidates;
- Retrieval models are required to score and rank the post Targeting ad candidates in the order of millions;
- Retrieval models will not be accessible to a lot of ML signals, especially the expensive real-time ones and will also not be able to leverage sophisticated model architectures due to the scalability consideration discussed in the previous point.

As a result, building performant Retrieval models under these constraints is a challenging problem in the machine learning domain. Currently, the Retrieval models in most ads platforms use the two-tower model architecture proposed by Covington et al. [5]. Among all the challenges associated with Retrieval model development and optimization, selection bias in the training data has been a long-lasting problem impairing the performance of these models.

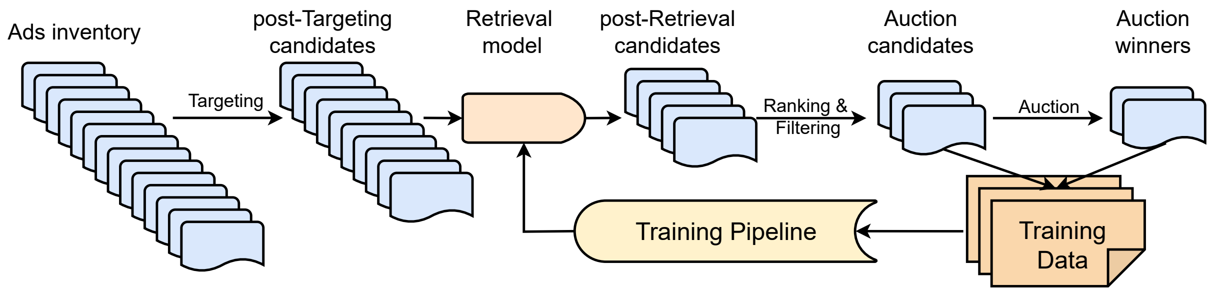
In this paper, we focus on the issue of data selection bias in the Ads Retrieval stage of Pinterest's multi-cascade ads ranking system. The training data used to train the model reflects not only real user preferences, but it also includes the production model's personalized recommendations. This means that the training data is not representative of the overall population of advertisements, which can lead to inaccurate results. In addition, the distribution discrepancy between the training data (with observed user actions as true labels) and the inference data (composed by the ad candidates after the Targeting stage) can further impact the model performance.

To address data selection bias in the Ads Retrieval funnel, we first investigated the data distribution across various types of ad candidates datasets, and we further assessed various ML techniques including Unsupervised Domain Adaptation (UDA) [22] to improve the performance of Retrieval models. As the number of ad candidates with real user action is small, it will be beneficial for the model training to leverage the unlabeled ad candidates data, particularly the ones with similar distribution as the inference data. One difficulty with this model training strategy is determining how to effectively use these unlabeled data points, which have more consistent distribution as compared to the model inference data. In this paper, we have leveraged various state-of-the-art (SoTA) methods to incorporate unlabeled data in training Retrieval models. Additionally, we developed a modified version of UDA (MUDA) to improve the performance of naive implementation of UDA in the Retrieval model training. Our online experimental results show that a couple of methods could potentially improve the performance of the ads ranking system, as compared to the knowledge distilled model in the current production environment and a few other methods. Thus, our contribution could be summarized as the following:

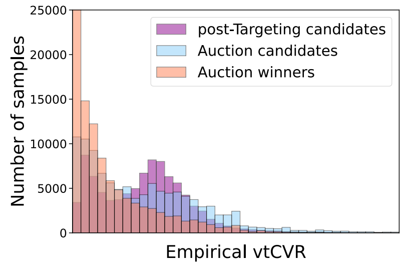
- We identified and characterized the selection bias issue in the upper funnel of the multi-cascade advertisement recommendation system.
- We surveyed a series of SoTA modeling strategies and evaluated their performance in both offline and online settings.
- We further proposed a modified version of Unsupervised Domain Adaptation (MUDA) that provides the best online performance among all the modeling strategies we have examined, and the online experiments show that MUDA also outperforms the current production model.

2 SELECTION BIAS IN PINTEREST ADS

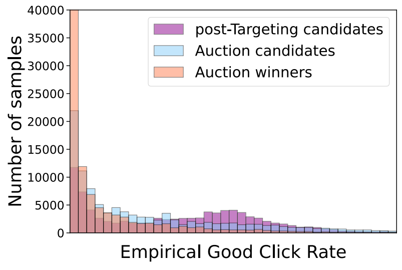
As illustrated in Figure 1, Pinterest's ads serving system consists of four stages: Ads Targeting, Ads Retrieval, Ads Ranking, and Ads Auction. Each stage scores and/or filters ad candidates based on the request and ads content features. Given an ad request, Ads Retrieval narrows down millions of ad candidates to a couple of thousands. These candidates are then sent to Ads Ranking for further accurate



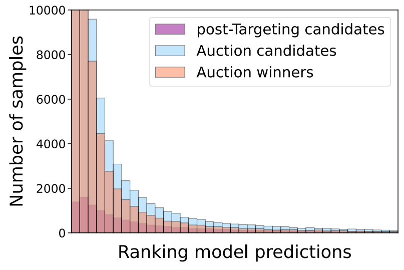
(a)



(b)



(c)



(d)

Figure 2: Distribution of features and labels across three ads datasets related to Retrieval modeling. (a) shows the flow of major ad candidates along the ads delivery funnel. (b) shows the distribution of Empirical vtCVR (one of key Retrieval model's features) across three datasets for Retrieval training/serving. (c) shows the distribution of Empirical Good Click Rate (one of key Retrieval model's features) across three datasets for Retrieval training/serving. (d) shows the distribution of the Ranking model predictions (used as the pseudo label in Retrieval model training) across three datasets. Note that the exact values on x-axes are hidden for confidentiality reasons.

prediction of user action as well as filtering. Finally we run Ads Auctions on survivors and determine auction winners based on a predefined utility function and advertiser's bid.

In the Retrieval stage, the latency limit is crucial because of the large number of ad candidates in the database. We adopt a two-tower DNN structure [5], where candidate embedding could be computed offline. During serving, the model will produce the score of each ad candidate by calculating the dot-product between the precomputed candidate embedding and the query embedding computed on-the-fly for each request.

2.1 Datasets and Training Pipeline

As mentioned earlier, the ads serving system consists of Targeting, Retrieval, Ranking, and Auction. As shown in figure 2a, millions of candidates in the ads inventory will flow through various stages across the ads delivery funnel, and only a small set of valuable ads will survive and be delivered to users. Specifically, the initial ads inventory candidates will be selected through Ads Targeting to refine the set of ad candidates (a.k.a post-Targeting candidates), which will then be scored and ranked by Retrieval models. After being selected by Retrieval models, the survivors (post-Retrieval candidates) will be further filtered or selected by various business logics and models in the Ranking stage. This leads to a new set of ad candidates (a.k.a Auction candidates), which will be evaluated in the Auction stage. The Auction stage will pick a dozen of winners out of the auction candidates and deliver these final survivors (a.k.a Auction winners) to Pinterest's users. For existing Retrieval models, two types of training data are collected — Auction candidates and Auction winners. The latter dataset includes observed user actions as true labels, and the former one includes ranking model predictions as pseudo labels.

The Ranking model predictions are used in the Auction stage to determine the winning ads from the auction candidates pool. Currently, we use these Ranking model predictions as pseudo labels to train Retrieval models, with the aim to maximize the funnel efficiency to deliver the most valuable ad candidates to Pinterest users. To ensure the model freshness, Retrieval models are continuously trained and evaluated on a daily basis. Specifically, the model snapshot trained on day $X - 1$ data is loaded to train on day X data, and the newly trained model is evaluated on day $X + 1$ data. This daily training setup enables the model to capture the most recent patterns, keeping it responsive to new trends. The second-day evaluation allows for detection of possible overfit and abnormal behavior before serving production traffic.

2.2 Selection Bias

As mentioned above, Retrieval models are currently trained on both Auction candidates and Auction winners, where the Ranking model predictions are used as pseudo labels. Such setup inevitably introduces the data with selection bias, particularly the inconsistency in the dataset between training and serving [18]. In serving time, however, the model needs to make predictions on the post-Targeting ad candidates. As Auction candidates and winners are a small subset of post-Targeting candidates (generated through various business logics and Ranking models), the distribution of these datasets will be inconsistent between model training and serving.

Figure 2a illustrates the concept of inconsistency on the ads datasets used in training and inferencing in the cycle of the Retrieval models. To further demonstrate the bias, we analyzed the distributions of pseudo labels and two important Retrieval model features across three different datasets: post-Targeting candidates, Auction candidates, Auction winners. Figure 2b, 2c, and 2d demonstrates that the distributions are different across all three datasets, and this distribution difference is much more significant between the two datasets used in current Retrieval model training and the one used in Retrieval model serving.

For simplicity, in the rest of this paper, we will interchangeably use the following terms: post-Targeting candidates, serving datasets for Retrieval models, and unbiased dataset.

2.3 Problem Formulation

For simplicity, we represent each data record as a tuple of three elements: (u, a, y):

- u: the feature of a request, containing user profile features and context features (e.g., search term if from Search surface),
- a: the advertisement candidate features,
- y: the groundtruth label, i.e., observed user actions.

Additionally, let $<\mathbb{U}, \mathbb{A}>$ represent distribution of request features, advertisement features in inventory and $\mathbb{D} = \mathbb{U} \times \mathbb{A}$ represent the full distribution of all request and ad candidates pairs. Finally, let F_θ and l represent the model with trainable parameter θ and the loss function we want to minimize:

- $\mathrm{F}(u, a) \rightarrow \mathbb{R}$: a model maps the request and candidate features to a numeric value,
- $l(y, y) \rightarrow \mathbb{R}$: a function maps two numeric values to a scalar (the loss value).

Ideally we want to minimize the training loss on unbiased data, i.e.,

$$\min_\theta \mathcal{L}_{ideal}(\mathrm{F}_\theta) = \frac{1}{|\mathbb{D}|} \sum_{(u,a) \in \mathbb{D}} l(y, \mathrm{F}_\theta(u, a)). \tag{1}$$

In reality, it is impossible to calculate the above loss function on the unbiased dataset, as the true labels are not available. As a result, we have to leverage the biased dataset whose true labels are available to us. In the next section, we will describe a series of methods to use both biased and unbiased datasets to build a model to score the post-Targeting ad candidates in our system.

3 SOLUTION

3.1 Naive Method: Binary Classification

The naive method is to train a simple classification model in the common way, i.e., training a click classification model based on the dataset with observed user actions, where the ones with user clicks are treated as positive examples and the ones with no clicks are treated as negatives. In this naive method, we will optimize the following loss function:

$$\min_\theta \mathcal{L}_{naive}(\mathrm{F}_\theta) = \frac{1}{|\mathbb{O}|} \sum_{(u,a) \in \mathbb{O}} l(y, \mathrm{F}_\theta(u, a)). \tag{2}$$

The dataset $\mathbb{O}$ denotes the set of request and auction winners pairs, where there are observed user actions.

3.2 In-batch Negative Classification

Similar to the naive classification method, we will build a classification model based on the biased dataset with observed user actions as the true labels. In the real-world advertising system, the viewed ads without user clicks are not necessarily reliable negative examples, Users could still find these ads to be valuable even if they did not take actions on them at that moment. Different from the naive classification method, we generate negative examples by introducing ad candidates from the other requests in the same training batch as the current request following the common setup [6, 10, 23, 25]. Specifically, only the delivered ads with user clicks are included in training data, and clicked ads in different requests in the same batch are treated as negative examples.

3.3 Knowledge Distillation

Ranking models are trained with complex architectures and numerous input features. In contrast, Retrieval models have to limit the architecture to two-tower DNN as well as available features due to the demanding requirement of scalability and low serving latency. To minimize the performance loss, knowledge distillation (kd) [8] is adopted, which means Retrieval models are trained with Ranking model's predictions as pseudo labels. Formally, with R denoting the Ranking model, we optimize the following loss function:

$$\min_\theta \mathcal{L}_{kd}(\mathrm{F}_\theta) = \frac{1}{|\mathbb{O}|} \sum_{(u,a) \in \mathbb{O}} l(\mathrm{R}(u, a), \mathrm{F}_\theta(u, a)). \tag{3}$$

3.4 Transfer Learning

The core idea of transfer learning is to train a model on source domain data and then fine tune part of its parameters on the target domain. Particularly for a DNN model, the early layers are usually fixed during fine tuning as they are shown to represent primitive and general features [14]. In our case, the Retrieval model is a two-tower DNN, and the data distribution discrepancy across different datasets is only from the ad candidates. As a result, we use the unbiased data to fine tune the ad's embedding tower, and keep the query tower unchanged.

3.5 Adversarial Regularization

Another view of bias issue is that the representation learned from biased data is not general enough to be applied to the unbiased dataset, leading to a performance degradation. We can therefore add regularization on the learning so that the intermediate output of the model has no information indicating its data source, a technique known as Adversarial (adv) Learning [7].

For a DNN model, we can split it into two parts. The former one takes the raw input and gives the intermediate output. The latter one takes the intermediate output and gives the final prediction. The Adversarial regularization trains a data source classifier from the intermediate output, the negative of whose loss function is then added to the original one as regularization.

Formally, let F_1 and F_2 denote such two parts of DNN, while the H denotes the classifier. The loss function of data source classifier is defined as Equation (4).

$$\mathcal{L}_{cls}(u, a) = -\mathbf{1}_{(u,a) \in \mathbb{D}} \log \mathrm{H}(\mathrm{F}_1(u, a)) - \mathbf{1}_{(u,a) \in \mathbb{O}} \log \left[1 - \mathrm{H}(\mathrm{F}_1(u, a)\right] \tag{4}$$

The final loss function for adversarial regularization is shown in Equation (5):

$$\mathcal{L}_{adv} = \mathcal{L}_{target}(\mathrm{F}_2(\mathrm{F}_1(u, a)), y) - \lambda \mathcal{L}_{cls}(u, a), \tag{5}$$

where the $\mathcal{L}_{target}$ is the original loss function that trains the target model and the λ is hyper parameter weighting the regularization.

The goal is to minimize the $\mathcal{L}_{adv}$ with regarding to F_1, F_2 and the $\mathcal{L}_{cls}$ with regards to H.

3.6 Unsupervised Domain Adaptation

UDA (Unsupervised Domain Adaptation) is a technique to train a model that works well on the target domain with unlabeled data by only using labeled samples on the source domain. UDA method has been applied to the situation where the feature distribution and the data labeling are different between the source and target domains. In Pinterest Ads system, the source domain is the biased dataset with labels and the target domain is the unbiased dataset without labels. As a result, this data selection bias could be formulated as a UDA problem[22].

3.6.1 Naive UDA. The naive method is to directly train the model on the unbiased dataset so that there will be no inconsistency between training and serving. As the ground truth labels of the unbiased dataset are missing, the pseudo labels will be generated from a separate model that is trained on the biased dataset from the source domain where the ground truth label is available. Following the same annotation scheme as above, let R denotes the Ranking model that is used to generate the pseudo labels for the unbiased dataset from source domain. The optimization goal becomes the following:

$$\min_{\theta} \mathcal{L}_{naiveUDA}(\mathrm{F}_\theta) = \frac{1}{|\mathbb{D}|} \sum_{(u,a)\in\mathbb{D}} l(\mathrm{R}(u,a), \mathrm{F}_\theta(u,a)), \tag{6}$$

where $\mathbb{D}$ is the data in the source domain.

However, this method has many drawbacks. In reality, the unbiased data is only a sampling, and the volume is small due to infra cost. This might lead to performance degradation. Additionally, the high-quality candidates might not be sufficiently representative in this training data from the source domain. We will discuss the performance in the experiment section.

3.6.2 Modified UDA. In UDA, the quality of pseudo labels is critical to the performance of trained models. In the above naive UDA, there is no mechanism to guarantee the quality of the pseudo labels, especially when the pseudo label generating model remains not sufficiently accurate. Previously, Saito et al. [17] proposed to use an asymmetric tri-training method where two separate pseudo label generating models are used as the mechanism to ensure the pseudo label quality. However, the requirement to maintain a second pseudo label generating model with reasonable performance will be too costly for the real-world advertising system where tens or even hundreds of Retrieval models are needed and retrained on a daily basis. Additionally, it will be inhibitively costly when a pseudo label has to be derived from a set of models, and then a second set of several models will be required to be developed and maintained to leverage the tri-training method.

To address the pseudo label quality issue for real-world ads retrieval, we transform the original numeric pseudo label (prediction of Ranking model) to a binary classification label based on carefully chosen thresholds. Formally, let δ_l and δ_h denote the two thresholds with $\delta_l < \delta_h$. As shown in Equation 8, numeric pseudo labels lower than the first threshold are treated as negative, those higher than the second threshold are treated as positive. Data records with numeric pseudo labels falling between these two thresholds are removed from the training dataset.

The rationale behind this is to only keep the records that the Ranking model is confident about and discard the ones that are close to the hyperplane of the Ranking classifier. Now, the optimization goal of training the Retrieval model becomes the following:

$$\begin{aligned} &\min_{\theta} \mathcal{L}_{MUDA}(\mathrm{F}_\theta) \\ &= \frac{1}{|\mathbb{O}|\cdot|\mathbb{D}|} \sum_{(u,a)\in\mathbb{O}\cup\mathbb{D}\wedge(\mathrm{R}(u,a)\le\delta_l\vee \mathrm{R}(u,a)\ge\delta_h)} l(\Phi_{\delta_l}^{\delta_h}(\mathrm{R}(u,a)), \mathrm{F}_\theta(u,a)) \end{aligned} \tag{7}$$

where $\Phi_{\delta_l}^{\delta_h}(\cdot)$ is a pseudo classification label indicator, converting ranking model predictions to binary label according to given thresholds, as shown in the equation below:

$$\Phi_{\delta_l}^{\delta_h}(y) = \begin{cases} 1, & \text{if } y \ge \delta_h \\ -1, & \text{if } y \le \delta_l \end{cases}. \tag{8}$$

To select the thresholds, we adopt a data driven method. Particularly, we bucketize the Ranking model prediction and check the corresponding empirical click rate for each bucket. Thresholds are chosen when there is sudden change of empirical click rates.

4 EXPERIMENTS AND RESULTS

In this section, we will first describe the model training details and introduce the evaluation settings and metrics. We will then present and discuss the results from offline and online experiments to compare the performance of the proposed solutions.

4.1 Datasets

As described in Section 2.1, the two existing training data sources are the Auction candidates and Auction winners (both are biased datasets). In Section 2, we introduced an unbiased dataset randomly sampled from the post-Targeting dataset. This unbiased dataset is required to be scored and ranked by Retrieval models in the production system. Taking into consideration the infrastructure cost and the volume of the resulting dataset, we sample 100,000 queries and 6,000 advertisement candidates for each query to create the unbiased dataset every day.

4.2 Experimental setting

To examine the performance of de-biasing methods on the Pinterest's ads dataset, we implement the models and conduct systematic experiments to collect evaluation results on the real-world production system. The binary classification models are trained on Auction winners with real user actions as the true labels, which aims to provide supplemental evidence to indicate the reason why the current production model is not directly trained with real user actions. The following describes the details of the baseline models:

- **Binary Classification**: Since the regression model is trained on the pseudo labels generated by the Ads Ranking models, the performance of the classification model directly trained on the user actions is worth examining. To train this model, we use the Auction winner as the training dataset with labels defined in Section 3.1 and binary cross entropy (BCE) as the loss function.

- **In-batch Negative Classification**: We also train a classification model with in-batch negative sampling which uses other candidates from the same batch of data as negative samples for a given query. We use 1000 as the batch size and use the batch size as the number of hard negatives in the loss function. The model is trained with only the Auction winner dataset, and we only use the candidates with user clicks.
- **Knowledge Distillation**: In the current production model's training, we use the Ads Ranking model's output as the pseudo label and mean absolute logarithmic error (LogMAE) as the loss function. For the production model, the training dataset includes the Auction candidates and Auction winners. Besides this production model, we also train another one with only Auction winners. In evaluation, we refer to the first one as the Production model and the second one as the knowledge distillation model.

We summarize the implementation details of the debiasing model as the following:

- **Transfer Learning**: For the transfer learning model, we use both the biased and unbiased dataset. We also use Ranking model predictions as the pseudo labels and LogMAE as the loss function to train the Retrieval model.
- **Adversarial Learning**: For the adversarial learning model, we implement the data source discriminator as a one-layer MLP with sigmoid as the activation function. Both the biased and unbiased datasets are used to train the Retrieval model, and the Ads Ranking model is used to generate pseudo labels for the training datasets.
- **Naive Unsupervised Domain Adaptation (UDA)**: To train the naive UDA model, we only use the unbiased dataset with pseudo labels generated from the Ads Ranking model predictions and LogMAE as the loss function.
- **Modified Unsupervised Domain Adaptation (MUDA)**: Here we use the unbiased dataset with pseudo labels derived as discussed in Section 3.6.2 by transforming the Ranking model predictions into binary classes and BCE for the loss function.

For model training hyper-parameters, we use 6144 as the batch size and 0.0001 as the learning rate unless defined specifically. In the two-tower model, we use four fully connected layers, and the final layer's output dimension is 32. We use `sigmoid` as the activation function for the output layer and use `selu` [11] for the other layers.

4.3 Evaluation Metrics

For offline evaluation metrics, we use AUC-ROC score for both the classification and regression models. We evaluate the models on one day of the Auction winners dataset. For online A/B experiments, we compare these models to the production model and report the change of total impressions numbers (ΔIMP), click through rate (ΔCTR), and 30 seconds click through rate (ΔgCTR30)

For ads evaluation, besides the user-side metrics mentioned above, we also report metrics that relate to advertiser experience. These metrics are:

- **Impression to Conversion Rate Ratio (iCVR)** measures the effectiveness of an ad campaign in converting impressions into conversions.
- **Cost per Action (CPA)** measures the cost to the advertiser for each positive user action and is currently exclusively applied to the conversion ads.

Due to information confidentiality, we only report the lift of these metrics compared to the current production model.

4.4 Offline Evaluation

Models	AUC-ROC
Production Model	0.895
Binary Classification	0.895
In-batch Negative	0.701
Knowledge Distillation	0.896
Transfer Learning	0.890
Adversarial Learning	0.896
Naive UDA	0.841
MUDA	0.844

Table 1: AUC-ROC on evaluation dataset. The models such as knowledge distillation, adversarial learning, binary classification trained with Auction Winners dataset usually have better offline evaluation results.

For offline evaluation, we evaluate both the regression and classification models using AUC-ROC. For the evaluation dataset, we use the Auction winners which contain real user clicks. As shown in Table 1, compared to the production model, the models such as knowledge distillation, transfer learning, binary classification, and adversarial models have similar performance in terms of AUC-ROC score. The results are expected because the training datasets include the Auction winners for these models. For the in-batch negative model, it is trained with only the positive candidates in the Auction winners dataset, so it does not perform well in the offline evaluation because the negative candidates were not included in the training dataset. For Native UDA and MUDA, both models are trained with only the post-Targeting datasets, and the feature distribution discrepancy (Figure 2a) leads to the lower performance than other models. To summarize, the offline evaluation is as expected because we see models trained and evaluated on the same source of data have better performance than those trained with different sources of data. However, the offline evaluation could not necessarily reflect the true model performance in the production system especially when the serving data used in the online experiments are from a completely different distribution. Thus, in the following sections, we conduct systematic online A/B experiments to compare the performance of aforementioned models.

4.5 Online A/B Experiments

4.5.1 Overall evaluation. Table 2 shows the overall online evaluation results of all models. Among the metrics, we will focus on the change of gCTR30 as our models are optimized towards this objective. The binary classification model has decreased gCTR30, indicating a significant drop in the quality of user engagement with the recommended ads. It also shows the largest decrease in CTR and highest increase in impressions, which means that while more ads were delivered to users, fewer of them got clicked. In contrast, the in-batch negative and knowledge distillation models have positive changes in gCTR30. However, the decrease in impression could be the main reason for the gCTR30 increase because less ads were shown to the users.

Although all three models (binary classification, in-batch negative and knowledge distillation) suffer from selection bias in the training dataset, the latter two perform better. In the binary classification model's training, the negative candidates are always from the same query; whereas the in-batch negative classification model is trained with random sampled candidates of different queries within the batch. The difference between the source of negative candidates provides the model with more diverse and informative training data, which results in not overfitting to the specific query. For knowledge distillation, the training data labels are the Ranking model predictions, whose values contain richer information than raw binary click-or-not labels.

Models	ΔIMP	ΔCTR	ΔgCTR30
Binary Classification	0.95%	-5.51%	-12.66%
In-batch Negative	-2.25%	4.45%	4.68%
Knowledge Distillation	-3.26%	0.25%	5.97%
Transfer Learning	0.43%	-1.88%	-4.35%
Adversarial Learning	0.28%	-0.45%	-0.66%
Naive UDA	0.45%	-3.05%	-4.80%
MUDA	0.92%	0.47%	5.07%

Table 2: Online lifts of impression (IMP), click-through rate (CTR), and good long click (gCTR30) observed with various models on all types of ads. Both in-batch negative and knowledge distillation methods improve gCTR30 at the cost of impression drop, and MUDA is the only method to recommend more ads with higher quality, as observed by the increased gCTR30 without impression drop.

The transfer learning model had a small increase in impressions, but also had a negative change in gCTR30. With the warm start weights, the transfer learning model has similar results with the decrease in user engagements. In our case, the problem of the transfer learning model is the fine tuning on the unbiased dataset. Candidates in unbiased dataset are randomly sampled for each query, where high quality ones might be underrepresented.

The adversarial model has similar results with a decrease in gCTR30 and a slight increase in impressions. Compared to the transfer learning model, the adversarial model has better performance in user engagements. In adversarial model, the classifier serves as a regularizer to prevent the embedding tower from learning a domain specific embedding for a certain log source. Unlike the transfer learning model, the debiasing technique in the adversarial model does not rely on the quality of the unbiased training data. The training of the classifier is unsupervised because we use the log source (i.e. Auction winner, Auction candidates) as the ground truth label. When we are able to successfully train a classifier to classify the log source, the classifier could be used as an adversarial regularizer to help train an unbiased embedding model. However, compared to the production model, the decrease in gCTR30 may indicate that the restriction on the embedding learning makes the model drop the information that are critical in online evaluations.

The Naive UDA model has an average performance compared to the other baseline models. The Naive UDA model is trained on the unbiased dataset which contains the pseudo label generated from the Ranking model. The reason why the Naive UDA model performs badly is similar to the reason why the transfer learning model performed poorly. Since the unbiased dataset is collected by random sampling of post-Targeting ad candidates in addition to the existing queries, these sampled candidates are mostly negative samples, which does not help to train a good Retrieval model.

In contrast, the Modified UDA (MUDA) model has a much higher gCTR30 than the production model. When the number of impressions increases, the higher user engagement suggests that the MUDA model delivers more ads with higher quality to users. Compared to the Naive UDA model, the MUDA model transforms numerical pseudo labels generated by the Ranking model into binary classes determined by certain thresholds. The model also uses BCE loss. The lift in user engagement metrics suggests that such label transformation improves the quality of pseudo labels used in MUDA. By transforming the numerical pseudo labels to binary ones, we prevented the model from overly fitting into the Ranking model's prediction of every single candidate, but to rank those on which the Ranking model has high confidence.

4.5.2 Evaluation by ads objective type. In overall evaluation, the in-batch negative and MUDA are two methods that demonstrate promising metrics. In Table 3 we show the evaluation results of the two methods broken down by different ads objective types, i.e., awareness, traffic and web conversion ads. Awareness ads aim to increase the visibility of a brand or product. Analyzing the performance of awareness ads helps understand the effectiveness of a brand's marketing strategy. As shown in Table 3, the in-batch negative model has significant increase on gCTR30 compared to other models for awareness ads. However such a boost might be due to the huge decrease in impressions. In-batch negative model is trained only on candidates with user long clicks. The awareness ads essentially have a lower chance of being clicked than other types, since its main goal is to increase the visibility of a brand. As a result, the in-batch negative model could bias towards other ads types, leading to the huge impression drop of awareness ads.

Traffic ads are designed to drive traffic to a specific website or landing page. They are typically used to increase brand awareness, generate leads, or drive sales. By analyzing metrics such as CTR and gCTR30, businesses can determine whether their ads are resonating with their target audience and whether they are successfully achieving their advertising goals. As can be seen, the in-batch negative model is the only ones that yield an increase in both ads impression and gCTR30. Traffic ads aim to attract users to click and could occupy a big portion of records with positive user actions. Therefore the in-batch negative model's training dataset, which only includes candidates with positive user actions, may have a higher proportion of candidates that are well-suited for driving traffic. As a result, the model could better identify candidates that are likely to drive traffic, resulting in an improvement in the gCTR30 metric for traffic ads.

Web-conversion ads aim to drive users to take a specific action on a website, such as making a purchase. These ads can provide insight for measuring the success of an online advertising campaign. As shown in the Table 3, the MUDA model favors the web-conversion ads objective type, as it has the highest improvement in CTR and gCTR30 among all models for this objective type. The in-batch

Models	Awareness			Traffic			Web Conversion		
	ΔIMP	ΔCTR	ΔgCTR30	ΔIMP	ΔCTR	ΔgCTR30	ΔIMP	ΔCTR	ΔgCTR30
In-batch Negative	-8.70%	2.41%	13.74%	1.03%	1.16%	2.56%	1.31%	1.69%	0.39%
MUDA	0.32%	2.71%	1.97%	0.43%	-4.28%	-3.07%	3.15%	5.19%	8.88%

Table 3: Online lifts of impression (IMP), click-through rate (CTR), and good long click (gCTR30) observed with two promising models on each type (awareness, traffic, web-conversion) of ads. In-batch negative classification model works better on the traffic ads, and MUDA model helps web-conversion ads the most.

negative model also performs well for web-conversion ads, with improvements in both CTR and gCTR30. The MUDA may favor web-conversion ads because pseudo labels generated by the Ads Ranking model may favor web-conversion ads as they are designed to attract users to stay on target websites longer for potential conversion behaviors. Additionally, the threshold selection strategy used in the MUDA model may be more effective at identifying high-quality candidates for web-conversion ads, which could also contribute to its better performance for this type of ad.

Models	ΔiCVR	ΔCPA
In-batch Negative	-2.55%	1.11%
MUDA	1.89%	-4.40%

Table 4: Online metrics performance of in-batch negative classification and MUDA models on web-conversion ads. In-batch negative classification model leads to lower conversion probability on each ads impression (iCVR) and thus has a higher CPA cost to advertisers. In contrast, MUDA model recommended ad candidates with higher conversion rate and therefore a lower CPA cost.

4.5.3 Conversion ads. In Table 4, we show the performance of in-batch negative and MUDA models with regard to conversion related metrics as these two show good performance on web-conversion ads. In general, the in-batch negative model has a decreased iCVR and increased CPA. This is not favorable for the advertiser as it increases their costs as measured by CPA. On the other hand, the MUDA model shows an opposite result with increased iCVR and decreased CPA, reducing the ads campaign cost to advertisers. These metrics indicate that while both increase the long clicks, MUDA model performs much better by generating more conversions out of these increased long clicks.

One reason why the MUDA model performs better is that the model could improve the performance of identifying the high-quality candidates that are more likely to lead to conversions, and thus decrease the cost per action for advertisers. Additionally, the fact that the MUDA model is trained on only unbiased data with pseudo labels generated from the Ads Ranking model could have an impact. The pseudo labels may capture more relevant information about the users' behaviors and preferences, leading to better performance in terms of CPA.

4.6 Variants of MUDA

In the MUDA method, we believe different threshold selection mechanisms could impact the quality of binary pseudo labels. As a result, we further investigate the impact of different thresholding mechanisms on the performance of trained Retrieval models. In the unbiased dataset, we first bucketize the candidates according to their numerical pseudo labels (gCTR30) predicted by the Ranking model, where we compute the percentile of the labels and use the adjacent percentile to create buckets. Then for each bucket, we adapt the following two strategies to calculate empirical gCTR30:

- compute the gCTR30 for candidates with the real user actions,
- divide the number of true good clicks by the number of candidates in the bucket.

We select the threshold by determining the elbow point of the graph. For example, when there is a sudden drop of true good clicks or good clicks rate between two adjacent bins, we use one of the bins as the negative threshold. This means, in the label transformation, we treat candidates with pseudo labels smaller than the threshold as the negative samples. For positive labels, we check if there is a sudden increase of true good clicks or good clicks rate between the bins. To study different threshold selection strategy, we propose three variants of the MUDA models:

- v1: We train the MUDA model on both the biased and unbiased datasets with the first threshold selection strategy.
- v2: We train the MUDA model on only the unbiased datasets with the first threshold selection strategy.
- v3: We train the MUDA model on only unbiased datasets with the second threshold selection strategy.

Models	ΔIMP	ΔCTR	ΔgCTR30
MUDA v1	-0.07%	11.26%	30.78%
MUDA v2	0.56%	3.52%	13.04%
MUDA v3	0.92%	0.47%	5.07%

Table 5: Online lifts of impression (IMP), click-through rate (CTR), good long click (gCTR30) observed with various MUDA variants on all types of ads. MUDA v1 achieves the highest gain on ads engagement (both CTR and gCTR30), and MUDA v3 achieves the most balanced gain across different metrics with good gCTR30 and impression lift.

Table 5 shows the overall performance of three variants of the UDA models, as measured by several evaluation metrics: impression, and click-through rate (CTR). At first glance, the v1 model may seem to work best, hugely increasing user engagement while keeping the impression neutral. However, if broken down by ad types, this mode actually leads to a large impression shift from awareness (-2.13%) and traffic ads (-5.12%) toward web conversion ads (+12.98%), as shown in Table 6. This observation may indicate that training UDA models on biased data would make the model favor web-conversion ads more than others. Comparing v2 and v3 models, the latter one shows a better balanced impression gains across all ad objective types. It could be due to the second strategy of calculating the approximate gCTR30 for the unbiased dataset. This strategy may better represent the true performance of the candidates and result in a more accurate threshold selection, leading to improved performance in the MUDA model.

Models	Awareness			Traffic			Web Conversion		
	ΔIMP	ΔCTR	ΔgCTR30	ΔIMP	ΔCTR	ΔgCTR30	ΔIMP	ΔCTR	ΔgCTR30
MUDA v1	-2.13%	2.77%	7.97%	-5.22%	0.47%	17.34%	12.98%	21.52%	29.63%
MUDA v2	0.10%	1.72%	5.97%	-1.53%	-2.69%	1.18%	5.16%	11.14%	17.83%
MUDA v3	0.32%	2.71%	1.97%	0.43%	-4.28%	-3.07%	3.15%	5.19%	8.88%

Table 6: Online lifts of impression (IMP), click-through rate (CTR), and good long click (gCTR30) observed with MUDA variants on each type (awareness, traffic, web-conversion) of ads, where MUDA v3 shows best balanced impression gains among them.

To better understand the performance of these MUDA variants, we also measure their online performance on two other useful engagement metrics: hide rate (HDR) and re-pin rate (RPR). Note that re-pin is a user action indicating if the user saves an ad to a Pinterest board. In Table 7, we show the change of the two metrics compared to the production model. Although the RPR is increased, both MUDA v1 and v2 models recommend more ads that will be hidden by users, suggesting some of the recommended ads from these models do not provide a good user experience. In contrast, MUDA v3 model generally has the most balanced improvement across all metrics, which shows positive lift in the user engagement and reduction in the unwanted user experience (HDR).

Models	ΔHDR	ΔRPR
MUDA v1	4.80%	13.88%
MUDA v2	6.35%	4.43%
MUDA v3	-2.81%	1.43%

Table 7: Online lifts of ads hide rate (HDR), re-pin rate (RPR) observed with MUDA variants on all types of ads. MUDA v3 achieves the most balanced performance with fewer ads being hidden and more ads being repined by the users.

5 RELATED WORKS

5.1 Pre-Ranking System

The pre-ranking system is one of the stages in the multi-stage cascade ranking architecture, and the system aims to rank millions of ads given a query. Considering the cost of computation power and limit latency restriction, a lightweight ranking system is usually applied. As summarized by Zhe [21], there are four generations in the development history of the pre-ranking system. The first generation applies a statistical model which ranks the ads by averaging the recent CTR. The second-generation applies the Logistic Regression (LR) model, and the Cascade Ranking model [12] and Ad Click Prediction model [13] are the examples. The third generation is the vector-production based deep learning models such as EENMF [24] and the two-tower model for YouTube recommnedataions [5]. The fourth generation aims to improve the model expression ability and the update frequency, and COLD [21] is proposed to accomplish it. However, the models mentioned above may suffer from the selection bias issue in the training and inference dataset.

5.2 Selection Bias

Research on selection bias in recommendation systems is increasing, exploring methods to reduce bias and enhance system performance. One of the approaches is through re-sampling techniques. This includes methods such as undersampling [9, 15] and SMOTE (Synthetic Minority Over-sampling Technique) [1, 2] which aims to balance out the distribution of data across different classes. Another popular approach is the use of cost-sensitive learning methods, which assign different costs to different types of errors in order to balance the trade-off between different types of bias. For example, the method of adversarial learning [4, 27] aims to minimize bias by adding an adversarial term to the loss function that encourages the model to produce fair predictions. Another area of research focuses on the use of debiasing techniques in the representation learning process, such as Fair Representation Learning [26] which learns representations that are invariant to certain sensitive attributes. There are also other recent studies that address selection bias by using counterfactual data augmentation (CFDA) [20], which creates new, hypothetical data points to increase the diversity of the training set. This can be done by generating synthetic data points that are similar to the original data points, but with different sensitive attributes. In addition, meta-learning [3, 19] have been applied to debiasing recommendation systems. For multi-stage cascade systems, Qin et al. [16] proposed the RankFlow to solve the selection bias in the joint-training system, but it could be expensive to deploy in the production system. Our work aims to solve the selection bias issue for independent-training models in the cascade system.

6 CONCLUSION

In conclusion, this paper has analyzed the impact of selection bias in Pinterest's online advertising system. We propose and evaluate several debiasing methods to mitigate the negative impacts of selection bias on recommender's performance. The results of our experiments show that our proposed methods, specifically the MUDA model, can effectively improve the performance of advertising systems by handling the selection bias. Additionally, our online experiment shows that this model also improves the cost efficiency of the ad campaigns. These findings demonstrate the importance of addressing selection bias in recommendation systems and provide valuable insights for practitioners in this field.

Future work can be done to investigate other debiasing methods and to further evaluate the proposed methods on different types of recommendation tasks and systems. Furthermore, it would be also beneficial to investigate the interaction between selection bias and other types of biases that may exist in recommendation systems, such as demographic bias and representation bias. Understanding how these biases interact and how they can be mitigated would be an important step towards building more performant and inclusive recommendation systems.

ACKNOWLEDGMENTS

The authors would like to thank Xiaofang Chen, Dmitry Lubensky, and Katharine Wang who have provided insightful reviews and feedback to this paper.

REFERENCES

[1] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. 2012. DBSMOTE: density-based synthetic minority over-sampling technique. *Applied Intelligence* 36 (2012), 664–684.

[2] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority over-Sampling Technique. *J. Artif. Int. Res.* 16, 1 (jun 2002), 321–357.

[3] Jiawei Chen, Hande Dong, Yang Qiu, Xiangnan He, Xin Xin, Liang Chen, Guli Lin, and Keping Yang. 2021. AutoDebias: Learning to Debias for Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 21–30.

[4] Daniel Cohen, Bhaskar Mitra, Katja Hofmann, and W. Bruce Croft. 2018. Cross Domain Regularization for Neural Ranking Models Using Adversarial Learning. In *The 41st International ACM SIGIR Conference on Research Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 1025–1028.

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems* (Boston, Massachusetts, USA) *(RecSys '16)*. Association for Computing Machinery, New York, NY, USA, 191–198.

[6] Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning Dense Representations for Entity Retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*. Association for Computational Linguistics, Hong Kong, China, 528–537.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (Eds.), Vol. 27. Curran Associates, Inc.

[8] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision* 129 (2021), 1789–1819.

[9] Gert Jacobusse and Cor Veenman. 2016. On Selection Bias with Imbalanced Classes. In *Discovery Science*, Toon Calders, Michelangelo Ceci, and Donato Malerba (Eds.). Springer International Publishing, Cham, 325–340.

[10] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online.

[11] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-Normalizing Neural Networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 972–981.

[12] Shichen Liu, Fei Xiao, Wenwu Ou, and Luo Si. 2017. Cascade Ranking for Operational E-Commerce Search. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada) *(KDD '17)*. Association for Computing Machinery, New York, NY, USA, 1557–1565.

[13] H. Brendan McMahan, Gary Holt, D. Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, Sharat Chikkerur, Dan Liu, Martin Wattenberg, Arnar Mar Hrafnkelsson, Tom Boulos, and Jeremy Kubica. 2013. Ad Click Prediction: A View from the Trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Chicago, Illinois, USA) *(KDD '13)*. Association for Computing Machinery, New York, NY, USA, 1222–1230.

[14] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359.

[15] Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson, and Gianluca Bontempi. 2015. Calibrating Probability with Undersampling for Unbalanced Classification. In *2015 IEEE Symposium Series on Computational Intelligence* (Cape Town, South Africa). IEEE, New York, NY, USA, 159–166.

[16] Jiarui Qin, Jiachen Zhu, Bo Chen, Zhirong Liu, Weiwen Liu, Ruiming Tang, Rui Zhang, Yong Yu, and Weinan Zhang. 2022. RankFlow: Joint Optimization of Multi-Stage Cascade Ranking Systems as Flows *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 814–824.

[17] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. 2017. Asymmetric Tri-Training for Unsupervised Domain Adaptation. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (Sydney, NSW, Australia) *(ICML'17)*. JMLR.org, 2988–2997.

[18] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) *(SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 115–124.

[19] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2021. Combating Selection Biases in Recommender Systems with a Few Unbiased Ratings. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (Virtual Event, Israel) *(WSDM '21)*. Association for Computing Machinery, New York, NY, USA, 427–435.

[20] Zhenlei Wang, Jingsen Zhang, Hongteng Xu, Xu Chen, Yongfeng Zhang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Counterfactual Data-Augmented Sequential Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 347–356.

[21] Zhe Wang, Liqin Zhao, Biye Jiang, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2020. Cold: Towards the next generation of pre-ranking system. *arXiv preprint arXiv:2007.16122* (2020).

[22] Garrett Wilson and Diane J. Cook. 2020. A Survey of Unsupervised Deep Domain Adaptation. *ACM Trans. Intell. Syst. Technol.* 11, 5, Article 51 (jul 2020), 46 pages.

[23] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable Zero-shot Entity Linking with Dense Entity Retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 6397–6407.

[24] Wenjin Wu, Guojun Liu, Hui Ye, Chenshuang Zhang, Tianshu Wu, Daorui Xiao, Wei Lin, and Xiaoyu Zhu. 2018. EENMF: An End-to-End Neural Matching Framework for E-Commerce Sponsored Search. *CoRR* abs/1812.01190 (2018). arXiv:1812.01190 http://arxiv.org/abs/1812.01190

[25] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-End Neural Ad-Hoc Ranking with Kernel Pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Shinjuku, Tokyo, Japan) *(SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 55–64.

[26] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning Fair Representations. In *Proceedings of the 30th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 28)*, Sanjoy Dasgupta and David McAllester (Eds.). PMLR, Atlanta, Georgia, USA, 325–333.

[27] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. 2018. Mitigating Unwanted Biases with Adversarial Learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society* (New Orleans, LA, USA) *(AIES '18)*. Association for Computing Machinery, New York, NY, USA, 335–340.