Category-Specific CNN for Visual-aware CTR Prediction at JD.com

Hu Liu
Business Growth BU, JD
liuhu1@jd.com

Jing Lu
Business Growth BU, JD
lvjing12@jd.com

Hao Yang
Business Growth BU, JD
yanghao17@mails.tsinghua.edu.cn

Xiwei Zhao
Business Growth BU, JD
zhaoxiwei@jd.com

Sulong Xu
Business Growth BU, JD
xusulong@jd.com

Hao Peng
Business Growth BU, JD
penghao5@jd.com

Zehua Zhang
Business Growth BU, JD
zhangzehua@jd.com

Wenjie Niu
Business Growth BU, JD
niuwenjie@jd.com

Xiaokun Zhu
Business Growth BU, JD
zhuxiaokun@jd.com

Yongjun Bao
Business Growth BU, JD
baoyongjun@jd.com

Weipeng Yan
Business Growth BU, JD
paul.yan@jd.com

ABSTRACT

As one of the largest B2C e-commerce platforms in China, JD.com also powers a leading advertising system, serving millions of advertisers with fingertip connection to hundreds of millions of customers. In our system, as well as most e-commerce scenarios, ads are displayed with images. This makes visual-aware Click Through Rate (CTR) prediction of crucial importance to both business effectiveness and user experience. Existing algorithms usually extract visual features using *off-the-shelf* Convolutional Neural Networks (CNNs) and *late* fuse the visual and non-visual features for the finally predicted CTR. Despite being extensively studied, this field still face two key challenges. First, although encouraging progress has been made in offline studies, applying CNNs in real systems remains non-trivial, due to the strict requirements for efficient end-to-end training and low-latency online serving. Second, the off-the-shelf CNNs and late fusion architectures are suboptimal. Specifically, off-the-shelf CNNs were designed for classification thus never take categories as input features. While in e-commerce, categories are precisely labeled and contain abundant visual priors that will help the visual modeling. Unaware of the ad category, these CNNs may extract some unnecessary category-unrelated features, wasting CNN's limited expression ability. To overcome the two challenges, we propose **Category-specific CNN** (CSCNN) specially for CTR prediction. CSCNN *early* incorporates the category knowledge with a light-weighted attention-module on each convolutional layer. This enables CSCNN to extract expressive category-specific visual patterns that benefit the CTR prediction.

Offline experiments on benchmark and a 10 billion scale real production dataset from JD, together with an Online A/B test show that CSCNN outperforms all compared state-of-the-art algorithms. We also build a highly efficient infrastructure to accomplish end-to-end training with CNN on the 10 billion scale real production dataset within 24 hours, and meet the low latency requirements of online system (20ms on CPU). CSCNN is now deployed in the search advertising system of JD, serving the main traffic of hundreds of millions of active users.

1 INTRODUCTION

As one of the largest B2C e-commerce platforms in China, JD.com also powers a leading advertising system, serving millions of advertisers with fingertip connection to hundreds of millions of customers. Every day, customers visit JD, click ads and leave billions of interaction logs. These data not only feed the learning system, but also boost technique revolutions that keep lifting both user experience and advertisers' profits on JD.com.

In the commonly used cost-per-click (CPC) advertising system, ads are ranked by effective cost per mile (eCPM), the product of bid price given by advertisers and the CTR predicted by the ad system. Accurate CTR prediction benefits both business effectiveness and user experience. Thus, this topic has attracted widespread interest in both machine learning academia and e-commerce industry.

Nowadays, most of the ads on e-commerce platforms are displayed with images, since they are more visually appealing and convey more details compared to textual descriptions. An interesting observation is that many ads get significantly higher CTR by only switching to more attractive images. This motivates a variety

of emerging studies on extracting expressive visual features for CTR prediction [2, 17]. These algorithms adopt various *off-the-shelf* CNNs to extract visual features and fuse them with non-visual features (e.g. **Category**, user) for the final CTR prediction. With the additional visual features, these algorithms significantly outperform their non-visual counterparts in *offline*-experiments and generalize well to *cold* and *long-tailed* ads. Although encouraging progress has been made in offline studies, applying CNN in real online advertising systems remains non-trival. The offline end-to-end training with CNN must be efficient enough to follow the time-varying online distribution, and the online serving needs to meet the low latency requirements of advertising system.

Furthermore, we notice that visual feature extraction in e-commerce is significantly different from the image classification setting where off-the-shelf CNNs were originally proposed. In classification, categories are regarded as the target to predict. While in e-commerce, categories of ads are clearly labeled, which contain abundant visual priors and will intuitively help visual modeling. Some academic studies have integrated the categorical information by building category-specific projecting matrix on top of the CNN embeddings [7] and by explicitly decomposing visual features into styles and categories [14]. These studies share a common architecture: the **late fusion** of visual and categorical knowledge, which however, is sub-optimal for CTR prediction. Namely, the image embedding modules seldom take advantage of the categorical knowledge. Unaware of the ad category, the embedding extracted by these CNNs may contain unnecessary features not related to this category, wasting CNN's limited expression ability. In contrast, if the ad category is integrated, CNN only needs to focus on the category-specific patterns, which will ease the training process.

To overcome the industrial challenges, we build optimized infrastructure for both efficient end-to-end CNN training and low latency online servicing. Base on this efficient infrastructure, we propose **Category-specific CNN** (CSCNN) specially for the CTR prediction task, to fully utilize the labeled categories in e-commerce. Our key idea is to incorporate the ad category knowledge into the CNN in an **early-fusion** manner. Inspired by the SE-net [11] and CBAM [22] which model the inter-dependencies between convolutional features with a light-weighted self-attention module, CSCNN further incorporates the ad category knowledge and performs a category-specific feature recalibration, as shown in Fig. 2. More clearly, we sequentially apply category-specific channel and spatial attention modules to emphasize features that are both important and category related. These expressive visual features contribute to significant performance gain in the CTR prediction problem.

In summary, we make the following contributions:

- To the best of our knowledge, we are the first to high-light the negative impact of **late fusion** of visual and non-visual features in visual-aware CTR prediction.
- We propose CSCNN, a novel visual embedding module specially for CTR prediction. The key idea is to conduct category-specific channel and spatial self-attention to emphasize features that are both important and category related.
- We validate the effectiveness of CSCNN through extensive offline experiments and Online A/B test. We verify that
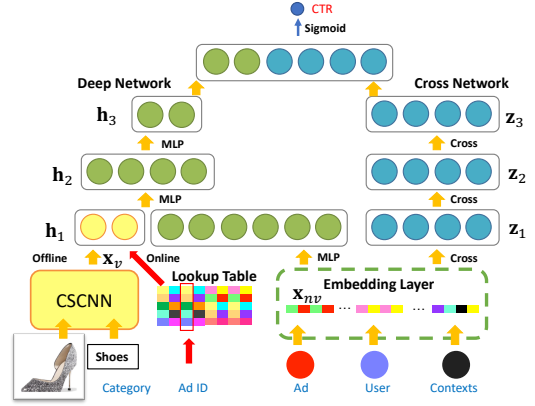

Figure 1: The Architecture of our CTR Prediction System. Bottom left: the proposed CSCNN, which embeds an ad image together with its category, to a visual feature vector $\mathbf{x}_v \in \mathbb{R}^{150}$. Note that CSCNN only runs offline. While in the online serving system, to meet the low latency requirement, we use an efficient lookup table instead. Bottom right: non-visual feature embedding, from (ad, user, contexts) to a non-visual feature vector $\mathbf{x}_{nv} \in \mathbb{R}^{380}$. Top: The main architecture, a modified DCN, which takes both the visual feature $\mathbf{x}_v$ and the non-visual feature $\mathbf{x}_{nv}$ as inputs.

the performance of various self-attention mechanisms and network backbones are consistently improved by plugging CSCNN.

- We build highly efficient infrastructure to apply CNN in the real online e-commerce advertising system. Effective acceleration methods are introduced to accomplish the end-to-end training with CNN on the 10 billion scale real production dataset within 24 hours, and meet the low latency requirements of online system (20ms on CPU). CSCNN has now been deployed in the search advertising system of JD.com, one of the largest B2C e-commerce platform in China, serving the main traffic of hundreds of millions of active users.

2 RELATED WORK

Our work is closely related to two active research areas: CTR prediction and attention mechanism in CNN.

2.1 CTR Prediction

The aim of CTR prediction is to predict the probability that a user clicks an ad given certain contexts. Accurate CTR prediction benefits both user experience and advertiser's profits thus is of crucial importance to the e-commerce industry.

Pioneer works in CTR or user preference prediction are based on linear regression (LR) [16], matrix factorization (MF) [20] and decision trees [9]. Recent years have witnessed many successful applications of deep learning in CTR prediction [4, 21]. Early works usually only make use of non-visual features, which however, is insufficient nowadays. Currently, most ads are displayed with

Table 1: Important Notations Used in Section 3

y	class label	σ	sigmoid	$\hat{y}$	predicted CTR
$\mathcal{D}$	dateset	$\mathbb{R}$	real number set	d	feature dimension
ℓ	loss	$\mathbf{x}$	feature vector	f	prediction function
l	layer index			$\mathbf{x}_{nv}$	non-visual features
$\mathbf{h}$	hidden layer			$\mathbf{x}_v$	visual features
$\mathbf{z}$	cross layer			$\mathbf{w}, \mathbf{b}$	cross layer parameter
$[]$	concatenation			d_e	embedding dimension
$\mathbf{x}_{emb}$	embedded feature			E	embedding dictionary
v	vocabulary size			$\mathbf{x}_{hot}$	one/multi hot coding
m	ad image			k	ad category
L	# layers			$\mathbf{A}_s, \mathbf{A}'_s$	spatial category prior
C	# channels			$\mathbf{A}_c$	channel category prior
W	width			$\mathcal{K}$	category set
H	height			$\mathbf{F}$	original feature map
$\mathbf{M}_s, \mathbf{M}_c$	attention map			$\mathbf{F}', \mathbf{F}''$	refined feature map
H', W', C'	size of $\mathbf{A}'_s, \mathbf{A}_c$			$\odot$	element-wise product

images which contain plentiful visual details and largely affect users' preference. This motivates many emerging studies on visual aware CTR prediction [2, 7, 8, 12, 14, 24, 26]. They first extract visual features using various off-the-shelf CNNs, and then fuse the visual features with non-visual features including categories, to build the preference predictor. Unfortunately, this late fusion is actually sub-optimal or even a waste in e-commerce scenario, where categories are clearly labeled and contain abundant visual priors that may somehow help visual feature extraction.

In contrast to existing works with late fusion, CSCNN differs fundamentally in early incorporating categorical knowledge into the convolutional layers, allowing easy category-specific inter-channel and inter-spatial dependency learning.

2.2 Attention Mechanism in CNN

Attention mechanism is an important feature selection approach that helps CNN to emphasize important parts of feature maps and suppress unimportant ones. Spatial attention tells *where* [22] and channel-wise attention tells *what* to focus on [11].

In literature, many works have attempted to learn the attention weights from the feature map, termed *self-attention*. State-of-the-art algorithms include CBAM [22], SE [11] among others [5, 10]. Besides self attention, the attention weights can also be conditioned on external information, for example nature language. Successful application fields include search by language [13], image captioning [3, 23] and visual question answering [25].

Our work is motivated by the attention mechanism. Rather than vision & language, we design novel architectures to adapt attention mechanism to address an important but long overlooked issue, the sub-optimal late fusion of vision and non-vision features in CTR prediction. We combine the advantages of both self-attention and attention conditioned on external information, namely the ad category. As a result, our image embedding is able to emphasize features that are both important and category related.

3 THE CTR PREDICTION SYSTEM IN JD.COM

We first review the background of the CTR prediction in Section 3.1. Then we describe the architecture of our CTR prediction system in Section 3.2. We further dig into details of our novel visual modeling module, the Category-Specific CNN in Section 3.3. Finally, we introduce essential accelerating strategies for online deployment in Section 3.4. The notations are summarized in Table 1.

3.1 Preliminaries

In the online advertising industry, when an *ad* is shown to a *user* under some *contexts*, this scenario is counted as an *impression*. The aim of CTR prediction is to predict the probability that a positive feedback, i.e. *click*, takes place in an impression (*ad, user, contexts*). Accurate CTR prediction directly benefits both the user experience and business effectiveness, which makes this task of crucial importance to the whole advertising industry.

CTR prediction is usually formulated as binary classification. Specifically, the goal is to learn a prediction function $f : \mathbb{R}^d \to \mathbb{R}$ from a training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{|\mathcal{D}|}, y_{|\mathcal{D}|})\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the feature vector of the i-th impression and $y_i \in \{0, 1\}$ is the class label that denotes whether a click takes place.

The objective function is defined as the negative log-likelihood:

$$\ell(\mathcal{D}) = -\frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i), \quad (1)$$

where $\hat{y}_i$ is the predicted CTR, scaled to $(0, 1)$ by sigmoid σ:

$$\hat{y}_i = \sigma(f(\mathbf{x}_i)). \quad (2)$$

3.2 The Architecture of CTR Prediction System

We now describe the architecture of our CTR prediction system that is serving on JD.com. Details are shown in Fig 1.

3.2.1 Deep & Cross Network. Deep & Cross network (DCN) [21] has achieved promising performance thanks to the ability to learn effective feature interactions. Here, we properly modify the DCN to take two inputs, a non-visual feature vector $\mathbf{x}_{nv} \in \mathbb{R}^{380}$ and a visual feature vector $\mathbf{x}_v \in \mathbb{R}^{150}$.

The visual feature is incorporated to the *deep* net. In layer 1, we transform the non-visual feature to 1024 dimension and concatenate it with the visual feature,

$$\mathbf{h}_1 = [\mathbf{x}_v, \text{ReLU-MLP}(\mathbf{x}_{nv})] \in \mathbb{R}^{150+1024} \quad (3)$$

Two deep layers follows,

$$\mathbf{h}_{l+1} = \text{ReLU-MLP}(\mathbf{h}_l), l \in \{1, 2\}, \mathbf{h}_2 \in \mathbb{R}^{512}, \mathbf{h}_3 \in \mathbb{R}^{256} \quad (4)$$

The *cross* net is used to process non-visual feature,

$$\mathbf{z}_{l+1} = \mathbf{z}_0 \mathbf{z}_l^\top \mathbf{w}_l + \mathbf{b}_l + \mathbf{z}_l, \quad (5)$$

where the input $\mathbf{z}_0 = \mathbf{x}_{nv}$, $\mathbf{z}_l, \mathbf{w}_l, \mathbf{b}_l \in \mathbb{R}^{380}$ for layer $l \in \{0, 1, 2\}$.

Finally, we combine the outputs for the predicted CTR,

$$\hat{y} = \sigma(\text{ReLU-MLP}[\mathbf{h}_3, \mathbf{z}_3]) \quad (6)$$
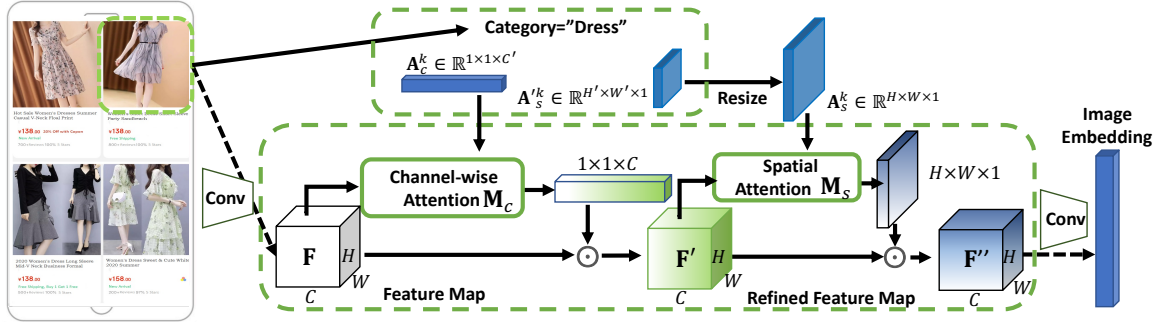
Figure 2: Our proposed Category-Specific CNN framework. Note that CSCNN can be added to any single convolutional layer, but for easy illustration, we only show details on single layer. Top: A map from the category to category prior knowledge that affects channel-wise and spatial attentions. Bottom: F is the output feature map of the current convolutional layer. Refined by channel-wise and spatial attention sequentially, the new feature map F″ is used as the input to the next layer.

3.2.2 Non-visual Feature Embedding. We now describe the embedding layer that transforms raw non-visual features of an impression, namely *(ad, user, contexts)*, to the vector $\mathbf{x}_{nv}$.

We assume that all features come in the categorical form (after preprocessings e.g. binning). Usually, a categorical feature is encoded in a one-hot / multi-hot vector $\mathbf{x}_{hot} \in \{0, 1\}^v$, where v is the vocabulary size of this feature. We show two examples below:

WeekDay=Wed $\implies$ [0,0,0,1,0,0,0]

TitleWords=[Summer, Dress] $\implies$[...,0,1,0,...,0,1,0...]

Unfortunately, this one/multi-hot coding is not applicable to industrial systems due to the extreme high dimensionality and sparsity. We thus adopt a low dimensional embedding strategy in our system,

$$\mathbf{x}_{emb} = E\mathbf{x}_{hot} \tag{7}$$

where $E \in \mathbb{R}^{d_e \times v}$ is the embedding dictionary for this specific feature and d_e is the embedding size. We then concatenate the $\mathbf{x}_{emb}$'s of all features to build $\mathbf{x}_{nv}$.

In practice, our system makes use of 95 non-visual features from users (historical clicks /purchases, location etc.), ads (category, title, # reviews etc.) and rich contexts (query words, visit time etc.) with 7 billion vocabularies in total. Setting $d_e = 4$, the total dimension is $95 \times 4 = 380$. We will further introduce the features and their statistics in Table 8 Appendix B.

3.3 Category-Specific CNN

Conventional CTR prediction systems mostly embed ad images using *off-the-shelf* CNNs. We say off-the-shelf since they were originally designed for classification, not for CTR prediction. They regard the image category as the target to predict, not as inputs. This is actually a huge waste on e-commerce platforms, where categories are precisely labeled and contain plentiful visual prior knowledge that would help visual modeling.

We address this issue by proposing a novel CNN specifically for CTR prediction, Category-Specific CNN, that embeds an ad image m, together with the ad category $k \in \mathcal{K}$, to the visual feature $\mathbf{x}_v$. Specifically, the category prior knowledge is encoded as category embeddings (trained jointly with the CTR model) and incorporated to the CNN using a conditional attention mechanism.

Theoretically, CSCNN can be adopted to any convolution layer in any network. In our systems, we plug CSCNN on ResNet18 [6] and would discuss the adaptability to other networks in ablation studies.

3.3.1 Framework on A Single Convolutional Layer. For each category k and each convolutional layer l, CSCNN learns a tensor $\mathbf{A}_c^k \in \mathbb{R}^{1 \times 1 \times C'}$ that encodes the impact of category prior knowledge on the channel-wise attention for this layer. We omit the subscript l for conciseness. The framework is shown in Fig 2.

Given an intermediate feature map $\mathbf{F} \in \mathbb{R}^{H \times W \times C}$, the output of convolutional layer l, CSCNN first learns a channel attention map $\mathbf{M}_c \in \mathbb{R}^{1 \times 1 \times C}$ conditioned on both the current feature map and the category. Then the channel-wise attention is multiplied to the feature map to acquire a refined feature map $\mathbf{F}' \in \mathbb{R}^{H \times W \times C}$,

$$\mathbf{F}' = \mathbf{M}_c(\mathbf{F}, \mathbf{A}_c^k) \odot \mathbf{F}, \tag{8}$$

where $\odot$ denotes the element-wise product with $\mathbf{M}_c$ broadcasted along spatial dimensions $H \times W$.

Similarly, CSCNN also learns another tensor $\mathbf{A}_s^k \in \mathbb{R}^{H \times W \times 1}$ that encodes the category prior knowledge for spatial attention $\mathbf{M}_s \in \mathbb{R}^{H \times W \times 1}$. These two attention modules are used sequentially to get a 3D refined feature map $\mathbf{F}'' \in \mathbb{R}^{H \times W \times C}$,

$$\mathbf{F}'' = \mathbf{M}_s(\mathbf{F}', \mathbf{A}_s^k) \odot \mathbf{F}', \tag{9}$$

where spatial attention is broadcasted along the channel dimension before element-wise product. A practical concern is the large number of parameters in $\mathbf{A}_s^k$, especially on the first a few layers. To address this problem, we propose to only learn a much smaller tensor $\mathbf{A}_s'^k \in \mathbb{R}^{H' \times W' \times 1}$, where $H' \ll H$ and $W' \ll W$, and then resize it to $\mathbf{A}_s^k$ through linear interpolation. The effect of H' and W' would be discussed with experimental results later. Note that $\mathbf{A}_s^k$ and $\mathbf{A}_c^k$ are randomly initialized and learnt during training, no additional category prior knowledge is needed except category id.

After refined by both channel-wise and spatial attention, $\mathbf{F}''$ is fed to the next layer. Note that CSCNN could be added to any CNNs, by only replacing the input to the next layer from $\mathbf{F}$ to $\mathbf{F}''$.

3.3.2 Category-specific Channel-wise Attention. Channel-wise attention tells "what" to focus on. In addition to the inter-channel
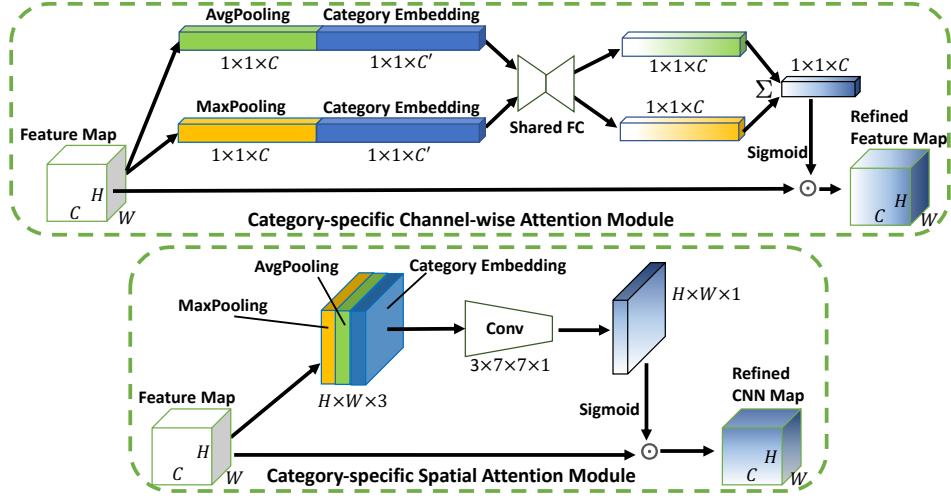
Figure 3: Modules of our proposed Category-Specific CNN: Channel-wise Attention (top) and Spatial Attention (bottom).

Table 2: # Parameters and # GFLOPs of CSCNN and Baselines. We use ResNet18 as the baseline and the backbone network to adopt CBAM and CSCNN modules. Note that there is only 0.03% addition computation from CBAM to CSCNN.

Algorithm	# params/M	#GFLOPs
Res18	17.9961	1.8206
Res18 + CBAM	18.6936	1.8322
Res18 + CSCNN	21.6791	1.8329

relationship considered previously, we also exploit the relationship between category prior knowledge and features (Fig 3, top).

To gather spatial information, we first squeeze the spatial dimension of $\mathbf{F}$ through max and average pooling. The advantage for adopting both is supported by experiments conducted by the CBAM. The two squeezed feature maps are then concatenated with the category prior knowledge $\mathbf{A}_c^k$ and forwarded through a shared two layer MLP, reducing the dimension from $1 \times 1 \times (C + C')$ to $1 \times 1 \times C$. Finally, we merge the two by element-wise summation.

$$\mathbf{M}_c(\mathbf{F}, \mathbf{A}_c^k) = \sigma(\text{MLP}[\text{AvgP}(\mathbf{F}), \mathbf{A}_c^k] + \text{MLP}[\text{MaxP}(\mathbf{F}), \mathbf{A}_c^k]), \quad (10)$$

3.3.3 Category-specific Spatial Attention. Our spatial attention module is illustrated in Fig 3 (bottom). Spatial attention tells where to focus by exploiting the inter-spatial relationship of features. Inspired by the CBAM, we first aggregate channel-wise information of feature map $\mathbf{F}'$ by average pooling and max pooling along the channel dimension. To incorporate the category prior knowledge, these two are then concatenated with $\mathbf{A}_s^k$ to form an $H \times W \times 3$ dimensional feature map. Finally, this feature map is forwarded through a 7×7 convolutional filter to get the attention weights.

$$\mathbf{M}_s(\mathbf{F}', \mathbf{A}_s^k) = \sigma(\text{Conv}_{7 \times 7}(\text{MaxP}(\mathbf{F}'), \text{AvgP}(\mathbf{F}'), \mathbf{A}_s^k)). \quad (11)$$

3.3.4 Complexity Analysis. Note that CSCNN is actually a lightweighted module. Specifically, we show the number of parameters
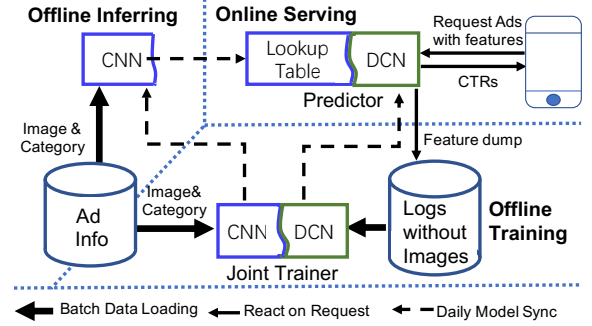
and giga floating-point operations (GFLOPs) of Baseline, CBAM and our proposed algorithm in Table 2.

We set $C \in \{64, 128, 256, 512\}$, $C' = 20$, and the bottleneck reduction ratio to 4, # categories $|\mathcal{K}| = 3310$ (real production dataset in Table 7). In the "Shared FC" in each convolutional layer in CBAM, # parameters is $2 * C * C/4$. For CSCNN # parameters in FC and channel category embedding are $C * C/4 + (C + C') * C/4 + C' * |\mathcal{K}|$. #params increased compared to CBAM in channel attention for 1 conv layer is 67~69k. And, $W' = H' = 6$, #additional params in spatial attention is $W' * H' * |\mathcal{K}| + 6 * 6 \approx 120$k. So, the total params increased is (120k+68k)*16 layers=3.0M. The additional params introduced by us are acceptable, and the additional computation is only 0.03% compared to CBAM.

3.4 System Deployment

We deploy CSCNN for the search advertising system of JD.com, the largest B2C e-commerce company in China, serving the main traffic of hundreds of millions of active users. Fig. 4 depicts the architecture of our online model system.

Figure 4: The architecture of the online model system.

Table 3: Amazon Benchmark Dataset Statistics.

Dataset	#Users	#Items	# Interact	#Category
Fashion	64,583	234,892	513,367	49
Women	97,678	347,591	827,678	87
Men	34,244	110,636	254,870	62

3.4.1 Offline training. CSCNN is trained jointly with the whole CTR prediction system, on our ten-billion scale real production dataset collected in the last 32 days. In our preliminary investigation, CNN is the key computational bottleneck during training. Taking ResNet18 network with input pictures sized 224x224, a single machine with 4 P40 GPUs can only train 177 million images per day. This means that even only considering CSCNN and with linear speedup in distributed training, we need 226 P40 GPUs to complete the training on the ten-billion impressions within 1 day, which is too expensive. To accelerate, we adopt the sampling strategy in [2]. At most 25 impressions with the same ad are gathered in one batch. The image embedding of one image is conducted only once and broadcasted to multiple impressions in this batch. Now with 28 P40 GPUs, training is can be finished in 1 day.

3.4.2 Offline inferring: Images and categories are fed into the well trained CSCNN to inference visual features. Features are made into a lookup table and then loaded in the predictor memory to replace the CSCNN. After dimension reduction and frequency control, a 20 GB lookup table can cover over 90% of the next day impression.

3.4.3 Online serving: Once a request is received, the visual feature is found directly from the lookup table according to ad id. The predictor returns an estimated CTR. Under the throughput of over 3 million items per second at traffic peak, the tp99 latency of our CPU online serving system is below 20ms.

4 EXPERIMENTAL RESULTS

We exam the effectiveness of both our proposed visual modeling module CSCNN and the whole CTR prediction system. The experiments are organized into two groups:

- Ablation studies on CSCNN aims to eliminate the interference from the huge system. We thus test the category-specific attention mechanism by plugging it onto *light-weighted* CNNs with a very *simple* CTR prediction model. We use popular benchmark datasets for repeatability.
- We further exam the performance gain of our CTR prediction system acquired from the novel visual modeling module. Experiments include both off-line evaluations on a ten-billion scale real production dataset collected from ad click logs (Table 7), and online A/B testing on the real traffic of hundreds of millions of active users on JD.com.

4.1 Ablation Study Setup

Our ablation study is conducted on the "lightest" model. This helps to eliminate the interference from the huge CTR prediction system and focus on our proposed category-specific attention mechanism.

Specifically, our light-weighted CTR model follows the Matrix Factorization (MF) framework, VBPR [8], since it has achieved state-of-the-art performance in comparison with various light models. The preference score of user u to ad a is predicted as:

$$\hat{y}_{u,a} = \alpha + \beta_u + \beta_a + \gamma_u^\top \gamma_a + \theta_u^\top \Phi(m_a, k_a), \quad (12)$$

where $\alpha \in \mathbb{R}$ is an offset, $\beta_u, \beta_a \in \mathbb{R}$ are the bias. $\gamma_u \in \mathbb{R}^{d'}$ and $\gamma_a \in \mathbb{R}^{d'}$ are the latent features of u and a. $\theta_u \in \mathbb{R}^{d_v}$ encodes the latent visual preference of u. Φ is the light-weighted CNN.

Following VBPR [8], we use CNN-F [1] as the base CNN Φ, which consists of only 5 convolutional layers and 3 fully connected layers. We plug CSCNN onto layers from conv-2 to conv-5. For comprehensive analysis, we will further test the effect of plugging attention modules on different layers (Figure 5) and our adaptability to other CNN structures (Table 6) in following sections.

4.2 Benchmark Datasets

The ablation study is conducted on 3 wildly used benchmark datasets about products on *Amazon.com* introduced in [15] [1]. We follow the identical category tree preprocessing method as used in [7]. The dataset statistics after preprocessing are shown in Table 3.

On all 3 datasets, for each user, we randomly withhold one action for validation $\mathcal{V}_u$, another one for testing $\mathcal{T}_u$ and all the others for training $\mathcal{P}_u$, following the same split as used in [8]. We report test performance of the model with the best AUC on the validation set. When testing, we report performance on two sets of items: *All* items, and *Cold* items with fewer than 5 actions in the training set.

4.3 Evaluation Metrics

AUC measures the probability that a randomly sampled positive item has higher preference than a sampled negative one,

$$\text{AUC} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{|\mathcal{D}_u|} \sum_{(i,j) \in \mathcal{D}_u} \mathbb{I}(\hat{y}_{u,i} > \hat{y}_{u,j}), \quad (13)$$

where $\mathbb{I}$ is an indicator function. i, j are indexes for ads. $\mathcal{D}_u = \{(i,j)|(u,i) \in \mathcal{T}_u \text{ and } (u,j) \notin (\mathcal{P}_u \cup \mathcal{V}_u \cup \mathcal{T}_u)\}$.

In our ablation studies, algorithms are evaluated on AUC which is almost the default off-line evaluation metric in the advertising industry. Empirically, when the CTR prediction model is trained in binary classification, off-line AUC directly reflects the online performance. In JD.com, every 1‰increase in off-line AUC brings 6 million dollars lift in the overall advertising income.

4.4 Compared Algorithms

The compared algorithms are either 1). representative in covering different levels of available information, or 2). reported to achieve state-of-the-art performance thanks to the effective use of category:

- **BPR-MF**: The Bayesian Personalized Ranking (BPR) [18], *No* visual features. Only includes the first 4 terms in Eq (12).
- **VBPR**: BPR + visual. The *visual* features are extracted from pre-trained and *fixed* CNN [8].
- **DVBPR**: The *visual* feature extractor CNN is trained *end-to-end* together with the whole CTR prediction model [12].

---

[1]Many works also use *Tradesy* [8] which is not suitable here due to the absence of category information.

Table 4: Comparison with State-of-the-arts. For all algorithms, we report the mean over 5 runs with different random parameter initialization and instance permutations. The std ≈0.1%, so the improvement is extremely statistically significant under unpaired t-test. CSCNN outperforms all due to 3 advantages: the additional category knowledge, the early fusion of category into CNN, effective structures to learn category-specific inter-channel and inter-spatial dependency.

Datasets		No Image	With Image		With Image + Category				
		BPR-MF	VBPR	DVBPR	DVBPR-C	Sherlock	DeepStyle	DVBPR-SCA	Ours
Fashion	All	0.6147	0.7557	0.8011	0.8022	0.7640	0.7530	0.8032	**0.8156**
	Cold	0.5334	0.7476	0.7712	0.7703	0.7427	0.7465	0.7694	**0.7882**
Women	All	0.6506	0.7238	0.7624	0.7645	0.7265	0.7232	0.7772	**0.7931**
	Cold	0.5198	0.7086	0.7078	0.7099	0.6945	0.7120	0.7273	**0.7523**
Men	All	0.6321	0.7079	0.7491	0.7549	0.7239	0.7279	0.7547	**0.7749**
	Cold	0.5331	0.6880	0.6985	0.7018	0.6910	0.7210	0.7048	**0.7315**

- **DVBPR-C**: DVBPR + category. The *Category* information is *late* fused into MF by sharing γ_a among items from the same category.
- **Sherlock**: DVBPR + category. *Category* is used in the linear transform *after* the visual feature extractor [7].
- **DeepStyle**: *Category* embedding is subtracted from the visual feature to obtain *style information* [14].
- **SCA**: This algorithm was originally designed for image captioning [3] where features of captioning were used in visual attention. To make it a strong baseline in CTR prediction, we slight modify this algorithm by replacing the captioning features to category embedding, so that the category information is early fused into CNN.

In literature, some compared algorithms were originally trained with the *pair-wise* loss (see Appendix A), which however, is not suitable for the industrial CTR prediction problem. For CTR prediction, the model should be trained in binary classification mode, or termed *point-wise* loss, so that the scale of $\sigma(f(\mathbf{x}))$ directly represents CTR. For fair comparison on the CTR prediction problem, in this ablation study, all algorithms are trained with the point-wise loss. While we also re-do all experiments with the pair-wise loss for consistent comparison with results in literature (Appendix A).

For fair comparison, all algorithms are implemented in the same environment using Tensor-flow, mainly based on the source code of DVBPR [2], following their parameter settings, including learning rate, regularization, batch size and latent dimension d etc. Our category prior knowledge dimension is set to $C' = 20, H' = W' = 7$. We will discuss the effects of hyper-parameters in Fig 5.

4.5 Comparison with State-of-the-arts

We aim to show the performance gain from both the incorporating and the effective utilization of valuable visual and category information. Results are shown in Table 4 [3].

First, we observe apparent performance gains from additional information along three groups, from "No Image" to "With Image", to "With Image + Category", especially on cold items. The gain from BPR-MF to group 2 validates the importance of visual features

Table 5: Adaptability to Various Attention Mechanisms (Amazon Men). Left: self attention nets. Right: our modified SE, CBAM-Channel, CBAM-All, by incorporating category prior knowledge A_c^k or A_s^k to attention using CSCNN. Results with CSCNN (right) significantly outperform original results (left), validating our effectiveness and adaptability.

	Original		+CSCNN	
	All	Cold	All	Cold
No Attention	0.7491	0.6985	–	–
SE	0.7500	0.6989	**0.7673**	**0.7153**
CBAM-Channel	0.7506	0.7002	**0.7683**	**0.7184**
CBAM-All	0.7556	0.7075	**0.7749**	**0.7315**

to CTR prediction. The gain from VBPR to DVBPR supports significance of end-to-end training, which is one of our main motivation. And the gain from group 2 to group 3 validates the importance of category information.

Second, by further comparing AUC within group 3, where all algorithms make use of category information, we find that CSCNN outperforms all. The performance gain lies in the different strategies to use category information. Specifically, Sherlock and DeepStyle incorporate the category into a category-specific linear module used at the end of image embedding. While in DVBPR-C, items from the same category share the latent feature γ_a and β_a. All of them late fuse the category information into the model after visual feature extraction. In contrast, CSCNN early incorporates the category prior knowledge into convolutional layers, which enables category-specific inter-channel and inter-spatial dependency learning.

Third, CSCNN also outperforms DVBPR-SCA, a strong baseline created by us through modifying an image captioning algorithm. Although DVBPR-SCA also early fuses the category priors into convolutional layers through attention mechanism, it lacks effective structures to learn the inter-channel and inter-spatial relationships. While our effective FC and convolutional structures in $\mathbf{M}_c$ and $\mathbf{M}_s$ are able to catch this channel-wise and spatial interdependency.

---

[2]For details, refer to https://github.com/kang205/DVBPR/tree/b91a21103178867fb70c8d2f77afeeb06fefd32c.
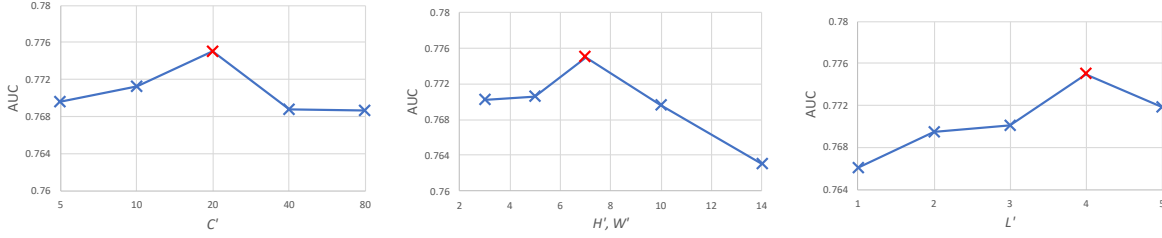[3]AUC on the first 3 columns are from [8]

Figure 5: Effects of Hyper-Parameters (Men, all). C': The size of category prior A_c^k for channel attention. H', W': The size of category prior A_c^k for spatial attention. L': Last # layers to apply CSCNN.

Table 6: Adaptability to Different Backbones (Amazon Men). We observe consistent improvement on CNN-F and Inception V1 over the self attention CBAM and no attention base, validating our adaptability.

		CNN-F	Inception
No Attention	All	0.7491	0.7747
	Cold	0.6985	0.7259
CBAM	All	0.7556	0.7794
	Cold	0.7075	0.7267
CSCNN	All	**0.7749**	**0.7852**
	Cold	**0.7315**	**0.7386**

Table 7: Real Production Dataset Statistics. Bil, Feats are short for Billion, Features. Besides the features listed, we also do manual feature interaction making the total # features= 95.

Field	# Feats	#Vocab	Feature Example
Ad	14	20M	ad id, category, item price, review
User	6	400 M	user pin, location, price sensitivity
Time	3	62	weekday, hour, date
Text	13	40M	query, ad title, query title match
History	14	7 Bil	visited brands, categories, shops
#Users 0.2 bil.	#Items 0.02 bil.	# Interact 15 bil.	#Cate 3310

Table 8: Experiments on Real Production Dataset.

Offline	AUC
DCN	0.7441
DCN + CNN fixed	0.7463 (+**0.0022**)
DCN + CNN finetune	0.7500 (+**0.0059**)
DCN + CBAM finetune	0.7506 (+**0.0065**)
DCN + CSCNN	0.7527 (+**0.0086**)

Online A/B Test	CTR Gain	CPC Gain	eCPM Gain
DCN	0	0	0
DCN+CSCNN	3.22%	-0.62%	2.46%

4.6 Adaptation to Various Attentions

Our key strategy is to use the category prior knowledge to guide the attention. Theoretically, this strategy could be used to improve any self attention module whose attention are originally learnt only from the feature map. To validate the adaptability of CSCNN to various attention mechanism, we test it on three popular self attention structures: SE [11], CBAM-Channel and CBAM-all [22]. Their results with self attention is shown in Table 5, left. We slightly modify their attention module using CSCNN, i.e. incorporating category prior knowledge A_c^k or A_s^k. Results are in Table 5, right.

Our proposed attention mechanism with category prior knowledge (right) significantly outperforms their self attention counterparts (left) in all 3 architectures, validating our adaptability to different attention mechanisms.

4.7 Adaptability to Various Network Backbones

As mentioned, our CSCNN can be easily adopted to any network backbones by replacing the input to the next layer from the feature map $\mathbf{F}$ to the refined feature map $\mathbf{F}''$. Now we test the adaptability of CSCNN to Inception V1 [19], results in Table 6.

CSCNN achieves consistent improvement on CNN-F and Inception V1 over CBAM, which validates our adaptability to different backbones. This improvement also reveals another interesting fact that even for complicated networks (deeper than CNN-F), there is still big room to improve due to the absent of category-specific prior knowledge. This again supports our main motivation.

4.8 Effects of Hyper-Parameters

We introduced 3 hyper-parameters in CSCNN: C', the size of category prior A_c^k for channel-wise attention; $H' = W'$, the size of category prior $A_s'^k$ for spatial attention; L', the number of layers with CSCNN module. Namely, we add CSCNN to the last L' convolutional layers of CNN-F. We exam their effects in Fig. 5.

When H', W', C' are small, larger H', W', C' result in higher AUC. This is because larger A_c^k and $A_s'^k$ are able to carry more detailed information about the category, which further supports the significance of exploiting category prior knowledge in building CNN. But too large H', W', C' will suffer from the overfit problem. We find the optimal setting $C' = 20$ and $H' = W' = 7$. Note that the setting is restricted to this dataset. For datasets with more categories, we conjecture a larger optimal size of H', W', C'.

When $L' \leq 4$, increasing L' benefits. Namely, the more layers that use CSCNN, the better performance. Furthermore, the continuous growth from $L' = 1$ to $L' = 4$ indicates that the AUC gain from CSCNN at different layers are complementary. But, adding CSCNN to cov1 harms the feature extractor. This is because cov1 learns general, low level features such as line, circle, corner, which never needs attention and are naturally independent to category.

4.9 Experiments On Real Production Dataset & Online A/B Testing

Our real production dataset is collected from ad interaction logs in JD.com. We use logs in the first 32 days for training and sample 0.5 million interactions from the 33-th day for testing. The statistics of our ten-billion scale real production dataset is shown in Table 7.

We present our experimental results in Table 8. The performance gain from the fixed CNN validates the importance of visual features. And the gain from finetuning validates the importance of our end-to-end training system. The additional contribution of CBAM demonstrates that emphasizing meaningful features using attention mechanism is beneficial. Our CSCNN goes one step further by early incorporating the category prior knowledge into the convolutional layers, which enables easier inter-channel and inter-spatial dependency learning. Note that on the real production data, 1‰increase in off-line AUC is significant and brings 6 million dollars lift in the overall advertising income of JD.com.

From 2019-Feb-19 to 2019-Feb-26, online A/B testing was conducted in the ranking system of JD. CSCNN contributes 3.22% CTR (Click Through Rate) and 2.46% eCPM (Effective Cost Per Mille) gain compared to the previous DCN model online. Furthermore, CSCNN reduced CPC (Cost Per Click) by 0.62%.

5 DISCUSSION & CONCLUSION

Apart from the category, what other features could also be adopted to lift the effectiveness of CNN in CTR prediction?

To meet the low latency requirements of the online systems, CNN must be computed offline. So dynamic features including the user and query should not be used. Price, sales and praise ratio are also inappropriate due to the lack of visual prior. Visual related item features including brand, shop id and product words could be adopted. Effectively exploiting them to further lift the effectiveness of CNN in CTR prediction would be a promising direction.

We proposed Category-specific CNN, specially designed for visual-aware CTR prediction in e-commerce. Our early-fusion architecture enables category-specific feature recalibration and emphasizes features that are both important and category related, which contributes to significant performance gain in CTR prediction tasks. With the help of a highly efficient infrastructure, CSCNN has now been deployed in the search advertising system of JD.com, serving the main traffic of hundreds of millions of active users.

REFERENCES

[1] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531* (2014).

[2] Junxuan Chen, Baigui Sun, Hao Li, Hongtao Lu, and Xian-Sheng Hua. 2016. Deep ctr prediction in display advertising. In *Proceedings of the 24th ACM international conference on Multimedia*. ACM, 811–820.

[3] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. 2017. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5659–5667.

[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[5] Zilin Gao, Jiangtao Xie, Qilong Wang, and Peihua Li. 2019. Global Second-order Pooling Convolutional Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3024–3033.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[7] Ruining He, Chunbin Lin, Jianguo Wang, and Julian McAuley. 2016. Sherlock: sparse hierarchical embeddings for visually-aware one-class collaborative filtering. *arXiv preprint arXiv:1604.05813* (2016).

[8] Ruining He and Julian McAuley. 2016. VBPR: visual bayesian personalized ranking from implicit feedback. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[9] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. ACM, 1–9.

[10] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Andrea Vedaldi. 2018. Gather-excite: Exploiting feature context in convolutional neural networks. In *Advances in Neural Information Processing Systems*. 9401–9411.

[11] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[12] Wang-Cheng Kang, Chen Fang, Zhaowen Wang, and Julian McAuley. 2017. Visually-aware fashion recommendation and design with generative image models. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 207–216.

[13] Shuang Li, Tong Xiao, Hongsheng Li, Bolei Zhou, Dayu Yue, and Xiaogang Wang. 2017. Person search with natural language description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1970–1979.

[14] Qiang Liu, Shu Wu, and Liang Wang. 2017. DeepStyle: Learning user preferences for visual recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 841–844.

[15] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.

[16] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*.

[17] Kaixiang Mo, Bo Liu, Lei Xiao, Yong Li, and Jie Jiang. 2015. Image feature learning for cold start problem in display advertising. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[18] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. [n.d.]. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*.

[19] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[20] Jialei Wang, Steven CH Hoi, Peilin Zhao, and Zhi-Yong Liu. 2013. Online multi-task collaborative filtering for on-the-fly recommender systems. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 237–244.

[21] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. 12.

[22] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. 2018. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 3–19.

[23] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. 2048–2057.

[24] Xiao Yang, Tao Deng, Weihan Tan, Xutian Tao, Junwei Zhang, Shouke Qin, and Zongyao Ding. 2019. Learning Compositional, Visual and Relational Representations for CTR Prediction in Sponsored Search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2851–2859.

[25] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2016. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 21–29.

[26] Zhichen Zhao, Lei Li, Bowen Zhang, Meng Wang, Yuning Jiang, Li Xu, Fengkun Wang, and Weiying Ma. 2019. What You Look Matters? Offline Evaluation of Advertising Creatives for Cold-start Problem. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2605–2613.

A ADDITIONAL EXPERIMENTAL RESULTS

In this appendix, we show many additional experimental results that are essential in supporting our claims and consistency with results reported in related works.

In all our previous ablation studies (Table 4, 5 and 6) and training for our online serving systems, we used **point-wise** loss. Namely, each impression is used as an independent training instance for binary classification. Although this is the default setting for the CTR prediction problem, we find that some existing works, which are also tested on *Amazon*, use the **pair-wise** loss (e.g. [8, 12, 18]).

Specifically, a model with pair-wise loss is trained on a dataset of triplets $(u, i, j) \in \mathcal{D}$, where (u, i, j) indicates that user u prefers ad i to ad j. Following Bayesian Personalized Ranking (BPR), their objective function is defined as

$$\max \sum_{(u,i,j) \in \mathcal{D}} \ln \sigma(\hat{y}_{u,i} - \hat{y}_{u,j}) - \lambda R, \tag{14}$$

where R is the regularization on all parameters.

To make direct comparison with existing results of pair-wise loss on the same dataset, here we re-do our ablation studies using **pair-wise loss**, with all other settings identical to that in Table 4, 5 and 6. The results are shown in Table 9, 10 and 11.

From these results, we can draw several observations. First, comparing our results in Table 9, 10 and 11 with that reported in related works, we confirm that our reimplementation of pair-wise loss and compared algorithms is comparable or sometimes better than the original literature. This validates our consistency and repeatability. Second, our CSCNN framework outperforms all compared algorithms in Table 9, 10 and 11, validating the advantages of the CSCNN framework. This superiority indicates that all our previous claims on point-wise loss still hold on pair-wise loss based models. Namely, plugging the CSCNN framework on various self-attention mechanisms and network backbones brings consistently improvement. Third, when comparing the performance across pair-wise and point-wise losses (Table 9 vs. 4, Table 10 vs. 5 and Table 11 vs. 6), we find that neither of the two losses enjoys absolute superiority over the other one in terms of AUC. However, the metric AUC only measures the relative preference of $\hat{y}_{u,i}$ compared to $\hat{y}_{u,j}$, not the absolute scale. While in practical advertising industry, point-wise loss is usually preferred since the scale of $\sigma(f(\mathbf{x}))$ trained in binary classification directly reflects the click through rate.

B STATISTICS OF OUR REAL PRODUCTION DATASETS

In this appendix, we show some specific statistics of our real production datasets, see Figure 6. Most of the features are extremely sparse, e.g. 80% of the queries have appeared less than 6 times in the training set, and follow the long tail distribution. User pin and price are relatively evenly distributed, but still 10% of the features cover 50% of the training set. As claimed in earlier studies [24], visual features contributes more when other features are extremely sparse. These statistics illustrate the difficulties in modeling on the real production dataset and the contribution of our methods.

Table 9: AUC under *Pairwise Loss,* Comparison with State-of-the Art. This table is to demonstrate our consistency to results with pair-wise loss reported in existing works on the same datasets. All other settings except the loss function are identical to that used in Table 4. Again, our proposed CSCNN framework outperforms all compared algorithms.

| Datasets		No Image	With Image		With Image + Category				
		BPR-MF	VBPR	DVBPR	DVBPR-C	Sherlock	DeepStyle	DVBPR-SCA	Ours
Fashion	All	0.6278	0.7479	0.7964	0.7993	0.7558	0.7559	0.7991	**0.8157**
	Cold	0.5514	0.7319	0.7718	0.7764	0.7359	0.7514	0.7776	**0.7941**
Women	All	0.6543	0.7081	0.7574	0.7595	0.7185	0.7244	0.7604	**0.7921**
	Cold	0.5196	0.6885	0.7137	0.7191	0.6919	0.7179	0.7186	**0.7519**
Men	All	0.6450	0.7089	0.7410	0.7483	0.7206	0.7282	0.7515	**0.7753**
	Cold	0.5132	0.6863	0.6923	0.7086	0.6969	0.7247	0.7099	**0.7303**
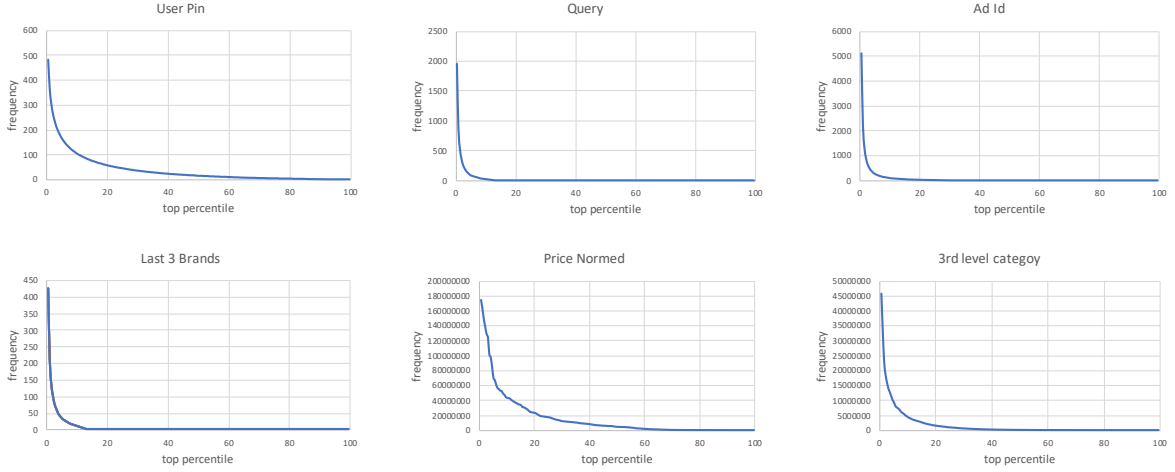
Figure 6: Feature statistics from the search advertising system of JD.com from 20200106 to 20200206 (32 days). For each feature, values are sorted in descending order of frequency.

Table 10: AUC under *Pairwise Loss,* Adaptability to Various Attention Mechanisms. All other settings except the loss function are identical to that in Table 5.

	Original		+CSCNN	
	All	Cold	All	Cold
No Attention	0.7410	0.6923	–	–
SE	0.7517	0.7094	**0.7637**	**0.7219**
CBAM-Channel	0.7541	0.7094	**0.7650**	**0.7219**
CBAM-All	0.7559	0.7081	**0.7753**	**0.7303**

Table 11: AUC under *Pairwise Loss,* Adaptability to Different Backbones (Amazon Men). All other settings except the loss function are identical to that in Table 6.

		CNN-F	Inception
No Attention	All	0.7410	0.7706
	Cold	0.6923	0.7314
CBAM	All	0.7559	0.7751
	Cold	0.7081	0.7344
CSCNN	All	**0.7753**	**0.7824**
	Cold	**0.7303**	**0.7461**