# A Visual Dialog Augmented Interactive Recommender System

Tong Yu, Yilin Shen, Hongxia Jin
Samsung Research America
Mountain View, CA, USA
{tong.yu,yilin.shen,hongxia.jin}@samsung.com

## ABSTRACT

Traditional recommender systems rely on user feedback such as ratings or clicks to the items, to analyze the user interest and provide personalized recommendations. However, rating or click feedback are limited in that they do not exactly tell why users like or dislike an item. If a user does not like the recommendations and can not effectively express the reasons via rating and clicking, the feedback from the user may be very sparse. These limitations lead to inefficient model learning of the recommender system. To address these limitations, more effective user feedback to the recommendations should be designed, so that the system can effectively understand a user's preference and improve the recommendations over time.

In this paper, we propose a novel dialog-based recommender system to interactively recommend a list of items with visual appearance. At each time, the user receives a list of recommended items with visual appearance. The user can point to some items and describe their feedback, such as the desired features in the items they want in natural language. With this natural language based feedback, the recommender system updates and provides another list of items. To model the user behaviors of viewing, commenting and clicking on a list of items, we propose a visual dialog augmented cascade model. To efficiently understand the user preference and learn the model, exploration should be encouraged to provide more diverse recommendations to quickly collect user feedback on more attributes of the items. We propose a variant of the cascading bandits, where the neural representations of the item images and user feedback in natural language are utilized. In a task of recommending a list of footwear, we show that our visual dialog augmented interactive recommender needs around 41.03% rounds of recommendations, compared to the traditional interactive recommender only relying on the user click behavior.

## CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → **Online learning settings**; *Natural language processing*; *Computer vision.*

## KEYWORDS

interactive recommender system; multimodal; visual dialog; online learning

**ACM Reference Format:**
Tong Yu, Yilin Shen, Hongxia Jin. 2019. An Visual Dialog Augmented Interactive Recommender System. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330991

## 1 INTRODUCTION

Many users are usually overwhelmed with various choices in E-commerce. As examples, Amazon and Google Express provide a huge number of products for users to browse and purchase. Recommender systems are designed to enable the users to efficiently find the items meeting their needs. In most existing recommender systems [3, 13], clicking or rating data are utilized to infer the user's preference and make personalized recommendations. However, there are important limitations when we utilize click or rating data to understand the users' preferences. The clicking or rating behavior is usually denoted by a binary or numeric variable, which is too simple to encode users' complex attitude towards different attributes of an item. For instance, the user may like the color and height of the recommended boots but dislike their closed-toe style. Obviously, this subtle preference can not be easily expressed by click or rating data. When the user can not effectively express the preferences via rating or clicking, the feedback from users may be very limited. Based on the very limited feedback, it is almost impossible for the system to learn and provide improved recommendations. Therefore, more effective user feedback to the recommended items should be designed, so that the system can effectively understand the user preference and improve the recommendations over time.

In this paper, we develop an interactive recommender system, where the user is able to provide natural language feedback to the visual appearances of some specific items in a list of recommendations. In each round, the system recommends a list of items to the user. After receiving the recommendations, the user examines this list, from the first item to the last, points to some items and give comments. With these comments, an improved list of items is recommended in the next round. We illustrate these interactions between the user and system by a use case in Figure 1. In this use case, the user aims to find blue high boots with high heel. In round 1, various shoes are recommended to the user. Visually, the second recommended shoe is blue but not the exactly desired one. So the user points to the second one and comments "*I prefer boots*". The user also comments on the fourth one by "*I prefer blue color*". With these user feedback, the system provides improved recommendations in round 2. After viewing the updated recommendations, the user makes comments on the first item "*I prefer high boots*" and also


*KDD '19, August 4–8, 2019, Anchorage, AK, USA*

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330991

on the third item "*I prefer high heel*". Based on these comments, the system provides further recommendations in the next round.

To model the user behavior of viewing, commenting and clicking on a list of recommendations in our system, we propose a visual dialog augmented cascade model. We further propose an online algorithm to online learn this model. As shown in Figure 1, exploration should be encouraged to provide more diverse recommendations in the list, such that we can quickly collect user feedback on more attributes of the items. To encourage explorations and interactively learn to recommend a list of items, we formulate the learning problem as a structured multi-armed bandit problem, cascading bandits [26, 51]. Note that traditional cascading bandit algorithms only handle user click data. To efficiently incorporate other format of data to augment the recommender system, we further propose a variant of the cascading bandit algorithm. The extra input of natural language and the visual features enables more efficient bandit learning.

Our work is related to previous works in various areas, including conversational recommender systems, interactive image retrieval and cascading bandits. However, there are three main differences between our work and previous works. First, to accurately understand the user preference, our system is multimodal and leverages multiple formats of data (images, positions, texts and clicks), while previous works usually use a subset of them. Second, to collect more user preference, our system recommends multiple items and collects multiple feedback at a time, while many previous works usually design algorithms to recommend one item at a time. Third, to efficiently collect the user preference to various attributes of items, we propose an online algorithm with explorations to make diverse recommendations in this multimodal recommender system. Please see detailed discussions in Section 2.

Our main contributions in this work are:

- We propose a novel interactive recommender system, where the user is able to provide natural language feedback to the visual appearances of some specific items in a list of recommendations. This enables the user to precisely express their preference and quickly find the desired items.
- We propose a visual dialog augmented cascade model to model the user behaviors of viewing, commenting and clicking on a list of recommended items.
- We propose a learning variant of the cascading bandits to balance exploration and exploitation, and efficiently learn the visual dialog augmented cascade model.

## 2 RELATED WORK

Our visual dialog interactive recommender system is related to the previous works in conversational recommender systems, interactive image retrieval, cascade models and cascading bandits.

### 2.1 Conversational Recommender System

Conversational recommender systems enable the user to express the opinions about current recommendations in conversations, to guide the system in providing further recommendations. Bridge [6] proposes an approach that uses both a dialogue grammar and a recommendation strategy, to achieve local and global dialogue coherence. Feature selection and multi-armed bandits have been applied to the question selection task in conversational recommender systems [8, 33]. To effectively understand the conversations and maximize the long-term benefits, deep learning and reinforcement learning based approaches are proposed in [15, 28, 42].


**Figure 1: A use case in our interactive recommender system. In each round, the user points to the items in the recommended list and provides comments. Based on the comments, the system provides an updated list of recommendations in the next round.**

Our system is different from the previous works in conversational recommender systems, as follows. First, our approach is multimodal such that different formats of user behavior data are used in the modeling. After receiving the recommendations, the user examines this list, from the first item to the last, points to some items and gives comments on the visual appearances of these items. If an item meets the user needs, it will be clicked. During these interactions between the user and system, various formats of data (images, positions, texts and clicks) are collected to infer the user preference and make accurate recommendations. Second, compared to many previous works where only one item is recommended in each round, our system recommends multiple items in each round. By recommending multiple items, multiple user feedback can be collected. Then, our model updates more efficiently and finds the desired items with less rounds, compared to the previous works where only one item is recommended. Recommending a list of multiple items is non-trivial in a real-world recommender system. We need to model the user behaviors of viewing, commenting and clicking on a list of items.

### 2.2 Image Retrieval with User Feedback

Image retrieval with user feedback has been comprehensively studied [14, 37, 45]. The user feedback can be in the form of relevance feedback [38, 47], or relative attributes feedback [23–25, 34, 49]. In these works, the set of image attributes is usually pre-defined and fixed. In contrast, Guo *et al.* [17] proposes an end-to-end approach to learn more flexible and accurate representation of the image attributes. User natural language feedback are also widely utilized in many other computer vision tasks, including image or video retrieval based on queries in natural language [2, 19, 29, 44], image

captioning [35, 46], visual question answering [1, 43], and visually grounded dialog systems [10–12, 39, 41].

To model the user natural language feedback to the visual appearances of a list of items, one component of our system follows the designs of the encoder and state tracker in [17]. There are two major differences between this work and Guo *et al.* [17]. First, in each round, only one item is retrieved in Guo *et al.* [17] while a list of items is recommended in our use case. Retrieving or recommending only one item at a time is inefficient and impractical in many real-world systems. For example, in personal assistants with screens (*e.g.*, Amazon Echo Show) or online shopping webpages, it is desirable to recommend multiple items at a time, since the users can have multiple options and user feedback to multiple items can be collected efficiently. It is non-trivial to directly extend the approach in [17] to support multiple recommendations at each time, because we need to model the user behaviors of viewing, commenting and clicking on a list of recommended items. To model these user behaviors, we propose a novel user behavior model as detailed in Section 3.2.1. Second, Guo *et al.* [17] focus on an image retrieval task and the goal is to retrieve a particular image, while we develop a recommender system and the goal is to find as many desired items as possible. With the different goals, a greedy approach without any exploration is used to find the image in [17], but exploration and exploitation should be balanced in our system to find more desired items efficiently. To efficiently find the desired items in our visual dialog augmented interactive recommender system, we propose a novel online algorithm as detailed in Section 3.2.3.

## 2.3 Cascade Models and Cascading Bandits

The cascade model is surprisingly effective in explaining how users scan lists of items [9, 26, 51]. The user examines the list, from the first item to the last item. The first attractive item is clicked. If none of the item within the list is attractive to the user, the user does not click on any item. Similarly, in our system, we aim to model the user behaviors on the list of recommendations. However, the traditional cascade model only considers the click behavior. To model the user comment behavior in addition to the click behavior, we propose an extension of the cascade model as detailed in Section 3.2.1.

To efficiently learn to recommend the list with maximized number of clicks in an online setting, an online learning variant of the cascade model is proposed in [26], which is referred as cascading bandits. Bandit algorithms handle the exploration and exploitation problem. At each step, the algorithm chooses an arm (*i.e.*, takes an action) and observes a reward from the environment. Based on the reward, the algorithm adapts the strategy of choosing the arm in the next step. The goal of the algorithm to maximize the reward over time. In cascading bandits, each arm corresponds to a list of items. For example, in a movie recommender, usually the system recommends a list of movies at a time. Then, the users click on some movies in the list of recommendations. The user feedback (*i.e.*, number of clicks) on the recommended list of movies, is the reward to the system. We want to maximize the number of user clicks over time. Several bandit algorithms [21, 26, 51] are proposed to maximize the reward in this setting where a list of items are recommended at each time. The major difference between our approach and these algorithms [21, 26, 51] is that we leverage the neural representation of the item visual appearance and user feedback to design more efficient exploration, as detailed in Section 3.2.3. Augmenting bandits by deep neural networks has recently been studied in [30, 36, 40, 50].

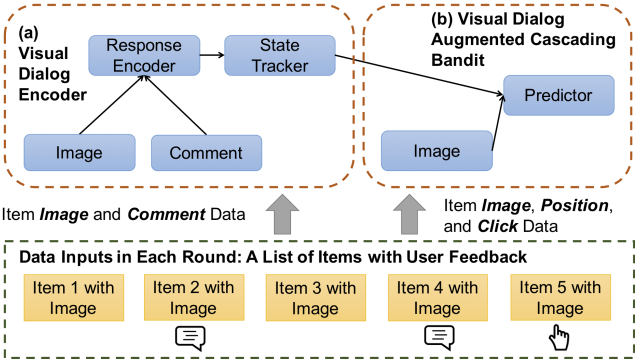


**Figure 2: Our visual dialog augmented interactive recommender system consists of two components: (a) visual dialog encoder and (b) visual dialog augmented cascading bandit. The data inputs in each round include item images, item positions, user clicks and comments.**

# 3 METHODS

We introduce our methods. First, we overview the components in our system. Second, we propose a variant of cascading bandit, which is augmented by the items' images and user comments in natural language.

## 3.1 Visual Dialog Augmented Interactive Recommender System

We describe an overview of our proposed visual dialog recommender system, as shown in Figure 2.

The data inputs to our system are the recommended lists and the user feedback. At each time, the user is provided with a list of items. If the user finds a desired item, the user will click it. The user may comment on some items, to express their preference to the item attributes. We may instruct the user to comment on some specific types of items when deploying the system. For example, one instruction can be "*Please comment on the items similar to your desired items to get further recommendations*". When a recommended item is very different from the desired item, it can be difficult for the user to use short sentences to describe the desired item based on the current recommended item. We discuss how these instructions affect the recommendations in Section 4.2.2.

There are two major components in our proposed visual dialog augmented interactive recommender system.

The first component is the *visual dialog encoder*. The visual dialog encoder consists of a response encoder and a state tracker. We follow the *Response Encoder* and *State Tracker* in Section 3.1 of Guo *et al.* [17]. The response encoder's input is a pair of item image $e$ and its comment $o_e$. The input image is first encoded by a convolutional neural network (CNN) and a linear mapping. The structure of the CNN follows ResNet101 [18]. The parameters of

the CNN are pre-trained on ImageNet and fixed. The words of the user feedback are represented by one-hot encoding. Following [22], this encoded words are used as the input to a linear mapping and then a CNN. The encoded image and words are further concatenated as the input to a state tracker. The output of the response encoder is used as the input of GRU followed by a linear mapping to achieve the final encoding of the pair of item image and its comment. This memory network by GRU is to model the sequence of item and comment pairs over time. The output of the state tracker is denoted as VisDiagEnc$(e, o_e)$. Let ResNetEnc$(j)$ be the representation of the desired item $j$ by ResNet101 followed by a linear mapping. By optimizing the loss functions in [17], the distance between VisDiagEnc$(e, o_e)$ and ResNetEnc$(j)$ is minimized. That is, the output of the visual dialog encoder is similar to the representation of the desired images.

The second component is the *visual dialog augmented cascading bandit*. The traditional cascading bandits only model the user click behavior. Instead, our visual dialog augmented cascading bandit receives additional information about the viewed items and user comments from the visual dialog encoder. A shown in Figure 2, to predict whether an item should be recommended, two factors are considered: the content of the item's image, and whether the representation of this item is close to the output of the visual dialog encoder. The details of this component are described in Section 3.2.

The models of the two components are updated as follows. The first component, visual dialog encoder, is pre-trained on a training dataset in the offline setting with the loss functions designed in [17]. The second component is trained in an online setting for a new user (not necessarily existing in the training data) to provide personalized recommendations.

## 3.2 Visual Dialog Augmented Cascading Bandit

First, we propose a novel visual dialog augmented cascade model to model the user behavior of viewing, commenting and clicking on a list of items. Second, we describe the online learning setting to learn this visual dialog augmented cascade model. Third, we propose a learning variant of the cascading bandits, the visual dialog augmented cascading bandit.

*3.2.1 Visual Dialog Augmented Cascading Model.* To model the user behaviors of viewing, commenting and clicking on a list of recommended items, we propose a novel visual dialog augmented cascading model.

Figure 3 shows our proposed model. We define the following variables to represent the user behavior: $a_r$, $V$, $U$, $C$ and $O$. The variable $a_r$ indicates the $r$-th item from the start of the list. The $V(a_r)$ indicates whether the item in the $r$-th position is attractive or not[1], $U(a_r)$ indicates whether the item in the $r$-th position is examined by the user or not, $C(a_r)$ indicates whether the item in the $r$-th position is clicked by the user or not, and $O(a_r)$ indicates whether the item in the $r$-th position is commented by the user or not. Similar to the traditional cascade model [9], we define several dependencies among these variables. The relations among these

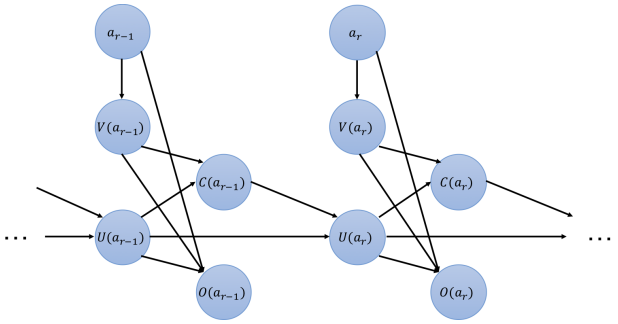


**Figure 3: The visual dialog augmented cascade model.**

variables are shown in Figure 3. First,

$$\begin{aligned} P(U(a_r) = 1 | U(a_{r-1}) = 1, C(a_{r-1}) = 0) &= 1 \\ P(U(a_r) = 1 | U(a_{r-1}) = 1, C(a_{r-1}) = 1) &= 0 \\ P(U(a_r) = 1 | U(a_{r-1}) = 0) &= 0. \end{aligned} \tag{1}$$

If an item is examined but not clicked, the user is going to examine the next item. If an item is examined and clicked, the user is not going to examine any items after current item. If an item is not examined, the user is not going to examine any items after current item. Second, the probability of the click behavior is

$$\begin{aligned} P(C(a_r) = 1 | U(a_r) = 1, V(a_r) = 1) &= 1 \\ P(C(a_r) = 1 | U(a_r) = 1, V(a_r) = 0) &= 0 \\ P(C(a_r) = 1 | U(a_r) = 0) &= 0, \end{aligned} \tag{2}$$

which mean that only when an item is attractive and examined, the user will click the item. Third, the probability of the comment behavior is

$$\begin{aligned} P(O(a_r) = 1 | U(a_r) = 1, V(a_r) = 1) &= 0 \\ P(O(a_r) = 1 | U(a_r) = 1, V(a_r) = 0, a_r) &= \beta \\ P(O(a_r) = 1 | U(a_r) = 0) &= 0. \end{aligned} \tag{3}$$

When an item is attractive and examined, the user is satisfied with the item and not going to comment the item to approach the desired item[2]. When an item is examined but not attractive, the user is going to comment the item with probability $\beta$. The value of $\beta$ depends on whether $a_r$ is examined, the content of $a_r$ and the user interest. The system can also affect the value of $\beta$ by providing instructions to users when the system is deployed. An example of instruction is "*Please comment on the items similar to your desired items to get further recommendations*". With this instruction, the items more similar to the desired items have higher $\beta$.

Although the assumptions of the dependencies between the variables of different user behaviors are simple, they are able to explain position bias and widely used in modeling the user behavior on a list of items [9, 16, 20, 21, 26, 32, 51].

[1] Attractive is a common term in click models [9]. Attractive and desired are used interchangeably in this paper.

[2] We instruct the user to provide comments to describe the desired visual attributes. The comment in our system is different from the traditional product reviews in a recommender system. The traditional reviews mainly discuss the current items, while the comments in our system are mainly about the attributes of the desired item.

*3.2.2 Setting of the Online Learning Problem.* The online learning setting is similar to the cascading bandits [26, 51]. There are $L$ items in total, $E = [L]$ is a ground set of the $L$ items and $P$ is a probability distribution over a binary hypercube $\{0,1\}^E$. The learning agent interacts with our problem as follows. Let $(V_t)_{t=1}^n$ be an i.i.d. sequence of $n$ weights drawn from $P$, and $V_t(e)$ is the preference of the user to item $e$ at time $t$. That is, $V_t(e) = 1$ if and only if item $e$ is attractive to user at time $t$. At each time $t$, the agent recommends a list of $K$ items $A_t = (a_1^t, \cdots, a_K^t) \in \Pi_K(E)$ to the user, where $\Pi_K(E)$ is the set of all $K$-permutations of set $E$. The list is a function of the observations up to time $t$ by the agent. The user examines the list, from the first item $a_1^t$ to the last item $a_K^t$. The first attractive item is clicked. If no item in the list is attractive to the user, the user does not click on any item. That is, the click feedback at time $t$ is $C_t = \min\{k \in [K] : V_t(a_k^t) = 1\}$, where it is assumed that $\min \emptyset = \infty$. Before the user clicking any item in the list, the user may comment on some items. The comment feedback at time $t$ is $O_t = \{ o_{a_k^t} \mid O_t(a_k^t) = 1 \text{ and } k < C_t \}$. The reward is defined as

$$f(A, V) = 1 - \prod_{k=1}^{K} (1 - V(a_k)), \tag{4}$$

which means that the reward is one if and only if at least one of the item attracts the user.

Following [26, 51], let the attraction weights in the ground set $E$ be independently distributed as $P(V) = \prod_{e \in E} \text{Ber}(V(e), \bar{V}(e))$, where $\text{Ber}(\cdot, \mu)$ is a Bernoulli distribution with mean $\mu$. Then, the expected reward of list $A \in \Pi_K(E)$ is $\mathbb{E}(f(A, V)) = f(A, \bar{V})$. Therefore, it is sufficient to learn a good approximation to $\bar{V}$ to act optimally. We discuss how to learn this approximation in Section 3.2.3. With the definition of reward, the regret is defined as

$$R(n) = \mathbb{E}\left[\sum_{t=1}^{n} R(A_t, V_t)\right], \tag{5}$$

where $R(A_t, V_t) = f(A^*, V_t) - f(A_t, V_t)$ is the instantaneous stochastic regret of the learning agent at time $t$, and

$$A^* = \underset{A \in \Pi_K(E)}{\arg\max}\ f(A, \bar{V}) \tag{6}$$

is the optimal list of items in hindsight by assuming all data is known in advance. The goal of the learning agent is to minimize the cumulative regret.

*3.2.3 Visual Dialog Augmented Cascading Bandit Algorithm.* As discussed in Section 3.2.1, the key is to learn a good approximation to $\bar{V}$ for the agent to act optimally. Inspired by the efficient cascading bandit learning for large-scale problems [51], we parameterize the model in Section 3.2.1 as follows. In our task, $\bar{V}(e)$ is a Bernoulli variable (attractive or not) and the items are represented by image features. Thus, we approximate the $\bar{V}$ function by assuming there exists $\theta^* \in \mathbb{R}^{d \times 1}$ such that for any $e \in E$

$$P(\bar{V}(e) = 1) \approx \sigma(x_e^\top \theta^*) = \sigma(\text{ResNetEnc}(e)^\top \theta^*), \tag{7}$$

where $\sigma(.)$ is the sigmoid function and $x_e = \text{ResNetEnc}(e) \in \mathbb{R}^{1 \times d}$ is the representation of $e$ by ResNet and then a linear mapping. Here the ResNet and the linear mapping is pre-trained in offline setting as discussed in Section 3.1, while $\theta^*$ is unknown to the learning agent in online learning. Note in the model proposed in Section 3.2.1, observing $C(e)$ is enough for us to know the value of $V(e)$: in a list, for all the items up to the *first clicked* item, $V(e) = C(e)$ according to (2). Therefore, we can learn $\theta^*$ to approximate $\bar{V}$ only based on the click data, as CascadeLinTS does in [51]. However, inspired by [17], we can leverage the comment data to achieve more efficient model learning. Assume $\mathcal{D}$ is the set of the desired items. Given an image $e$ and its comment $o_e$, the visual dialog encoder in Section 3.1 outputs a representation $x_e^{enc}$, such that the distance between $x_e^{enc}$ and $\text{ResNetEnc}(j)$ is minimized, where $j \in \mathcal{D}$. That is, if $x_e^{enc}$ is known and accurate, the learning agent should prefer to recommend the image $a$ whose $\text{ResNetEnc}(a)$ is closer to $x_e^{enc}$.

According to the above discussions, to predict whether an item $e'$ should be recommended, two factors are considered: whether the item is attractive according to $\bar{V}(e')$, and whether $\text{ResNetEnc}(e')$ is close to the output of the visual dialog encoder. This consideration is illustrated in Figure 2. Based on this idea, our algorithm is proposed and summarized in Algorithm 1. The inputs are $\lambda$ for the Gaussian prior $\mathcal{N}(\bar{\theta}_t, \lambda^{-1} I_d)$, the total number of items $L$, the list size $K$ and the dimensionality $d$ of $\text{ResNetEnc}(e)$ and a tunable parameter $K'$. The algorithm is online and runs for $n$ steps. At each step, we first sample the model parameter $\theta_{t-1}$ from its posterior, as shown in line 4. Second, from line 5 to 8, from set $\mathcal{L}$ we choose the items with the top $K$ highest expected reward, and recommend these items to the user. Third, the user feedback are collected, as shown from

**Algorithm 1:** Visual Dialog Augmented Cascading Bandit

1 **Input:** $\lambda$, $L$, $K$, $K'$, $d$
2 $\tau = 1$, $\tau' = 1$, $\bar{\theta}_1 = \mathbf{0}$, $S_1 = \lambda^{-1} I_d$, $x^c = \mathbf{0}$, $\mathcal{L} = [L]$
3 **forall** $t = 1, \cdots, n$ **do**
4  Sample the parameter $\theta$ from its posterior $\theta_t \sim \mathcal{N}(\bar{\theta}_t, S_t)$
5  **forall** $k = 1, \cdots, K$ **do**
6  $a_k^t \leftarrow \arg\max_{e \in \mathcal{L} - \{a_1^t, \cdots, a_{k-1}^t\}} x_e^\top \theta_t$
7  **end**
8  $A_t \leftarrow (a_1^t, \cdots, a_k^t)$
9  Observe click $C_t \in \{1, \cdots, K, \infty\}$
10  **forall** $k = 1, \cdots, \min\{C_t, K\}$ **do**
11  $e \leftarrow a_k^t$
12  **if** *the user gives a comment* $o_e$ *on item* $e$ **then**
13  $x_e^{enc} \leftarrow \text{VisDiagEnc}(e, o_e)$
14  $x^c \leftarrow \frac{x^c \times (\tau' - 1) + x_e^{enc}}{\tau'}$, $\tau' \leftarrow \tau' + 1$
15  **end**
16  $z_\tau \leftarrow x_e$, $y_\tau \leftarrow \mathbb{1}\{k = C_t\}$, $\tau \leftarrow \tau + 1$
17  **end**
18  **forall** $k' = 1, \cdots, K'$ **do**
19  $b_{k'} \leftarrow \arg\min_{e \in [L] - \{b_1, \cdots, b_{k'-1}\}} ||\text{ResNetEnc}(e) - x^c||$
20  **end**
21  **if** $\tau' > 1$ **then** $\mathcal{L} \leftarrow (b_1, \cdots, b_{k'})$ ;
22  Estimating $\bar{\theta}_{t+1}$ by solving (8)
23  $S_{t+1} = \left(\sum_{i=1}^{\tau} \sigma(z_i^\top \theta_t)(1 - \sigma(z_i^\top \theta_t)) z_i z_i^\top + \lambda I_d\right)^{-1}$
24 **end**

line 9 to 17. Given the input of an item $e$ and its comment $o_e$, the visual dialog encoder outputs a representation $x_e^{enc}$. Then, $x^c$ is updated as the average of $x_e^{enc}$ for all item $e$ being commented over time. In line 16, the user click data is collected. The set $\mathcal{L}$ is updated as these items whose representations are the top $K'$ closest to $x^c$, as shown from line 18 to 21. Finally, we update the posterior of $\theta$ in line 22 and 23. We assume the prior of $\theta_{t-1}$ is sampled from a Gaussian distribution and the data likelihood is sampled from a Bernoulli distribution. Then, based on Laplace approximation [5, 7], the posterior of $\theta_{t-1}$ is approximated by another Gaussian distribution, whose mean is estimated in line 22 by solving

$$\bar{\theta}_t = \arg\min_{\theta} \lambda\theta^\top\theta - \sum_{i=1}^{\tau} y_i \log(\sigma(z_i\theta)) - \sum_{i=1}^{\tau}(1-y_i)\log(1-\sigma(z_i\theta)), \quad (8)$$

and covariance is estimated in line 23. Compared to the cascading bandits [51], the neural representation of the comments and the commented items' visual appearance is used as the extra input of our algorithm. It reduces the set of candidate items by filtering out some items visually different from the user's comments, and enables us to quickly identify more items with positive feedback in the bandit learning. Similar to [31], this design is to prevent the system from exploring more than needed.

# 4 EXPERIMENTS

## 4.1 Experimental Setup

**Dataset and Setting.** Our system is evaluated on a footwear dataset [4, 17]. In the evaluation, each run of the experiment corresponds to a use case where the user aims to find the items belonging to a specific category. Some examples of the category are "sneakers", "boots" and "flats". If an item belongs to the desired category of a user, the item will be clicked. Otherwise, the item will not be clicked. If at least one item within the recommended list is clicked by the user, the system will receive a reward of 1. If no item within the recommended list is clicked by the user, the system will receive a reward of 0. Following the previous work [17], the visual dialog encoder is trained on 10,000 images. We evaluate different interactive recommenders on a separate dataset including 4,658 images.

Similar to [8, 17], a challenge in the evaluation is that we need to have access to user feedback to any possible list of recommendations. In practice, it is exhaustive and unrealistic to collect the user feedback to all possible lists, because the number of possible lists is a high order polynomial function of the total number of shoes. When we choose $K$ items from 4,658 items to form the recommended list, in total there are $\binom{4,658}{K}$ lists. Following [17], we adopt a similar setting where the user comments on all possible lists are generated, by a relative captioner trained on a real-world dataset. We consider two questions when generating user comments.

*4.1.1 What the User Will Comment.* We use the *relative captioner* [17] to act as a surrogate for real human users by generating their natural language comments, which describe the visual differences between any pair of desired and candidate images[3]. A dataset with 10,751 pairs of desired item and candidate item is collected to train this captioner. Given a pair of images, one comment in natural language is collected by crowdsourcing in Amazon Mechanical Turk. A simplified but regular, specific and relative comment is adopted. When collecting the data, the annotators act as the customers to talk with the shopping assistant. Specifically, the annotators describe the desired attributes of items by completing sentences with given prefix. All images are encoded by a pre-trained ResNet101 [18]. Then, the image encodings of the desired item and candidate item are concatenated as the input of the *Show, Attend and Tell* model [48], to generate the relative captions. A visual attention is introduced to better capture the localized visual differences. As the comprehensive study shows in [17], this captioner outperforms other alternatives and only has minor errors in human evaluations.

The input desired item to the captioner in our experiments is different from that in [17]. In the evaluation of the image retrieval system in [17], in each use case there is always only one desired image. However, in a recommender system, usually multiple items meet the user's need. Finding one of the desired items results in a successful recommendation, no matter which one of them is found. Therefore, in our experiments, each time when the captioner generates the comments, the input desired item is randomly chosen from the items belonging to the desired category.

In our experiments, each time a user observe a list of recommended items. If a user decides to comment on one item, the comment is generated by the above caption generator. However, it is still unclear whether the user will give a comment on a specific item in the list of items. We discuss it in Section 4.1.2.

*4.1.2 Whether the User Will Comment.* We discuss several experimental settings where the user has different behaviors in deciding which item to comment on. Given a list of recommendations, if an item perfectly meets the user requirement, we do not expect the user to comment on it[2]. If the user has clicked an item, the user will not comment on any item after the clicked item [9]. Otherwise, the user may comment on an item with probability $\beta$ depending on the content of the item as discussed in Section 3.2.1.

Ideally, we should collect the user comments on all possible lists as groundtruth, which is unrealistic because the number of possible lists is a high order polynomial function of the total number of shoes. As alternatives, we discuss the following settings. First, a simple yet reasonable experimental setting is that, the user randomly comments on the items before the user finds the desired item in the list of recommendations. Second, as discussed in Section 3.1 and 3.2.1, we can also affect the user commenting behavior by giving user instructions. A reasonable instruction is "*Please comment on the items similar to your desired items to get further recommendations*". With this instruction, the items more similar to the desired items have higher probability of being commented. Third, oppositely we can also instruct the user to comment on the items different from their desired item. All the above settings are feasible in practice and we compare them in Section 4.2.2.

**Research Questions.** Our experiments are designed to address the following research questions:
RQ 1. Can user natural language feedback to the items' visual appearances improve the traditional recommender system?

[3] We follow the captioner code at https://github.com/XiaoxiaoGuo/fashion-retrieval.

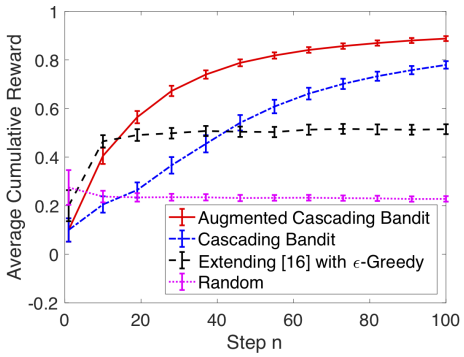


**Figure 4: The comparison between our augmented cascading bandit and other baseline methods.**

RQ 2. If the system can instruct the user to comment on some items, what kinds of items should be commented?
RQ 3. How does the number of comments affect the performance?
RQ 4. Will increasing the size of the recommended list improve the performance?

**Evaluation Metric.** We evaluate different approaches under the metric of average cumulative reward $r(n) = \frac{1}{n}\sum_{t=1}^{n} f(A_t, V_t)$ [7, 26, 27, 51]. For each category of desired item, we conduct 10 runs of experiments. Over all runs for all categories, the average results along with the standard errors are reported. We show the results up to $n = 100$ steps, when the algorithms converge. A step of the algorithms corresponds to a round in the use case.

## 4.2 Experimental Results

We show and discuss various results to address the above research questions 1 to 4. Finally, a use case is to illustrate how user interacts with the system to get satisfied recommendations.

*4.2.1 Can User Natural Language Feedback to the Items' Visual Appearances Improve the Traditional Recommender System? (RQ 1).* In this section, we show how traditional interactive recommender system improves when it is augmented by visual dialog. We compare our system to the interactive recommender system without incorporating the visual dialog. That is, we compare the proposed visual dialog augmented cascading bandit to a traditional cascading bandit algorithm [51]. For sanity check, we also evaluate two other baselines. The first baseline randomly select $K$ items to recommend at each time. The second baseline extends the approach in [17] to handle multiple items. At each step, we collect the pairs of image and its comment in the current list. The average representation of these pairs by the response encoder and then state tracker are used as the input of the KNNs to sample top $K$ candidate images as the list of recommendations for the next step. To introduce explorations, we use $\epsilon$-Greedy when sampling top $K$ recommendations. With probability $\epsilon$, we randomly sample $K$ items from the test data to form the recommended list . We report the results when $\epsilon = 0.1$. We tried different values of $\epsilon$ (even $\epsilon = 0$ with no explorations). The performance of this approach is not that sensitive to $\epsilon$. In the experiments in this section, the recommended list has $K = 10$ items, and the user is assumed to randomly comment on 10% items within all items in test data.

We report the results in Figure 4. First, significant improvements are achieved by the augmented interactive recommender system, compared to the traditional interactive recommender system. Specifically, after 10 steps, the augmented recommender system achieves 2 times reward, compared to the traditional recommender system: the traditional one suggests the desired item in the list with probability 0.2, while the augmented one finds the desired item in the list with probability more than 0.4. After 10 steps, this probability gap further increases from 0.2 to 0.3 at step 40. To achieve average cumulative reward of 0.5, our system needs around 41.03% rounds (*i.e.*, steps) of recommendations, compared to the traditional interactive recommender only relying on the user click data. These results clearly show the benefits of introducing visual dialog to a traditional interactive recommender system. Second, our method outperforms the baselines of $\epsilon$-Greedy and Random. $\epsilon$-Greedy performs better than our method before first 14 steps, because our method explores the item space. Later on, our method exploits more and achieves more than 0.3 improvement of average cumulative reward at step 100, compared to $\epsilon$-Greedy.

*4.2.2 If the System Can Instruct the User to Comment on Some Items, What Kinds of Items Should be Commented? (RQ 2).* We study how the instructions can potentially change the user experience and system performance. Two possible instructions are (i) *"Please comment on the items similar to your desired items to get further recommendations"* and (ii) *"Please comment on the items different to your desired items to get further recommendations"*. Under these two instructions, we assume the user will always comment on the items visually (i) similar or (ii) different from their desired items.

Figure 5a shows the results. Compared to the setting where user randomly comment on some items, about 0.05 reward improvement can be achieved by the setting where user is instructed to comment on similar or different items. This means that, even the user just randomly choose some items to comment, it is enough for our system to leverage these comments to improve. Encouraging the user to comment on some specific items can helpfully increase the performance of the system, especially in the first 30 steps.

*4.2.3 How Does the Number of Comments Affect the Performance? (RQ 3).* With imperfect recommendations, some users are more likely to provide comments while others are less likely to provide comments. Thus, we vary the number of comments in the recommended list at each time, to understand how this factor affects the performance. In each experiment, the user randomly comments on different ratios of items among all the items in the test data.

The results are shown in Figure 5b. When the ratio is low (*i.e.*, 5%), increasing comments lead to higher reward. This matches the intuition that with more comments from the user, we can better understand the user preference. As the ratio increases to 30%, the improvement becomes limited and the results converge. This indicates that it is not necessary for the users to comment on all items to achieve the best recommendation performance.

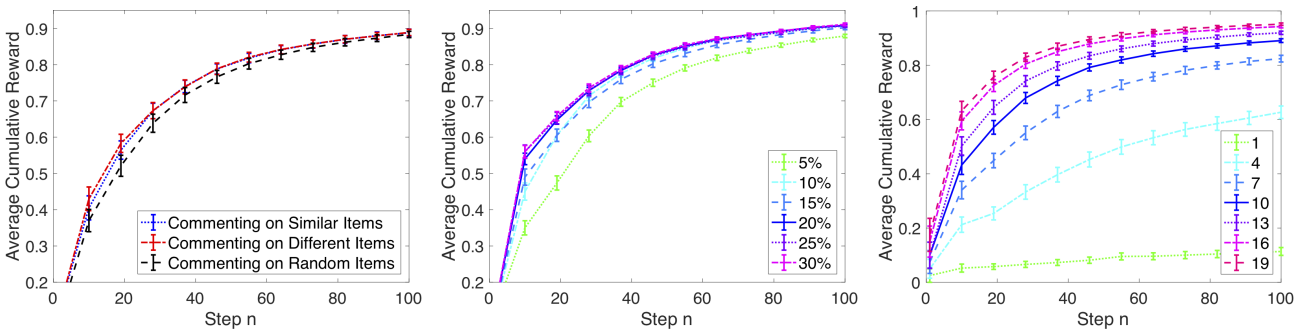


**Figure 5: The results of different experiments: (a) the results by different comment behaviors, (b) the effects of varying the number of comments, and (c) the effects of increasing the size of the recommended list.**


**Figure 6: A use case where the user interacts with our system to find flats. In round 1, the user comments on the third item by "*I prefer black with a wedge heel*", and also comments on the fourth item by "*I prefer black with a wedge heel*". In round 4, the user comments on the first item and the second item. The third item satisfies the user requirement and is checked by the user. A round in the use case corresponds to a step of the algorithms.**

*4.2.4 Will Increasing the Size of the Recommended List Improve the Performance? (RQ 4).* Intuitively, increasing the size of the recommended list helps improve the performance, because the recommender gives the user more options and collects more user feedback in each round. In this section, we validate this by varying the number of items in the recommended list in each round and reporting the results.

Figure 5c shows the results. We have two observations. First, when we increase the size of the recommended list, the performance of our system increases. Especially, there is a significant improvement when we increase the size from 1 to 4. This validates that recommending more than one items is very necessary for the system to achieve good performance. Second, when we keep increasing the size, the improvement becomes more and more limited. This indicates that when the size of the list is large enough, increasing the size will not always improve the performance.

*4.2.5 A Use Case.* We show a use case in Figure 6 to illustrate how our recommender learns with increasing number of interactions between the user and system. Due to limited space, we show a use case where $K = 5$ items are recommended in each round. The first row gives some examples of the desired flats in this use case. Then, the recommended list of items and the user comments in rounds 1, 4, 7, and 10 are presented. We observe that the comments in these rounds are successfully understood by the system to improve the recommendations. In rounds 4 and 10, the user scans the list and finds desired flats. Note that the recommendations in round 7 is a bit irrelevant to the user desired items. One possible reason is that the algorithm is still exploring the item space, when the number of rounds is small.

## 5 CONCLUSION

In recommender systems, a traditional way to infer the user's opinions to the items is to collect the user rating or clicking feedback. However, there may be many reasons why a user like or dislike an item, and the user rating or clicking feedback is not able to precisely convey these information to the system. If a user is not satisfied with the current recommendations and can not find an easy way to express the preference, the system is not able to learn to improve the recommendations. To resolve this issue, we design a

more effective user feedback to the recommended items, to enable the user to easily express the preference. Specifically, we propose a visual dialog augmented interactive recommender system, where the user is able to provide natural language feedback to the visual appearances of some specific items in a list of recommendations. To model the user's behavior of viewing, commenting and clicking on a list of items, we propose an extension of the cascade model. To learn this model interactively in an online setting, we further propose a novel variant of cascading bandits. With experiments, we validate that our recommender can effectively and efficiently understand the user preferences, and provide satisfactory recommendations with much fewer user interactions, compared to the traditional interactive recommender systems.

## REFERENCES


[1] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *ICCV*, pages 2425–2433, 2015.

[2] Daniel Paul Barrett, Andrei Barbu, N Siddharth, and Jeffrey Mark Siskind. Saying what you're looking for: Linguistics meets video search. *TPAMI*, 38(10):2069–2081, 2016.

[3] Robert M Bell and Yehuda Koren. Lessons from the netflix prize challenge. *ACM SIGKDD Explorations Newsletter*, 9(2):75–79, 2007.

[4] Tamara L Berg, Alexander C Berg, and Jonathan Shih. Automatic attribute discovery and characterization from noisy web data. In *ECCV*, pages 663–676. Springer, 2010.

[5] Christopher M Bishop. Pattern recognition and machine learning (information science and statistics). 2006.

[6] Derek G Bridge. Towards conversational recommender systems: A dialogue grammar approach. In *ECCBR Workshops*, pages 9–22, 2002.

[7] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pages 2249–2257, 2011.

[8] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. Towards conversational recommender systems. In *KDD*, pages 815–824. ACM, 2016.

[9] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 7(3):1–115, 2015.

[10] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, Stefan Lee, José Moura, Devi Parikh, and Dhruv Batra. Visual dialog. *TPAMI*, 2018.

[11] Abhishek Das, Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. Learning cooperative visual dialog agents with deep reinforcement learning. In *ICCV*, pages 2970–2979. IEEE, 2017.

[12] Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. Guesswhat?! visual object discovery through multi-modal dialogue. In *CVPR*, pages 5503–5512, 2017.

[13] Gideon Dror, Noam Koenigstein, Yehuda Koren, and Markus Weimer. The yahoo! music dataset and kdd-cup'11. In *Proceedings of the 2011 International Conference on KDD Cup 2011-Volume 18*, pages 3–18. JMLR. org, 2011.

[14] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, et al. Query by image and video content: The qbic system. *computer*, 28(9):23–32, 1995.

[15] Claudio Greco, Alessandro Suglia, Pierpaolo Basile, and Giovanni Semeraro. Converse-et-impera: Exploiting deep learning and hierarchical reinforcement learning for conversational recommender systems. In *Conference of the Italian Association for Artificial Intelligence*, pages 372–386. Springer, 2017.

[16] Zhiwei Guan and Edward Cutrell. An eye tracking study of the effect of target rank on web search. In *CHI*, pages 417–420. ACM, 2007.

[17] Xiaoxiao Guo, Hui Wu, Yu Cheng, Steven Rennie, Gerald Tesauro, and Rogerio Feris. Dialog-based interactive image retrieval. In *NIPS*, pages 676–686. 2018.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[19] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In *CVPR*, pages 4555–4564, 2016.

[20] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR*, pages 154–161. ACM New York, 2005.

[21] Sumeet Katariya, Branislav Kveton, Csaba Szepesvari, and Zheng Wen. Dcm bandits: Learning to rank with multiple clicks. In *ICML*, pages 1215–1224, 2016.

[22] Yoon Kim. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751, 2014.

[23] Adriana Kovashka and Kristen Grauman. Attribute pivots for guiding relevance feedback in image search. In *ICCV*, pages 297–304, 2013.

[24] Adriana Kovashka and Kristen Grauman. Attributes for image retrieval. In *Visual Attributes*, pages 89–117. Springer, 2017.

[25] Adriana Kovashka, Devi Parikh, and Kristen Grauman. Whittlesearch: Image search with relative attribute feedback. In *CVPR*, pages 2973–2980. IEEE, 2012.

[26] Branislav Kveton, Csaba Szepesvari, Zheng Wen, and Azin Ashkan. Cascading bandits: Learning to rank in the cascade model. In *ICML*, pages 767–776, 2015.

[27] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *WWW*, pages 661–670. ACM, 2010.

[28] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. Towards deep conversational recommendations. In *Advances in Neural Information Processing Systems*, pages 9725–9735, 2018.

[29] Shuang Li, Tong Xiao, Hongsheng Li, Bolei Zhou, Dayu Yue, and Xiaogang Wang. Person search with natural language description. In *CVPR*, pages 5187–5196. IEEE, 2017.

[30] Bing Liu, Tong Yu, Ian Lane, and Ole Mengshoel. Customized nonlinear bandits for online response selection in neural conversation models. In *AAAI*, 2018.

[31] Bo Liu, Ying Wei, Yu Zhang, Zhixian Yan, and Qiang Yang. Transferable contextual bandit for cross-domain recommendation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[32] Lori Lorigo, Maya Haridasan, Hrönn Brynjarsdóttir, Ling Xia, Thorsten Joachims, Geri Gay, Laura Granka, Fabio Pellacini, and Bing Pan. Eye tracking and online search: Lessons learned and challenges ahead. *Journal of the American Society for Information Science and Technology*, 59(7):1041–1052, 2008.

[33] Nader Mirzadeh, Francesco Ricci, and Mukesh Bansal. Feature selection methods for conversational recommender systems. In *IEEE International Conference on e-Technology, e-Commerce and e-Service*, pages 772–777. IEEE, 2005.

[34] Devi Parikh and Kristen Grauman. Relative attributes. In *ICCV*, pages 503–510. IEEE, 2011.

[35] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *CVPR*, pages 7008–7024, 2017.

[36] Carlos Riquelme, George Tucker, and Jasper Snoek. Deep bayesian bandits showdown. In *ICLR*, 2018.

[37] Yong Rui, Thomas S Huang, and Shih-Fu Chang. Image retrieval: Current techniques, promising directions, and open issues. *Journal of visual communication and image representation*, 10(1):39–62, 1999.

[38] Yong Rui, Thomas S Huang, Michael Ortega, and Sharad Mehrotra. Relevance feedback: a power tool for interactive content-based image retrieval. *IEEE Transactions on circuits and systems for video technology*, 8(5):644–655, 1998.

[39] Paul Hongsuck Seo, Andreas Lehrmann, Bohyung Han, and Leonid Sigal. Visual reference resolution using attention memory for visual dialog. In *NIPS*, pages 3719–3729, 2017.

[40] Yilin Shen, Yue Deng, Avik Ray, and Hongxia Jin. Interactive recommendation via deep neural memory augmented contextual bandits. In *RecSys*, pages 122–130. ACM, 2018.

[41] Florian Strub, Harm De Vries, Jeremie Mary, Bilal Piot, Aaron Courvile, and Olivier Pietquin. End-to-end optimization of goal-driven and visually grounded dialogue systems. In *IJCAI*, pages 2765–2771. AAAI Press, 2017.

[42] Yueming Sun and Yi Zhang. Conversational recommender system. In *SIGIR*, SIGIR '18, pages 235–244, 2018.

[43] Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Movieqa: Understanding stories in movies through question-answering. In *CVPR*, pages 4631–4640, 2016.

[44] Stefanie Tellex and Deb Roy. Towards surveillance video search by natural language query. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, page 38. ACM, 2009.

[45] Bart Thomee and Michael S Lew. Interactive search in image retrieval: a survey. *International Journal of Multimedia Information Retrieval*, 1(2):71–86, 2012.

[46] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015.

[47] Hong Wu, Hanqing Lu, and Songde Ma. Willhunter: interactive image retrieval with multilevel relevance. In *ICPR*, volume 2, pages 1009–1012. IEEE, 2004.

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

[49] Aron Yu and Kristen Grauman. Fine-grained comparisons with attributes. In *Visual Attributes*, pages 119–154. Springer, 2017.

[50] Ruiyi Zhang, Zheng Wen, Changyou Chen, Chen Fang, Tong Yu, and Lawrence Carin. Scalable thompson sampling via optimal transport. In *AISTATS*, 2019.

[51] Shi Zong, Hao Ni, Kenny Sung, Nan Rosemary Ke, Zheng Wen, and Branislav Kveton. Cascading bandits for large-scale recommendation problems. In *UAI*, pages 835–844. AUAI Press, 2016.