Offline Interactive Recommendation with Natural Language Feedback

Ruiyi Zhang[1] **Tong Yu**[2] **Yilin Shen**[2] **Hongxia Jin**[2]

[1] Duke University
[2] Samsung Research America

Abstract

Interactive recommendation with natural-language feedback can provide richer user feedback and has demonstrated advantages over traditional recommender systems. However, the classical online paradigm involves iteratively collecting experience via interaction with users, which is expensive and risky. We consider an offline interactive recommendation to exploit *arbitrary* experience collected by *multiple unknown* policies. A direct application of policy learning with such fixed experience suffers from the distribution shift. To tackle this issue, we develop an off-policy correction framework to make offline training possible. Specifically, we leverage the adversarial training to perform off-policy evaluation, which enables learning effective policies from fixed datasets without further interaction. Empirical results on real-world datasets demonstrate the effectiveness of our proposed offline training framework.

Introduction

Traditional interactive recommender systems continuously collect user preferences over time. Most existing interactive recommender systems are designed based on simple user feedback, such as clicking data or updated ratings (Chapelle and Li 2011; Kveton et al. 2015; Li et al. 2010). However, this type of feedback contains little information to reflect complex user attitude towards various aspects of an item. Especially in certain scenarios of personal assistants, such as Amazon Echo show and Google home hub, visual items are recommended (Guo et al. 2018, 2019). The click or numeric rating is neither sufficient to express a preference nor desirable when focusing on other aspects for users. Interactive recommendation with natural-language feedback provides richer information, where a user can describe features of desired items that are lacking in the current recommended ones. The recommender can then incorporate feedback and subsequently recommend more suitable items. This type of recommendation is referred to as *text-based interactive recommendation*.

The classical online paradigm involves iteratively collecting experience via interacting with users, which is not realistic in the real-world. Offline interaction recommendation is a promising setting in, for example, safety critical or production systems, where learned policies should not be applied on the real system until their performance and safety is verified. Further, we consider the scenario of personal assistants, where users usually interact with their personal assistants on devices. These devices can collect interaction data but can only perform light adaptation for personalized recommendation. This offline data can be shared by the users if they agree for service improvement, but some personalized data cannot be shared. Thus, the personalized policies on-device are usually unknown when training a policy in an offline manner on the server. One can apply imitation learning on successful interaction data, but the policy can be sub-optimal because the recommender cannot exploit failure experience. Directly learning a recommender policy via off-policy reinforcement learning will suffer from distribution shift as the experience are usually collected by multiple unknown policies on devices. Previous offline training usually considers importance sampling for distribution correction (Chen et al. 2019), but it assumes all the offline data is collected by a single known policy.

To overcome these issues, we consider behaviour-agnostic offline reinforcement learning, where a distribution correction is efficiently estimated in an adversarial manner. This simple adversarial correction estimator is derived from the property of stationary distributions, and compatible with offline data collected by multiple unknown policies. Empirical results on real-world datasets show that the proposed framework can accurately estimate the distribution correction compared with some standard baselines. Further, the offline training scheme shows superior performance compared with direct off-policy training in an offline interactive recommendation.

Background

Reinforcement Learning

Reinforcement learning aims to learn an optimal policy for an agent interacting with an unknown (and often highly complex) environment. A policy is modeled as a conditional distribution $\pi(\boldsymbol{a}|\boldsymbol{s})$, specifying the probability of choosing action $\boldsymbol{a} \in \mathcal{A}$ when in state $\boldsymbol{s} \in \mathcal{S}$. Formally, an RL problem is characterized by a Markov decision process (MDP) (Puterman 2014), $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, T, R, \mu_0 \rangle$. In this work, we consider interactive recommendation as finite-horizon environments

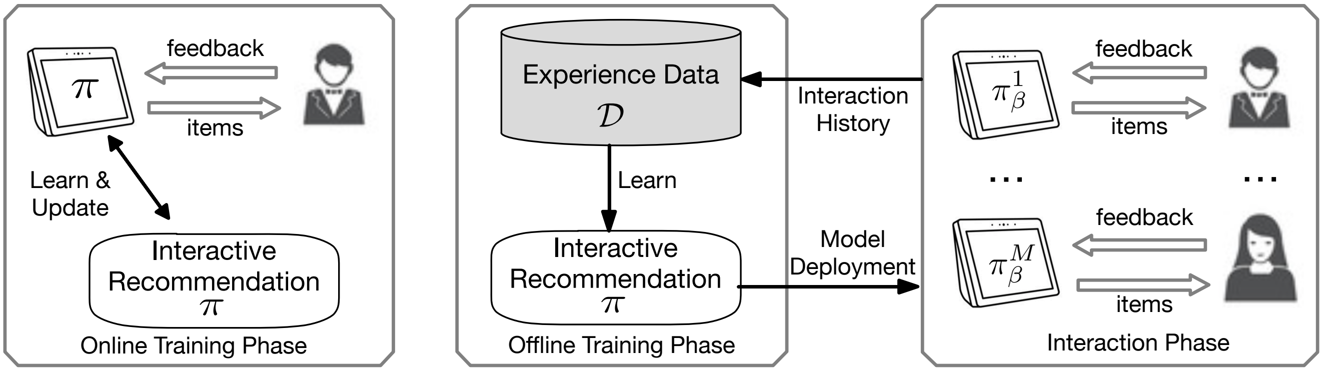

Figure 1: Comparison between the online and offline interactive recommendation model.

with the discounted reward criterion. If the agent chooses action $\boldsymbol{a} \in \mathcal{A}$ at state $\boldsymbol{s} \in \mathcal{S}$, then the agent will receive an immediate reward $r(\boldsymbol{s}, \boldsymbol{a})$, and the state will transit to $\boldsymbol{s}' \in \mathcal{S}$ with probability $T(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})$. The expected total reward of a policy π is defined as (Sutton and Barto 2018):

$$J(\pi) = \mathbb{E}_{\tau \sim p_\pi}\left[\sum_{t=0}^{H} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{1}$$

$$= \mathbb{E}_{\tau \sim p_\pi} Q(\boldsymbol{s}_t, \boldsymbol{a}_t) \ \ , \ \text{ where}$$

$$p_\pi(\boldsymbol{s}, \boldsymbol{a}) = \begin{cases} \frac{1}{H+1}\sum_{t=0}^{H} d_t^\pi(\boldsymbol{s}_t, \boldsymbol{a}_t), & \text{if } \gamma = 1, \\ (1-\gamma)\sum_{t=0}^{H} \gamma d_t^\pi(\boldsymbol{s}_t, \boldsymbol{a}_t), & \text{if } \gamma < 1. \end{cases}, \tag{2}$$

where $\tau = (\boldsymbol{s}_0, \boldsymbol{a}_0, \ldots, \boldsymbol{s}_H, \boldsymbol{a}_H)$ is a sequence of states and actions (*i.e.*, the trajectory), and the trajectory distribution is defined as $d_t^\pi(\boldsymbol{s}, \boldsymbol{a}) := \mathbb{P}\left\{\boldsymbol{s}_t = s, \boldsymbol{a}_t = a | \boldsymbol{s}_0 \sim \mu_0, \boldsymbol{a}_i \sim \pi(\cdot|\boldsymbol{s}_i), s_{i+1} \sim T(\cdot|\boldsymbol{s}_i, \boldsymbol{a}_i)\right\}$. Given a dataset of trajectories $\mathcal{D}$ collected under a behavior policy π, Q-learning maintain a parametric Q-function $Q_\psi(s, a)$. Q-learning methods with greedy action selection train the Q-function by iteratively applying the Bellman operator $\mathcal{T}^* Q(\boldsymbol{s}, \boldsymbol{a}) = r(\boldsymbol{s}, \boldsymbol{a}) + \gamma \mathbb{E}_{\boldsymbol{s}' \sim T(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})}[\max_{\boldsymbol{a}'} Q(\boldsymbol{s}', \boldsymbol{a}')]$. Actor-critic methods using a learned policy, $\pi_\phi(a|s)$ instead of the greedy one. Accordingly, the Q-value is estimated uses an empirical Bellman operator based on a single action given by $\pi_\phi(a|s)$:

$$\hat{Q}^{k+1} \leftarrow \arg\min_{Q} \mathbb{E}_{\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}' \sim \mathcal{D}}[((r(\boldsymbol{s}, \boldsymbol{a}) + \gamma \mathbb{E}_{\boldsymbol{a}' \sim \pi_\phi(\boldsymbol{a}'|\boldsymbol{s}')}[\hat{Q}^k(\boldsymbol{s}', \boldsymbol{a}')]) - Q(\boldsymbol{s}, \boldsymbol{a}))^2] \tag{3}$$

The goal of an agent is to learn an optimal policy that maximizes $J(\pi)$, *i.e.*, maximize the expected Q-value.

Interactive Recommendation as Reinforcement Learning

We employ an RL-based formulation for interactive recommendation, utilizing user feedback in natural language. Denote $\boldsymbol{s}_t \in \mathcal{S}$ as the state of the recommendation environment at time t and $\boldsymbol{a}_t \in \mathcal{A}$ as the recommender-defined items from the candidate items set $\mathcal{A}$. In the context of a recommendation system, as discussed further below, the state $\boldsymbol{s}_t$ corresponds to the state of sequential recommender, implemented via a LSTM (Hochreiter and Schmidhuber 1997) state tracker. At time t, the system recommends item $\boldsymbol{a}_t$ based on the current state $\boldsymbol{s}_t$ at time t. After viewing item $\boldsymbol{a}_t$, a user may comment on the recommendation in natural language (a sequence of natural-language text) $\boldsymbol{x}_t$, as feedback. The recommender then receives a reward r_t and perceives the new state $\boldsymbol{s}_{t+1}$.

Accordingly, we can model the recommendation-feedback loop as a MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, T, R \rangle$, where $T: \ \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is the environment dynamic of recommendation and $R: \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function used to evaluate recommended items. The recommender seeks to learn a policy, $\pi(\boldsymbol{a}|\boldsymbol{s})$, that corresponds to the distribution of items conditioned on the current state of the recommender. The recommender is represented as an optimal policy that maximizes the expected reward as $J(\pi) = \sum_t \mathbb{E}_{\tau \sim p_\pi}[r(\boldsymbol{s}_t, \boldsymbol{a}_t)]$.

Offline Reinforcement Learning

The offline reinforcement learning considers optimizing Equation (8) from a fixed dataset $\mathcal{D}$ (similar to the training set in supervised learning). In more details, the agent cannot interact with the environment and collect more experience, and need to understand the underlying MDP $\mathcal{M}$ from a fixed dataset and learn a policy that can attain higher rewards when interacting with the MDP (when testing). We denote the behaviour policy as π_β and the fixed dataset is collected by π_β. Importance sampling (Precup, Sutton, and Dasgupta 2001) has been widely investigated in offline recommendation (Chen et al. 2019), and one can show that:

$$J(\pi) = \mathbb{E}_{\tau \sim p_{\pi_\beta}}\left[\sum_{t=0}^{H} \frac{\pi(\boldsymbol{a}|\boldsymbol{s})}{\pi_\beta(\boldsymbol{a}|\boldsymbol{s})} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right], \tag{4}$$

where p_{π_β} is trajectories collected by the behaviour policy π_β. However, importance sampling requires the knowledge of the behaviour policy and the offline data should be collected by a *single* policy. In many realistic settings, only a fixed dataset, which is collected by multiple unknown policies, is given. Even if one can assume the behaviour policy can be estimated from data, it is known that that straightforward importance sampling estimators suffer a exponential variance (Chen et al. 2019), known as the "curse of horizon" (Liu et al. 2018). Learning from arbitrary experience of multiple policies is a natural but challenging problem.

The Proposed Framework

We consider offline interactive recommendation of visual items (Guo et al. 2018, 2019) with natural language feedback. In this scenario, a user views a recommended item and gives feedback in natural language, describing the desired aspects that the current recommended item lacks. The system then incorporates the user feedback and recommends (ideally) more suitable items, until a desired item is found. As shown in Figure 5, offline interactive recommendation model cannot directly interact with users but learn from experience data collected by multiple unknown policies; while classical interactive recommendation iteratively improves via interacting with users. Specifically, we assume there are many personalized devices (*e.g.,* Amazon Echo show and Google Home hub) collecting experience while interacting with users. The experience data are uploaded to train an interactive recommendation policy in an offline manner. It is allowed to upload some experience for service improvement but some local personalized information cannot be shared (usually caused by privacy issue), thus the behaviour policies are usually unknown, and different.

Policy Learning from Arbitrary Experience

It is difficult to learn from a fixed experience dataset collected by multiple unknown policies. The usually adopted importance-sampling based methods have unrealistic assumptions on the fixed data as discussed later. Directly performing off-policy learning on this fixed dataset $\mathcal{D}$ will suffer from distribution shift, rendering sub-optimal policies. To alleviate this issue, we consider the offline policy learning via a density regularization (Kumar et al. 2019; Wu, Tucker, and Nachum 2019):

$$J(\pi) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a})\sim p_\pi}[r(\boldsymbol{s},\boldsymbol{a})] - \alpha D_\phi(p_\pi \| p)\,, \tag{5}$$

with $\alpha > 0$ and D_ϕ denoting the f-divergence induced by a convex function ϕ:

$$D_\phi(p_\pi \| p) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a})\sim p_\pi}\left[\phi\left(\boldsymbol{w}(\boldsymbol{s},\boldsymbol{a})\right)\right] \tag{6}$$

with $\boldsymbol{w}(\boldsymbol{s},\boldsymbol{a}) := \frac{p_\pi(\boldsymbol{s},\boldsymbol{a})}{p(\boldsymbol{s},\boldsymbol{a})}$, p is the distribution of fixed dataset $\mathcal{D}$ and p_π is the state visitation distribution induced by π. The regularization $D_\phi(p_\pi \| p)$ encourages conservative behaviour, compelling the state-action occupancy of π to remain close to the off-policy distribution, which can improve generalization. Different divergences can be obtained by choosing appropriate f. Note the original objective in Equation 9 not only requires on-policy samples from p_π, but also involves the f-divergence term, which is difficult to compute. To bypass these difficulties, we first eliminate the on-policy sample requirement by following reformulation:

$$J(\pi) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a})\sim p}[\boldsymbol{w}(\boldsymbol{s},\boldsymbol{a}) \cdot r(\boldsymbol{s},\boldsymbol{a}) - \alpha\phi(\boldsymbol{w}(\boldsymbol{s},\boldsymbol{a}))]\,, \tag{7}$$

Equation 7 is very similar to standard offline policy learning, except that the policies are unknown in our realistic settings. The ratio $\boldsymbol{w}(\boldsymbol{s},\boldsymbol{a})$ can be estimated via importance sampling (IS) if offline data are collected by a single known policy. However, we do not make this assumption as previous works do, because we assume the offline data are collected by multiple unknown devices. Following Nachum et al. (2019); Zhang et al. (2020), we can solve the above offline policy optimization problem.

Distribution correction estimation (DICE) (Nachum et al. 2019; Liu et al. 2018; Zhang et al. 2020) is usually challenging. DICE uses marginalized importance sampling by directly estimating the state(-action)-distribution importance ratios, showing less variance than classical importance sampling and enabling learning from arbitrary experience. However, there are constraints on the output of the neural correction estimator, rendering its difficulty of model optimization. Alternatively, we can perform offline consecutive policy evaluation (Kumar et al. 2020):

$$\begin{aligned}\hat{Q}^{k+1} \leftarrow \arg\min_Q \alpha \cdot [&\mathbb{E}_{\boldsymbol{s}\sim\mathcal{D},\boldsymbol{a}\sim\pi}\, Q(\boldsymbol{s},\boldsymbol{a}) \\ &- \mathbb{E}_{\boldsymbol{s},\boldsymbol{a}\sim\mathcal{D}}\, Q(\boldsymbol{s},\boldsymbol{a})] \\ &+ \frac{1}{2}\mathbb{E}_{\boldsymbol{s},\boldsymbol{a},\boldsymbol{a}'\sim\mathcal{D}}\left[\left(\hat{Q}^k(\boldsymbol{s},\boldsymbol{a}) - \hat{\mathcal{T}}^\pi Q(\boldsymbol{s},\boldsymbol{a})\right)^2\right],\end{aligned} \tag{8}$$

where α is the trade-off factor, and π is the target policy we aim to evaluate, $\hat{\mathcal{T}}^\pi$ is the empirical Bellman operator which backs up a single sample, $\hat{Q}^k$ is the estimated Q-value function. Standard Q-learning only queries the Q-function at unseen actions, but in offline RL, it queries the Q-value at unobserved states. The second term is the standard Bellman update as defined in Equation 8. The first term restrict the target policy π to match the state-marginal in the dataset $\mathcal{D}$, such that $p_\pi(\boldsymbol{s},\boldsymbol{a}) = \boldsymbol{w}(\boldsymbol{s},\boldsymbol{a})p(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})$. Intuitively, $\boldsymbol{w}(\boldsymbol{s},\boldsymbol{a})$ is the distribution correction to eliminate the bias for offline policy learning, and the policy distribution p_π should be close to the data distribution p to avoid the potential penalty in the conservative Q-function learning.

With the estimated $\hat{Q}(\boldsymbol{s}_t,\boldsymbol{a}_t)$, *i.e.*, target policy π can be evaluated based on a fixed dataset $\mathcal{D}$, one can perform policy improvement, which is very similar to standard policy learning. Following Kumar et al. (2020), we add an entropy regularization $\mathcal{H}(\pi)$ in policy improvement, and the optimal policy is $\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp(\hat{Q}^(\boldsymbol{s},\boldsymbol{a}))$, which leads to soft actor-critic (SAC) (Haarnoja et al. 2018) updates:

$$\begin{aligned}\min_Q \max_\pi \alpha \cdot [&\mathbb{E}_{\boldsymbol{s}\sim\mathcal{D},\boldsymbol{a}\sim\pi}\, Q(\boldsymbol{s},\boldsymbol{a}) \\ &- \mathbb{E}_{\boldsymbol{s},\boldsymbol{a}\sim\mathcal{D}}\, Q(\boldsymbol{s},\boldsymbol{a})] + \mathcal{H}(\pi) \\ &+ \frac{1}{2}\mathbb{E}_{\boldsymbol{s},\boldsymbol{a},\boldsymbol{a}'\sim\mathcal{D}}\left[\left(\hat{Q}^k(\boldsymbol{s},\boldsymbol{a}) - \hat{\mathcal{T}}^\pi Q(\boldsymbol{s},\boldsymbol{a})\right)^2\right],\end{aligned} \tag{9}$$

Note the proposed framework is general and can adopt different offline reinforcement learning algorithms such as Consecutive Q-Learning (CQL) (Kumar et al. 2020), batch constraint Q-learning (BCQ) (Fujimoto, Meger, and Precup 2019) and AlgaeDICE (Nachum et al. 2019).

Model Architecture

We next discuss details on model design in an offline text-based recommender system. Different from online interactive recommendation, where the recommender improves via interacting with users, offline interactive recommendation can only access to a fixed experience dataset $\mathcal{D}$ and learn from it. As illustrated in Figure 2, the feature extractor takes natural

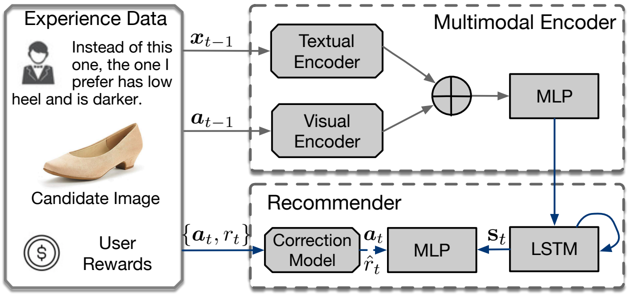

Figure 2: Offline training scheme for interactive recommendation.

language feedback $\boldsymbol{x}_{t-1}$ and its corresponding item $\boldsymbol{a}_{t-1}$ to update the LSTM state tracker. Then the recommender observes recommended item $\boldsymbol{a}_t$ and its corrected reward $\hat{r}_t$ in $\mathcal{D}$ to perform offline training.

Multimodal Encoder Before the system making recommendations, the multimodal encoder understands the current context, based on the textual and visual embedding encoded from the raw data of the user query and candidate image in the experience data. To understand the user query $\boldsymbol{x}_t$, a textual encoder is applied to extract the textual embedding $\boldsymbol{c}_t^{txt}$ by word embedding, a GRU, and a linear layer. To understand the visual content of a candidate image $\boldsymbol{a}_t$, a visual encoder extracts the visual embedding $\boldsymbol{c}_t^{vis} = \texttt{ResNet}(\boldsymbol{a}_t)$ by a convolutional neural network. In practice, we use a pre-trained residual neural network (*i.e.*, ResNet50) (He et al. 2016) as the visual encoder. The textual embedding and visual embedding are then concatenated as the input to a linear layer $g(.)$ for the multimodal fusion. This linear layer's output $\boldsymbol{c}_t = g([\boldsymbol{c}_t^{vis}, \boldsymbol{c}_t^{txt}])$ serves as the input of the recommender below.

Recommender To incorporate the temporal information in a user session, the recommender extracts the state by a LSTM. At time t, the input from the multimodal encoder is $\boldsymbol{c}_t$. The state is then represented by $\boldsymbol{s}_t = f(\boldsymbol{c}_t, \boldsymbol{s}_{t-1})$, where $f(.)$ is a LSTM (Hochreiter and Schmidhuber 1997). We follow the recommendation setting in (Christakopoulou, Radlinski, and Hofmann 2016; Sun and Zhang 2018; Lei et al. 2018), where the items are associated with a number of attributes. In each session, the user is to assumed to find items with specific attribute values. Accordingly, the learning agent with policy π_ϕ is designed to take action in a multi-discrete space (Hill et al. 2018; Dhariwal et al. 2017). Each action value corresponds to an attribute value sampled from a categorical distribution over the space. With the state as the input, we approximate the probability of taking an action by a fully connected neural network with a softmax activation function. At each time t, by observing $\boldsymbol{s}_t$, the learning agent takes actions and accordingly the system recommend the items with the corresponding attribute values.

Algorithm 1 Offline Interactive Recommendation

```
Input: Approximate Q-value function $Q_\psi$, or distribution correction: $\boldsymbol{w}_\omega$, learning rates $\eta$, and collected offline dataset $\mathcal{D}$.
Initialize recommender $\pi_\phi$, and perform pretraining.
repeat
    Sample a batch of experience from $\mathcal{D}$
    Estimate the distribution correction $\boldsymbol{w}$ or update the conservative Q-value function.    [Policy Evaluation]
    Update recommender policy $\pi_\phi$ via soft actor-critic with (9).    [Policy Improvement]
until Model converges
return recommender (policy) parameters $\phi$.
```

Related Work

Offline Reinforcement Learning Off-policy policy learning with importance sampling (IS) has been explored in the contextual bandits (Strehl et al. 2010), and episodic RL settings (Precup, Sutton, and Dasgupta 2001). In recommendation, importance sampling is usually used to correct distribution shift (Chen et al. 2019; Ma et al. 2020). Unfortunately, IS-based methods suffer from exponential variance in long-horizon problems, known as the "curse of horizon" (Liu et al. 2018). Recently developed off-policy learning considers behaviour regularization, *i.e.*, the policy should be close to the behaviour policy. By rewriting the accumulated reward as an expectation w.r.t. a stationary distribution, (Liu et al. 2018; Gelada and Bellemare 2019) recast OPE as estimating a correction ratio function, which significantly alleviates variance. However, these methods still require the off-policy data to be collected by a *single and known* behaviour policy, which restricts their practical applicability. However, DualDICE (Nachum et al. 2019) was developed for discounted problems and its results become unstable when the discount factor approaches. GenDICE (Zhang et al. 2020) can cope with the more challenging problem of undiscounted reward estimation in the general behaviour-agnostic setting.

Conversational Recommender System With the advance of natural language understanding and dialog systems, the conversations between users and systems have been leveraged to improve the traditional recommender systems (Jannach et al. 2020). Aliannejadi, *et al.* (Aliannejadi et al. 2019) proposes a neural question selection model for the task of asking clarifying questions in open-domain information-seeking conversations. In a two stage solution by (Christakopoulou et al. 2018), a RNN-based model is proposed for generating interesting topics to the user, and a state-of-the-art RNN-based video recommender is extended to incorporate the user's selected topic. By integrating and revising several conversational recommenders, Lei, *et al.* (Lei et al. 2018) proposes a three-stage solution, to better converse with users and achieve accurate recommendations. The conversational recommendation task is also formulated as a reinforcement learning problem in various previous works (Sun and Zhang 2018; Greco et al. 2017; Zhang et al. 2019), by optimizing various reward functions. We follow the setting in (Christakopoulou,

Round	User Feedback	Round	User Feedback
1	I want boots	1	The shoes I want has flat
2	Please provide some shoes for women	2	Show me more shoes with men
3	I am looking for shoes for women	3	Please provide some shoes with lace up
4	Please provide some shoes with round toe	4	I am looking for shoes with flat
5	I prefer shoes for women	5	Do you have sneakers and athletic shoes
6	-	6	Show me more shoes with men
1	I am looking for shoes with men.	1	Do you have shoes with ankle.
2	Do you have shoes with medallion.	2	I want pull-on.
3	Show me more shoes with flat.	3	Do you have shoes with pull-on.
4	I am looking for shoes with flat.	4	I prefer shoes with men.
5	I do not need the shoes without flat.	5	Do you have flat.
6	-	6	Show me more shoes with flat.
1	The shoes I want has Slip-On.	1	Please provide some shoes with Elastic Gore.
2	Do you have shoes with Women.	2	Do you have shoes with Flat.
3	Do you have shoes with Capped Toe.	3	I am looking for shoes with Sandals.
4	I prefer Flats.	4	Show me more shoes with Sandals.
5	-	5	I want 1in - 1 3/4in.
6	-	6	-

Table 1: Examples of the generated feedback by the user simulator.

Radlinski, and Hofmann 2016; Sun and Zhang 2018; Lei et al. 2018), where the recommended items are associated with a number of attributes.

Multimodal Retrieval and Recommender System To improve the interactive recommendations, data from multiple modalities have been leveraged to understand the user preference more accurately (Thomee and Lew 2012). Depending on the feedback format, previous works can be categorized into relevance feedback (Rui et al. 1998; Wu, Lu, and Ma 2004) and relative attributes feedback (Kovashka, Parikh, and Grauman 2012; Parikh and Grauman 2011; Yu and Grauman 2017; Zhu et al. 2019). Specifically, user's natural language feedback to visual content of items has been studied to achieve more efficient user interactions (Guo et al. 2019). Guo, *et al.* (Guo et al. 2018) proposes an end-to-end system by reinforcement learning, to enable the multi-turn multimodal interactive retrieval. In the VAL framework (Chen, Gong, and Bazzani 2020), a composite transformer is proposed to selectively preserve and transform the visual features conditioned on language semantic. To retrieve complex scenes, the drill-down framework (Tan et al. 2019) is proposed to capture the fine-grained alignments between local region of images and multiple text queries.

Experimental Results

We conduct experiments to evaluate the proposed framework on two aspects: (*i*) how good is the distribution correction estimation when the fixed experience dataset is collected by multiple unknown policies. (*ii*) whether the offline interactive recommendation training can handle more challenging scenarios than previous methods. All experiments are conducted on a single Tesla V100 GPU.

Environment and Dataset We compare our method with various baseline approaches on UT-Zappos50K (Yu and Grauman 2014a,b). This dataset includes $50{,}025$ shoes. For each shoe, there is an image and some meta information (*e.g.*, the attribute values of the shoes). In the evaluation, we randomly select $40{,}020$ shoes to form a training set and the rest shoes to form a test set. In the training set, we assume shoes are well-labeled with accurate attribute value labels (*i.e.*, "seen"). In the test set, the shoes are assumed to be newly included to the database and have no attribute labels (*i.e.*, "unseen"). With the unseen data, we can evaluate the generalization ability of the models to the newly included shoes. There are rich attribute information in this dataset, and our evaluation focus on the attributes of shoes category, shoes subcategory, heel height, closure, gender and toe style.

In the reinforcement learning of the online recommender, the reward can be the visual similarity between the recommended item $\boldsymbol{a}_t$ and the target item $\boldsymbol{a}^*$. This similarity can be measured by either the visual attribute similarity or the image embedding similarity between the items (Guo et al. 2019). By considering both similarities, in practice we design and maximize the following reward $r_t = 1-(1-\lambda_{\text{att}})||\texttt{ResNet}(\boldsymbol{a}_t)-\texttt{ResNet}(\boldsymbol{a}^*)||_2 - \lambda_{\text{att}}||\texttt{AttrNet}(\boldsymbol{a}_t) - \texttt{AttrNet}(\boldsymbol{a}^*)||_0$, where $||\cdot||_2$ denotes the $\mathcal{L}_2$ norm, $||\cdot||_0$ denotes the $\mathcal{L}_0$ norm, and λ_{att} is set to 0.5, $\texttt{AttrNet}(\cdot)$ is a fully connected network, which predicts attribute values given an image. We set the maximum length of a user session as 50: if a user interacts with the system for more than 50 times and still can not find the target item, we terminate the system an give an extra penalty reward -3 to the learning agent.

Interaction with Users

In our offline recommender, the model training relies on the experience data collected from an online recommender. Thus, we need to train and evaluate the online recommender as the first step. However, traditionally, it is difficult to train and evaluate an online model: the model is updated online and it is difficult to access labels (*i.e.*, the user feedback) to all possible items (Christakopoulou, Radlinski, and Hofmann

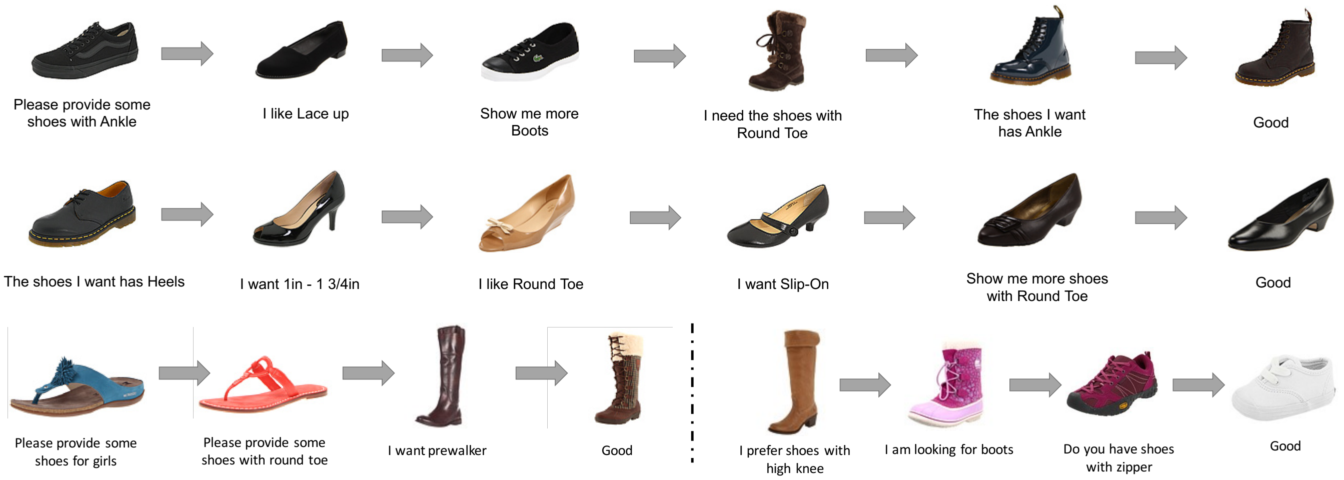
Please provide some shoes with Ankle
I like Lace up
Show me more Boots
I need the shoes with Round Toe
The shoes I want has Ankle
Good
The shoes I want has Heels
I want 1in - 1 3/4in
I like Round Toe
I want Slip-On
Show me more shoes with Round Toe
Good
Please provide some shoes for girls
Please provide some shoes with round toe
I want prewalker
Good
I prefer shoes with high knee
I am looking for boots
Do you have shoes with zipper
Good

Figure 3: Use cases of offline interactive recommendation with natural-language feedback. Each text below the image is user's comment for current recommendation.

2016; Guo et al. 2018; Zhang et al. 2019). Therefore, in practice, we train the online recommender on a *simulator* derived from the UT-Zappos50K dataset.

To derive this simulator, we train an utterance generation model. In our recommendation setting, the items are associated with a number of attributes (Christakopoulou, Radlinski, and Hofmann 2016; Sun and Zhang 2018; Lei et al. 2018), and the user goal is to find items with specific attribute values. Therefore, we assume the granularity of the user utterances are in the level of visual attributes. That is, the utterance generation model outputs a sentence describing the visual attribute difference between a candidate item and a target item. The inputs of the model are the differences on an attribute value between the two items. We prepare $10{,}000$ pairs of candidate items and target items, where each item is associated with an image with visual attributes. For each pair, we collect a real-world sentence about the target visual attributes, from a fixed set of attributes. To derive extra training data, the collected data is further augmented by template-based sentences. Specifically, we derive several sentence templates from the collected real-world sentences, and generate $20{,}000$ sentences by filling these templates with attribute values. With the attribute labels in the seen data, we pretrain the textual encoder under a cross-entropy loss.

The fact that the online model training needs a simulator also motivates our work of developing an offline reinforcement learning based recommender: instead of relying on a simulator, in offline reinforcement learning we only need a fixed data set collected by multiple unknown recommender. The latter option is more practical in the real-world setting, and has not been investigated yet. We show some examples of the generated feedback by the user simulator in Figure 3 and Table 1. To evaluating how the recommended item's visual attributes satisfy a user's previous feedback, our simulator only generates simple comments on the visual attribute difference between the candidate image and the desired image: we can calculate how many attributes violate the users' previous feedback based on the visual attribute ground truth available in UT-Zappos50K.

Implementation Details In the textual encoder, the dimension of the word embedding layer is 32, the dimension of the GRU is 128, and the dimension of the linear mapping layer is 32. The textual encoder is optimized by the Adam optimizer, with an initial learning rate of 0.001. In the recommender policy network, the dimension of the LSTM is 256. The policy network is optimized by the RMSProp optimizer. In RMSProp optimizer, the intial learning rate is $7e-4$, and the decay rate is set to 0.99. The discount factor of reinforcement learning is 0.99. The neural estimation corrector is a one-layer feedforward neural network with hidden size of 64. The discriminator is in the same size of the neural estimation corrector. We train the distribution correction estimation model using the Adam optimizer with batches of size 2048 and learning rate is 0.001.

Offline Interactive Recommendation

We further verify the proposed framework in offline interactive recommendation training, where the recommender is evaluated online after offline training. The performance is evaluated under the following evaluation metrics: (i) task success rate (SR@K), the rate of success after K interactions and (ii) number of interactions before success (NI) and number of violated attributes (NV). In each user session, we assume the user aims to find items with a set of desired attributes. Results are averaged over 100 sessions with standard error.

Setup In this experiment, we start from a random initialized offline recommender (trained for 5,000 steps) has low accuracy and learns on the experience data collected from online recommenders (Behaviour). We totally collected 40,000 user sessions in an iterative manner, and compare offline training with off-policy (w/o correction) and imitation learning (behaviour cloning with successful sessions).

Results We report the results in Table 2. It can be observed that by learning from these user sessions, both the offline and off-policy recommender improve upon initial policy. Since

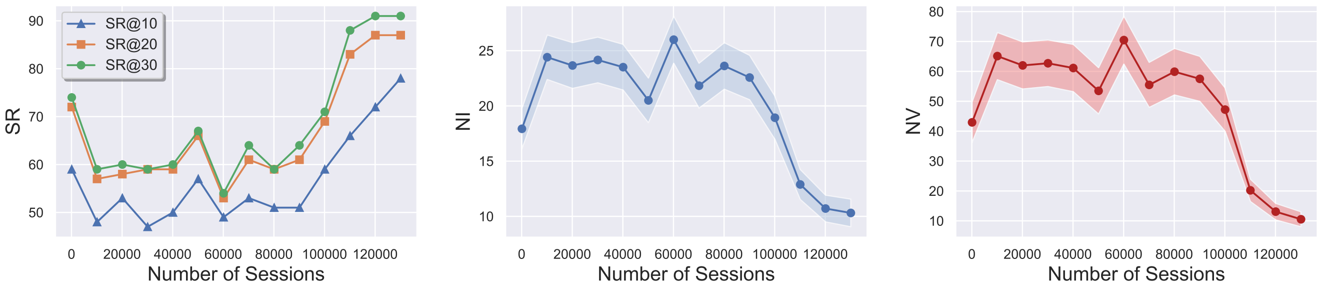

Figure 4: Training Curves of Iterative Offline Interactive Recommendation.

	SR@10 ↑	**SR@20** ↑	**SR@30** ↑	**NI** ↓	**NV** ↓
Behaviour	64%	83%	94%	12.72 ± 0.93	16.47 ± 2.75
On-policy	84%	90%	91%	9.91 ± 1.24	9.35 ± 1.87
Off-policy	59%	74%	78%	16.69 ± 1.72	39.92 ± 6.93
Imitation Learning	53%	81%	92%	12.91 ± 0.86	15.74 ± 2.94
Offline Learning	72%	87%	91%	10.71 ± 1.21	13.08 ± 2.78
Iterative Offline	78%	87%	91%	10.31 ± 1.25	10.53 ± 2.52

Table 2: Comparisons between different approaches. Behaviour is composed with multiple policies and its result is reported by averaging on all the collected trajectories.

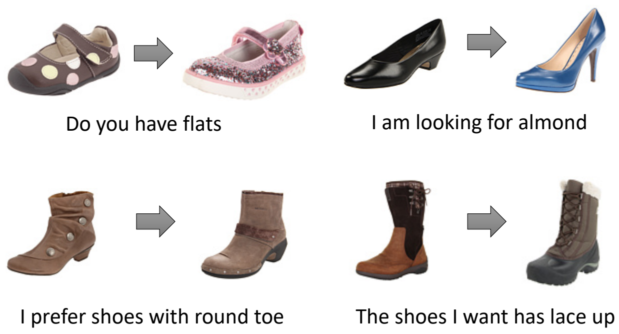

Figure 5: Examples of the generated feedback by the user simulator.

the off-policy suffers from distribution shifts, the improvement is very marginal. The on-policy recommender is the one trained via directly interacting with users, which should be the upper bound of the offline training in our experiments. It is also reasonable to see imitation learning shows a little worse results than the behaviour. Some use cases of offline interactive recommendation are also shown in Figure 3.

Iterative Offline Training

The results in previous Sections show that offline learning can improve the recommender with arbitrary experience collected by multiple unknown policies (recommenders). We find the quality of experience affects the performance, *i.e.* the recommender policy cannot achieve its best performance when the experience $\mathcal{D}$ is collected by policies with poor performance. Hence, we consider the iterative offline training: (*i*) collect offline experience $\mathcal{D}$ with multiple behaviour policies $\{\pi_i^\beta\}_{i=1}^M$. (*ii*) update the offline recommender π with $\mathcal{D}$. (*iii*) update the behaviour policies via model deployment and collect the new offline dataset $\mathcal{D}'$, and let $\mathcal{D} = \mathcal{D} \cup \mathcal{D}'$.

Model Deployment We consider the specific scenario where the recommender policy is trained in an offline manner and then deployed on the devices. The model on the device is usually smaller due to the limited storage and computation resources, but the model distillation is complicated and beyond the scope of this paper. Since the policies are different between devices, thus in the iterative training, the policies used to collect data are injected with Gaussian noise (Kusner, Hernández-Lobato, and Miguel 2016) when choosing actions.

Results We perform the iterative offline training with every 5000 user sessions as described above. Figure 4 shows the results on unseen test dataset and the best performance is reported in Table 2. With the iterative training scheme, the offline interactive recommendation can achieve similar performance as the classical on-policy learning.

Conclusions

Motivated by the on-device personal assistants in the real-world, and inspired by offline policy learning, we propose an offline interactive recommendation framework, where a neural network is parameterized and dynamically updated to tackle the distribution shift between the true policy and collected experience data. By applying this new framework to interactive recommendation with natural language feedback, we demonstrate the effectiveness of our proposed model in this challenging and realistic setting. The proposed framework is general, and can be extended to more complex real-world scenarios, such as Amazon Echo show and Google Home hub.

Ethical Impact

Interactive recommendation with natural-language feedback has demonstrated advantages with the rise of personal assistants, such as Amazon Echo, Google Home, etc. The classical online paradigm involves iteratively collecting experience via interaction with users. In the scenario of personal assistants, users usually interact with their personal assistants on devices. These devices can collect interaction data and can be shared by the users if they agree for service improvement, but some penalized data cannot be shared. Thus, the personalized policies on-device are usually unknown when training a policy in an offline manner on the server. Our proposed framework moves one-step forward in offline interactive recommendation, *i.e.* exploit arbitrary experience collected by *multiple unknown* policies, which widely exists in the personal assistant scenarios and is very challenging.

References

Aliannejadi, M.; Zamani, H.; Crestani, F.; and Croft, W. B. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *SIGIR*.

Chapelle, O.; and Li, L. 2011. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, 2249–2257.

Chen, M.; Beutel, A.; Covington, P.; Jain, S.; Belletti, F.; and Chi, E. H. 2019. Top-k off-policy correction for a REINFORCE recommender system. In *WSDM*.

Chen, Y.; Gong, S.; and Bazzani, L. 2020. Image Search with Text Feedback by Visiolinguistic Attention Learning. In *CVPR*.

Christakopoulou, K.; Beutel, A.; Li, R.; Jain, S.; and Chi, E. H. 2018. Q&R: A two-stage approach toward interactive recommendation. In *KDD*. ACM.

Christakopoulou, K.; Radlinski, F.; and Hofmann, K. 2016. Towards conversational recommender systems. In *KDD*, 815–824. ACM.

Dhariwal, P.; Hesse, C.; Klimov, O.; Nichol, A.; Plappert, M.; Radford, A.; Schulman, J.; Sidor, S.; Wu, Y.; and Zhokhov, P. 2017. OpenAI Baselines.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *ICML*.

Gelada, C.; and Bellemare, M. G. 2019. Off-policy deep reinforcement learning by bootstrapping the covariate shift. In *AAAI*.

Greco, C.; Suglia, A.; Basile, P.; and Semeraro, G. 2017. Converse-Et-Impera: Exploiting Deep Learning and Hierarchical Reinforcement Learning for Conversational Recommender Systems. In *Conference of the Italian Association for Artificial Intelligence*, 372–386. Springer.

Guo, X.; Wu, H.; Cheng, Y.; Rennie, S.; Tesauro, G.; and Feris, R. 2018. Dialog-based Interactive Image Retrieval. In *NIPS*, 676–686.

Guo, X.; Wu, H.; Gao, Y.; Rennie, S.; and Feris, R. 2019. The Fashion IQ Dataset: Retrieving Images by Combining Side Information and Relative Natural Language Feedback. *arXiv:1905.12794*.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hill, A.; Raffin, A.; Ernestus, M.; Gleave, A.; Kanervisto, A.; Traore, R.; Dhariwal, P.; Hesse, C.; Klimov, O.; Nichol, A.; Plappert, M.; Radford, A.; Schulman, J.; Sidor, S.; and Wu, Y. 2018. Stable Baselines.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Jannach, D.; Manzoor, A.; Cai, W.; and Chen, L. 2020. A Survey on Conversational Recommender Systems. *arXiv preprint arXiv:2004.00646*.

Kovashka, A.; Parikh, D.; and Grauman, K. 2012. Whittlesearch: Image search with relative attribute feedback. In *CVPR*.

Kumar, A.; Fu, J.; Soh, M.; Tucker, G.; and Levine, S. 2019. Stabilizing off-policy q-learning via bootstrapping error reduction. In *NeurIPS*.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative q-learning for offline reinforcement learning. *NeurIPS*.

Kusner, M. J.; Hernández-Lobato; and Miguel, J. 2016. Gans for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*.

Kveton, B.; Szepesvari, C.; Wen, Z.; and Ashkan, A. 2015. Cascading Bandits: Learning to Rank in the Cascade Model. In *ICML*, 767–776.

Lei, W.; He, X.; Miao, Y.; Wu, Q.; Hong, R.; Kan, M.-Y.; and Chua, T.-S. 2018. Estimation–Action–Reflection: Towards Deep Interaction Between Conversational and Recommender Systems. In *WSDM*.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *WWW*, 661–670. ACM.

Liu, Q.; Li, L.; Tang, Z.; and Zhou, D. 2018. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *NeurIPS*.

Ma, J.; Zhao, Z.; Yi, X.; Yang, J.; Chen, M.; Tang, J.; Hong, L.; and Chi, E. H. 2020. Off-policy Learning in Two-stage Recommender Systems. In *WWW*.

Nachum, O.; Chow, Y.; Dai, B.; and Li, L. 2019. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. In *NeurIPS*.

Parikh, D.; and Grauman, K. 2011. Relative attributes. In *ICCV*.

Precup, D.; Sutton, R. S.; and Dasgupta, S. 2001. Off-Policy Temporal-Difference Learning with Funtion Approximation. In *Proceedings of the 18th Conference on Machine Learning (ICML)*.

Puterman, M. L. 2014. *Markov Decision Processes.: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Rui, Y.; Huang, T. S.; Ortega, M.; and Mehrotra, S. 1998. Relevance feedback: a power tool for interactive content-based image retrieval. *IEEE Transactions on circuits and systems for video technology*.

Strehl, A.; Langford, J.; Li, L.; and Kakade, S. M. 2010. Learning from logged implicit exploration data. In *NeurIPS*.

Sun, Y.; and Zhang, Y. 2018. Conversational Recommender System. In *SIGIR*.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Tan, F.; Cascante-Bonilla, P.; Guo, X.; Wu, H.; Feng, S.; and Ordonez, V. 2019. Drill-down: Interactive Retrieval of Complex Scenes using Natural Language Queries. In *NeurIPS*.

Thomee, B.; and Lew, M. S. 2012. Interactive search in image retrieval: a survey. *International Journal of Multimedia Information Retrieval*.

Wu, H.; Lu, H.; and Ma, S. 2004. WillHunter: interactive image retrieval with multilevel relevance. In *ICPR*.

Wu, Y.; Tucker, G.; and Nachum, O. 2019. Behavior Regularized Offline Reinforcement Learning. *arXiv preprint arXiv:1911.11361*.

Yu, A.; and Grauman, K. 2014a. Fine-grained visual comparisons with local learning. In *CVPR*.

Yu, A.; and Grauman, K. 2014b. Semantic jitter: Dense supervision for visual comparisons via synthetic images. In *ICCV*.

Yu, A.; and Grauman, K. 2017. Fine-Grained Comparisons with Attributes. In *Visual Attributes*.

Zhang, R.; Dai, B.; Li, L.; and Schuurmans, D. 2020. Gendice: Generalized offline estimation of stationary values. In *ICLR*.

Zhang, R.; Yu, T.; Shen, Y.; Jin, H.; and Chen, C. 2019. Text-Based Interactive Recommendation via Constraint-Augmented Reinforcement Learning. In *Advances in Neural Information Processing Systems*, 15188–15198.

Zhu, Y.; Gong, Y.; Liu, Q.; Ma, Y.; Ou, W.; Zhu, J.; Wang, B.; Guan, Z.; and Cai, D. 2019. Query-based Interactive Recommendation by Meta-Path and Adapted Attention-GRU. In *CIKM*.