Visual Encoding and Debiasing for CTR Prediction

Guipeng Xv†
xvguipeng.xgp@alibaba-inc.com
Xiamen University
Xiamen, China

Si Chen†
yasui.cs@alibaba-inc.com
Alibaba Group
Hangzhou, China

Chen Lin†
chenlin@xmu.edu.cn
Xiamen University
Xiamen, China

Wanxian Guan
Xingyuan Bu
wanxian.gwx@alibaba-inc.com
buxingyuan.bxy@alibaba-inc.com
Alibaba Group
Hangzhou, China

Xubin Li†
Hongbo Deng
lxb204722@alibaba-inc.com
dhb167148@alibaba-inc.com
Alibaba Group
Hangzhou, China

Jian Xu
Bo Zheng
xiyu.xj@alibaba-inc.com
bozheng@alibaba-inc.com
Alibaba Group
Hangzhou, China

ABSTRACT
Extracting expressive visual features is crucial for accurate Click-Through-Rate (CTR) prediction in visual search advertising systems. Current commercial systems use off-the-shelf visual encoders to facilitate fast online service. However, the extracted visual features are coarse-grained and/or biased. In this paper, we present a visual encoding framework for CTR prediction to overcome these problems. The framework is based on contrastive learning which pulls positive pairs closer and pushes negative pairs apart in the visual feature space. To obtain fine-grained visual features, we present contrastive learning supervised by click-through data to fine-tune the visual encoder. To reduce sample selection bias, firstly we train the visual encoder offline by leveraging both unbiased self-supervision and click supervision signals. Secondly, we incorporate a debiasing network in the online CTR predictor to adjust the visual features by contrasting high impression items with selected, low impression items. We deploy the framework in a mobile E-commerce app. Offline experiments on billion-scale datasets and online experiments demonstrate that the proposed framework can make accurate and unbiased predictions.

CCS CONCEPTS
• Information systems → Recommender systems.

KEYWORDS
visual-aware CTR, bias, contrastive learning

†This work was completed when Guipeng Xv was an intern at Alibaba Group. Guipeng Xv and Si Chen contribute equally.
†Corresponding authors. Supported by the Natural Science Foundation of China (No. 61972328), Alibaba Group through Alibaba Innovative Research program.

ACM Reference Format:

1 INTRODUCTION
Visual search advertising systems have been embraced by E-commerce platforms such as Amazon and Taobao, and have become a billion dollar business. In visual search advertising systems, items are displayed with images and the images of items are accepted as queries. E-commerce platforms display ads in search results whenever a user searches for an item, and gain revenue whenever the user clicks an ad. The decision of ad placements is based on the product of predicted Click-Through-Rate (CTR) and the bid price. Therefore, to improve the performance of CTR prediction and consequently increase revenue, extracting expressive visual features from item images to match user intent, is of vital importance.

Current commercial visual search systems consist of two components [6], the Visual Encoder with various Deep Neural Network (DNN) based modules which is trained off-the-shelf to extract visual features, and the CTR predictor fuses visual features with non-visual features in different DNN modules to make predictions. However, training the Visual Encoder is challenging. On one hand, training the Visual Encoder with non-click-through signals leads to sub-optimal, coarse-grained feature representations. For example, if the training task uses category labels, the feature representations can distinguish the category of clothes, but can not discover subtle style differences, which have a significant impact on user behaviors [11]. On the other hand, if the Visual Encoder uses interaction signals such as clicks or purchases to be labels, it will face the problem of sample selection bias. For example, ads with low impressions (i.e., displayed less often in the system) will receive fewer positive labels and therefore are under-represented in the learning process.

This paper describes our solution in Alibaba. To facilitate real-time CTR prediction at scale, our solution also consists of two components, i.e., an off-the-shelf Visual Encoder and an online CTR Predictor. Our work is based on contrastive learning, i.e.,
learned feature representation of positive sample is pulled closer to the anchor image, while representation of a negative sample is pushed apart. Visual Encoder is firstly pre-trained with self-supervised contrastive loss, with random negative samples. It is then fine-tuned with supervised contrastive loss, where the selection of positive and negative samples is dependent on user clicks. Thus, we obtain finer-grained, more expressive visual features for CTR prediction, by leveraging user behavior information. In CTR predictor, we feed the extracted visual features through a debiasing network before fusing with non-visual features. The debiasing network regularizes the CTR prediction loss with a contrastive loss, which encourages similar images from low impression items and high impression items to assemble. In this manner, we reduce the selection sample bias which has been introduced in the previous finetuning stage, while preserving the CTR prediction accuracy.

In summary, our contributions are three-fold. (1) We study the problem of sample selection bias in visual features in advertising systems, which has not been explored in literature. Solutions to this problem shed light on the well-known “accuracy-diversity” dilemma in recommender systems. (2) We present a novel approach, which operates at scale, to extract effective visual features for accurate and unbiased CTR prediction. (3) Offline experiments on ten-billion scale real production datasets demonstrate that the proposed pretraining-finetuning-debiasing framework has increased the accuracy of CTR prediction, especially for long-tail ads. Online A/B testing shows that, deploying the solution in a mobile advertising app improves performance metrics such as the click-through rate and revenue per mille.

2 RELATED WORK

Previous work [1] extracts visual features of raw image and predicts CTR in one step. To speed up training online advertising system which encounters massive responses everyday, adopting off-the-shelf visual feature extraction modules has recently gained popularity [3–7, 15, 16, 18, 19]. Most of them use CNNs as a visual encoder and pre-train the CNNs on image classification task. To learn visual compatibility across categories for fashion recommendation, the visual encoder in [16] is pre-trained with weakly-labeled clothing collocation data. To learn category-specific inter-channel dependency, category-specific CNNs are adopted [6]. While images can be similar from multiple perspectives, training the visual encoder with image category labels is sub-optimal for CTR prediction.

The click-through data is inherently biased, because ads must be exposed before being clicked. There are fruitful literature to alleviate Sample Selection Bias in the search and recommendation community, e.g., adapt models from causal inference [9, 12, 17], or leverage purchase-related actions with multi-tasking [8, 13, 14] to transfer features in entire action space. However, to the best of our knowledge, SSB in visual feature extraction has not been explored.

3 METHODOLOGY

As shown in Figure 1, the Visual Encoder (Section 3.1) extracts visual features for any image. It consists of two stages: S1 and S2, both of which are based on contrastive learning. The Visual Encoder is trained offline separately, while the online serving system is the CTR predictor (Section 3.2). A debiasing network is plugged in CTR predictor to process visual features for ad items.

3.1 Visual Encoder

S1: Pretraining Visual Encoder. The standard self-supervised contrastive learning scheme is adopted. In a mini-batch of images \( N^{S1} \), for each anchor image \( i \in N^{S1} \), we augment it with a series of transformation, including random cropping, random color jitter, random grayscale, and random flipping. Thus, the positive sample \( i' \) is obtained by \( i' = t(i) \), where \( t(·) \) represents the transformation. The rest of the images within the mini-batch are considered as negative samples. Then, the anchor image, the positive sample, and the negative sample go through a visual encoder to obtain their visual features, by minimizing the contrastive loss:

\[
\mathbb{L}_{S1} = - \sum_{i \in N^{S1}} \log \frac{\exp (g(v_{S1}^{i}, v_{S1}^{i'}))}{\sum_{j \in N^{S1} \cup \{t(i)\}} \exp (g(v_{S1}^{i}, v_{S1}^{j}))}.
\]  

where \( v_{S1}^{i} \in \mathbb{R}^{D} \) is the output visual feature vector of image \( i \), \( D \) is the embedding size, \( g(v_{S1}^{i}, v_{S1}^{j}) = \cosine(v_{S1}^{i}, v_{S1}^{j}) \) is the cosine similarity between two visual feature vectors.

S2: Finetuning Visual Encoder. After pre-training the visual encoder, we fine-tune its parameters. The difference between S2 and S1 lies in the construction of positive and negative samples. Clicks are one of the most invaluable sources to estimate visual relevance of an item given the query image. Thus we use the image of a clicked item as positive sample for an image query. However, it is well known that lack of clicks does not indicate irrelevance. To improve the quality of negative samples, we use the category information to build a negative sample pool. In E-commerce, each image is clearly labeled by its category (e.g., in the clothing section, an image could be labeled as “dress” or “pants”, etc.).

For each query image \( q \), we sample a clicked image \( i \) as \( q \)’s positive image. The category label of \( i \) is denoted by \( c_{i} \), \( N^{S2}_{c} \) is a collection of images of category label \( c_{j} \) which can be seen as a negative sample pool.

\[
\mathbb{L}_{S2} = - \sum_{q \in Q} \log \frac{\exp (g(v_{S2}^{q}, v_{S2}^{i}))}{\sum_{j \in N^{S2}_{c_{i}} \cup \{t(i)\}} \exp (g(v_{S2}^{q}, v_{S2}^{j}))},
\]

where \( v_{S2}^{i} \in \mathbb{R}^{D} \) is the output visual feature of image \( i \) in stage S2. It is of the same size as \( v_{S1}^{i} \), \( j \in N^{S2}_{c_{j}} \) restricts negative samples belong to the same category as anchor, thus the negative samples are more informative and the contrastive task will be more difficult.

3.2 CTR Predictor

The CTR predictor aims to rank items in a pool of candidate ads to be displayed by predicting the possibility of each item \( p \) being clicked by user \( u \) given query \( q \) under context \( x \). The inputs include the image of the item (to simplify notations, we also use \( p \) to denote the item image), other item metadata such as item ID, shop ID, brand, category, price, and so on, user features such as user ID, user demographic features, preferred categories, and so on, context features such as device and position. Each query is an image, also denoted as \( q \).

Debiasing Network. It is possible that S2 introduces sample selection bias to the visual features. For example, long-tail items
with small impressions (i.e., number of times the ad has been displayed in total) are less likely to be clicked, so they consequently make little contribution to S2. To eliminate such bias, in the CTR predictor, each item image goes through a debiasing network, which is based on contrastive learning. Our intuition is to pull image-pairs that are visually similar but significantly different in the number of impressions closer. In order to mine such sample pairs, we use unbiased S1 representation to depict the similarity of images and construct debiasing samples.

To construct positive sample for each anchor item image \( p \), item image \( p \) to be displayed, we go through two steps. Firstly we retrieve a set \( \mathcal{P} = \{ p' \} \) of \( K \) most similar images of non-displayed items with the same category label. We use the visual features extracted by stage S1 to compute the similarity, i.e., \( \text{sim}(p, p') = \text{cosine}(v^S_1 p, v^S_1 p') \), so that the similarity will not be biased against long-tail items. Secondly, the positive sample is selected based on the similarity, i.e., \( \text{Pr}(p') = \text{sim}(p, p') / \sum_{p' \in \mathcal{P}} \text{sim}(p, p') \), where \( \text{Pr}(p') \) is the probability of \( p' \) being selected as a positive sample. The negative sample of each anchor is randomly selected.

Then, the debiasing network \( D \) feeds a Multilayer Perceptron (MLP) with the visual features obtained by S2, i.e., \( v^S_2 \). The image \( p \) is then contrasted positively with \( p' \) and negatively with other images in the mini-batch \( N_{CTR} \).

\[
\mathbb{L}_D = - \sum_{p \in N_{CTR}} \log \frac{\exp(g(v^D_p, v^D_p))}{\sum_{o \in N_{CTR} \cup \{p'\}} \exp(g(v^D_p, v^D_o))}. \tag{3}
\]

where \( v^D_p \in \mathbb{R}^D \) is the output visual features of image \( p \) in by the MLP, i.e., \( v^D_p = \text{MLP}(v^S_2 p) \). Minimizing \( \mathbb{L}_D \) pushes item images with high impressions to be closer to similar item images with low impressions, and thus mitigates the bias of \( v^S_2 \).

Next, \( v^S_2 \) and \( v^D_p \) go through a gating layer to generate effective and unbiased visual features for item \( p \). \( \alpha = \sigma(W^T[v^S_2 p, v^D_p]) \), where \( \sigma(\cdot) \) is the sigmoid function, \( W \) is a learnable weight matrix, \( \ldots \) is the concatenation of several vectors/scalers. Finally, the visual feature of item \( p \) is obtained: \( v_p = \alpha v^S_2 p + (1 - \alpha)v^D_p \).

Since in this paper we focus on visual encoding, the rest of the CTR predictor can be very flexible as the pretraining-finetuning-debiasing network can plug into various frameworks. In the experiments, the visual feature of the query image \( q \) is generated by the fine-tuned Visual Encoder, i.e., \( v_q = v^S_2 q \). The CTR predictor takes input of non-visual features, transforms them into embedding vectors through lookup tables, and feeds the concatenation of all embedding vectors to a tower MLP to make the prediction. Overall, the CTR predictor is optimized by minimizing the loss function:

\[
\mathbb{L}_{CTR} = \mathbb{L}_{pred} + \mathbb{L}_D. \tag{4}
\]

where \( \mathbb{L}_{pred} = - \sum_{y \in N_{CTR}} \log \hat{y} + (1 - y) \log (1 - \hat{y}) \) is the cross-entropy loss to evaluate CTR prediction accuracy, \( y \in \{0, 1\} \) is the actual click, and \( \hat{y} \) is the predicted click probability. By incorporating \( \mathbb{L}_D \), the debiasing network is trained jointly with CTR predictor to achieve accurate and unbiased predictions.

4 EXPERIMENTS

In this section we analyze our experimental results in offline and online evaluations. The backbone of the visual encoder in S1 and S2 is ResNet50. We set the dimension size of visual features as \( D = 512 \). In the debiasing network, we select \( K = 15 \) similar images, the MLP has three hidden layers with 128, 16, 128 units, and the activation functions are ReLU, tanh, ReLU. The output layer has 512 units to output the visual feature vector. The tower MLP in CTR predictor has three hidden layers with 512, 256, 128 units, and the activation functions are ReLU, the output layer applies the sigmoid function to bound the prediction to \( (0, 1) \). We use the Adagrad optimizer with learning rate 0.05.

4.1 Offline Visual Search Evaluation

Dataset. To evaluate whether the extracted visual features are effective in identifying items, we perform a visual search task on an internal dataset. The dataset contains tens of thousands of item images sampled from multiple categories in our production system (e.g., clothing section, digital device section, furniture section, and so on.). The relevant image-pairs are manually annotated. The relevance judgement is binary (i.e., relevant or irrelevant), and it is based on a set of factors including style and design.

Baselines. We compare the following visual encoders, including deep neural network classifiers and basic contrastive learning methods. (1) ResNet-C: a ResNet50 is trained on the item images to predict the correct category labels. (2) S1: ResNet50 trained with self-supervised contrastive loss as in stage S1; (3) S2: the ResNet50 trained with click-through supervisions as described in stage S2; (4) S1+S2: first pre-train the ResNet50 as in stage S1 and then fine-tune it as in stage S2.
Evaluation Metric. After training each visual encoder $M$, visual feature vectors are extracted, we rank the images based on cosine similarity of visual feature vectors to the query image $q$. The result is denoted as $M_q$. We adopt three evaluation metrics. (1) The primary metric is HitRatio, i.e., $HR = \sum_q |Q_q \cap M^K_q|/|\sum_q |Q^K_q|$, where $n_q$ is the number of relevant images in the groundtruth $|Q_q| = n_q$. Higher $HR$ suggests higher search accuracy. (2) To reveal the diversity of results, we compute the ratio of images with low impressions in the returned images, i.e., $LR@K = \sum_q |L_q \cap M^K_q|/\sum_q |M^K_q|$, where $L_q$ is the set of images which receive less than five impressions during the last 30 days, and $M^K_q$ is the top-K results. Higher $LR@K$ suggests that the visual encoder is more fair to items with low impressions. (3) We also compute a supplementary metric, the ratio of images with the same categories in the results, i.e., $CR@K = \sum_q |C_q \cap M^K_q|/\sum_q |M^K_q|$, where $C_q$ is the set of images which are under the same category label of query image $q$. $CR$ provides information about the granularity of the visual features.

Analysis. As shown in Table 1, the proposed off-the-shelf visual encoding framework (i.e., $S1+S2$) achieves both highest accuracy (i.e., HR) and highest coverage of low impression items (i.e., LR).

4.2 Offline CTR Evaluation

Dataset. The offline CTR evaluation is conducted on a billion-scale dataset, which is collected from our production system, the training data spans for a period of 15 days sampled from July, 2021, with 0.4 billion different item images and 1 billion samples. The testing data is collected from the next day of the last training date.

Evaluation protocols. The competitors are CTR predictors using different visual encoding modules, including (1) ResNet-C, (2) VGG trained with category labels [10], (3) VIT trained with category labels [2]. We also conduct ablation study with different combinations of $S1$, $S2$, and D (debiasing network). The evaluation metric is AUC. We report the average AUC results and the AUC results on items with the lowest impressions (bottom 10%) and the highest impression (top 10%).

Analysis. As shown in Table 2, compared with the best competitor ResNet, the proposed framework $S1+S2+D$ increases AUC on testing set by 5%. Given the scale of our data, this is a significant improvement. Comparing among the different combinations of pretraining, finetuning and debiasing, we can see that neither $S1$ nor $S2$ alone can achieve optimal results. Furthermore, although $S1+S2$
REFERENCES