# Learning from Cross-Modal Behavior Dynamics with Graph-Regularized Neural Contextual Bandit

Xian Wu
University of Notre Dame
xwu9@nd.edu

Suleyman Cetintas
Yahoo Research
cetintas@verizonmedia.com

Deguang Kong
Google
doogkong@gmail.com

Miao Lu, Jian Yang
Yahoo Research
{mlu,jianyang}@verizonmedia.com

Nitesh V. Chawla
University of Notre Dame
nchawla@nd.edu

## ABSTRACT

Contextual multi-armed bandit algorithms have received significant attention in modeling users' preferences for online personalized recommender systems in a timely manner. While significant progress has been made along this direction, a few major challenges have not been well addressed yet: (i) a vast majority of the literature is based on linear models that cannot capture complex non-linear inter-dependencies of user-item interactions; (ii) existing literature mainly ignores the latent relations among users and non-recommended items: hence may not properly reflect users' preferences in the real-world; (iii) current solutions are mainly based on historical data and are prone to *cold-start* problems for new users who have no interaction history.

To address the above challenges, we develop a Graph Regularized Cross-modal (GRC) learning model, a general framework to exploit transferable knowledge learned from user-item interactions as well as the external features of users and items in online personalized recommendations. In particular, the GRC framework leverage a non-linearity of neural network to model complex inherent structure of user-item interactions. We further augment GRC with the cooperation of the metric learning technique and a graph-constrained embedding module, to map the units from different dimensions (temporal, social and semantic) into the same latent space. An extensive set of experiments are conducted on two benchmark datasets as well as a large scale proprietary dataset from a major search engine demonstrates the power of the proposed GRC model in effectively capturing users' dynamic preferences under different settings by outperforming all baselines by a large margin.

## KEYWORDS

Contextual bandits, online recommendation, neural networks

**ACM Reference Format:**
Xian Wu, Suleyman Cetintas, Deguang Kong, Miao Lu, Jian Yang, and Nitesh V. Chawla. 2020. Learning from Cross-Modal Behavior Dynamics with Graph-Regularized Neural Contextual Bandit. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380178


*The Web Conference, Apr, 2020, Taipei*

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380178

## 1 INTRODUCTION

Personalized recommender systems have been widely applied in many real-world services such as e-commerce platforms, online advertising, consumption of online content (*e.g.*, books, music, etc.) [13]. Effective personalized recommendations not only can help customers identify items of interest more effectively, but can also substantially increase the profit for the service providers [38]. To this end, there exists rich literature devoted to developing collaborative filtering based approaches for modeling user-item interactions with the assumption that all data is collected beforehand. However, operating on the entire data in a batch fashion makes these approaches less suited for online recommendation scenarios, where new users or items arrive continually at a several orders of magnitude higher rate: *e.g.*, new ads in online advertising platforms or news at an online newspaper. Under these severe *cold-start* challenges, these algorithms either do not scale well when operating on a growing dataset as vast amount of new data arrive, or they completely ignore previously computed results and run from scratch on recent data without exploiting all available data [36].

To address the *cold-start* challenge in online recommendations, a number of attempts have been working on exploring contextual bandit algorithms to model user-item interactions in a timely manner, which yield the state-of-the-art performance [27, 33]. These methods adaptively learn the underlying representations of users or items (*e.g.*, user's preference or item's characteristic) by introducing the trade-off strategies in context-based exploration/exploitation for online decision-making [20]. In particular, the basic idea of *exploitation* is to maximize immediate reward given the current information, while *exploration* aims to gather more unbiased samples to improve the accuracy of preference learning. In each round of contextual bandit algorithms, they update users/items feature representations based on current positive (e.g., user click recommended items) and negative (e.g., user ignore recommended items) user-item interactions.

However, existing bandit methods are mostly limited to linear models or combine user and item feature embeddings via a simple non-linear concatenation [9, 20, 27], which cannot capture the complex non-linearity of latent user-item interactive structures, leading to suboptimal online recommendation results [29]. In this work, we strive to generalize the contextual bandit framework with modeling of non-linearities based on deep neural network architectures. There are several key technical challenges, in order to fully explore the neural architecture of contextual bandit framework:

**Inter-dependencies across multi-modal interactions**. To simplify the model design, conventional contextual bandit methods did not fully explore the negative user-item interactions (user's dislike for items) and completely ignored latent relations between users and non-recommended item candidates (unobserved user-item interactions) [3, 5]. However, in real life, users' preference can be learned from not only his/her positive feedback (*e.g.*, click), but also the knowledge of his/her negative and unobserved interactions with items [34]. For example, users' negative feedback on dislike recommended items may carry helpful information to reconsider the relations between users and other non-recommended candidates. In such cases, the multi-modal (*i.e.*, positive, negative and unobserved) user-item interactions are no longer independent and are highly correlated in a hierarchical way. Hence, it is challenging to distill cross-modal signals from the collective behaviors of users.

**Efficient feature learning for newly emerged users/items**. Although there exist recent work leveraging external features from users or items (*e.g.*, user social network information and item dependent relations) to quantify potential interactive structures for new users and items (without interaction logs), a deficiency is that they use the entire network structure to generate features for new users/items (*e.g.*, using Laplacian matrix computation) [1], which makes these methods computationally expensive and not scalable to online recommendation scenarios. In online recommendation scenarios, we believe it is of critical importance to develop a contextual bandit model that can exploit external features for newly emerged users and items in an efficient and explicit manner.

To overcome the aforementioned issues, this work develops a Graph Regularized Cross learning framework (GRC) by jointly modeling cross-modal user-item interactions and contextual features from either users or items in capturing users' future preferences in online recommendation. Specifically, we first propose to enhance the conventional contextual bandit framework with the neural network architecture, empowering it to model complex inherent user-item interactions with non-linearities. In addition, to comprehensively model effects of positive and negative interactions as well as the unobserved interactions (implicit feedback) between users and non-recommended item candidates, we augment our GRC model by developing a novel representation framework with a cross-interaction metric learning framework.

Furthermore, to realize the efficient user preference modeling of newly emerged users/items with the extracted ancillary features, we propose to leverage local bipartite graph structures between users and items. In particular, we develop a graph-regularized embedding module which allows external knowledge to guide the embedding initialization process of new users/items. With the cooperation of the metric learning framework and a graph-regularized embedding module, multi-modal user-item interactions and external network structural information of users/items can be leveraged to enhance the representation learning of both users and items. GRC bridges the gap between dynamic user behavior modeling and latent representations using graph embedding, which enables the speed up of capturing dynamic users' preferences.

The main contributions of this paper are summarized as follows:

- We study the problem of modeling users' dynamic preferences in online recommender systems, with attention to cross-modal user-item interactions and external features.
- We develop a novel framework GRC with a graph-regularized embedding module which is tailored to cooperate with the metric learning technique to model cross-modal user-item interactions. GRC is a general neural network architecture for contextual multi-armed bandit problem with the careful consideration of both positive and negative user feedback as well as the implicit feedback from non-recommended item candidates.
- Through extensive experiments conducted on three different real-world datasets, we demonstrate that GRC consistently outperforms several state-of-the-art baselines across various settings.

The organization of this paper is as follows. Section 2 formalizes the problem of contextual bandit learning. Section 3 presents the details of GRC framework to solve the problem. We explain the experimental results in Section 4. The related work is discussed in Section 5. Finally, we conclude this work in Section 6.

## 2 PRELIMINARIES AND PROBLEM FORMULATION

We first introduce some terminologies and formalize the problem. Then, we shortly recapitulate the widely used contextual bandit algorithms and discuss their limitations in online recommendations. To better explain the proposed method, we list the main notations we use in this paper in Table 1.

### 2.1 Problem Formulation

**Contextual Multi-armed Bandit Problem.** Contextual multi-armed bandit algorithms have been widely applied for online personalized recommendations to balance exploration and exploitation with the incorporation of various contextual information. In the multi-armed bandit problem, we consider a scenario of $I$ users (i.e., $u_1, ..., u_i, ..., u_I$) and $J$ items (i.e., $v_1, ..., v_j, ..., v_J$). For simplicity, we use the index of user $i$ representing $u_i$, and the index of item $j$ representing $v_j$ for rest of the paper. In each trail $t$, we regard the candidate item set as arms denoted as $\mathcal{A}^t = \{a_1^t, ..., a_k^t, ..., a_K^t\}$, where $K$ is the number of arms indexed by $k$.

At each round $t$, the algorithm observes a given user $u_i$ from the user set and $K$ arms. We denote user embedding and arm embedding at round $t$ as $\theta_{u_i^t}$ and $\theta_{a_k^t}$, respectively. Then, the arm with the highest expected reward is selected to recommend to user $i$ at timestamp $t$ and then receives the his/her feedback. In general, the contextual multi-armed bandit problem can be considered as a sequential decision problem, which aims to achieve highest long-term rewards. Formally, the objective is to maximize the accumulated rewards $R_T$ for previous $T$ trails as follows:

$$R_T = \sum_{t=1}^{T} r_k^t, \tag{1}$$

where $r_k^t$ is the actual reward of presented arm $a_k^t$ pulled by the bandit algorithm in trail $t$. A nature goal is to pull the arm with the

**Table 1: Symbols and Definitions**

| Symbol | Definition |
|---|---|
| $i, j, k, t$ | the indices of users, items, arms, trails |
| $I, J, K, T$ | the number of users, items, arms, trails |
| $\mathcal{A}^t$ | the set of arms for candidate item selection in trail $t$ |
| $\mathbf{x}_k^t$ | the context feature vector by integrating user $u_t$ and arm $a_k$ |
| $r_k^t, \hat{r}_k^t$ | the observed, expected reward of the pulled arm in trail $t$ |
| $e_k^t, d_k^t$ | the reward expectation and deviation of the pulled arm in trail $t$ |
| $\theta_{u_i^t}$ | the embedding of user $i$ in trial $t$ |
| $\theta_{a_k^t}$ | the embedding of arm $k$ in trial $t$ |

highest estimated reward in each trail. The key idea of contextual bandit algorithm is learning a reward mapping function in order to infer the arm with the highest reward to pull.

## 2.2 Linear Upper Contextual Bandit (LinUCB)

Among various contextual bandit algorithms, Linear Upper Confidence Bound (LinUCB) [20] is a key architecture for online personalized recommendation tasks and has shown to provide superior performance over others [1]. Many subsequent extensions enhanced the basic LinUCB framework by incorporating analysis on various properties of users and items [3, 9, 27]. Specifically, the reward mapping function of the LinUCB framework consists of two important components: (i) *reward expectation*: it estimates the interaction score between user $i$ and arm $a_k^t$ indicating the likelihood of user $i$'s interest in arm $a_k^t$. (ii) *reward deviation*: it applies upper confidence bound to assess the uncertainly of the reward expectations, which aims to form unbiased samples by pulling aims with high uncertainty to improve the learning accuracy. A smaller confidence interval indicates the lower uncertainty in the derived reward and a larger confidence interval means that the derived reward has a higher uncertainty. Formally, the reward between user $i$ and arm $a_k^t$ in trail $t$ can be calculated by the summation of reward expectation and deviation as follows:

$$\hat{r}_k^t = \underbrace{\mathbf{x}_k^t \mathbf{w}_k}_{\text{reward expectation } e_k^t} + \alpha \underbrace{\sqrt{\mathbf{x}_k^{t\,T} \mathbf{A}_k^{-1} \mathbf{x}_k^t}}_{\text{reward deviation } d_k^t}, \tag{2}$$

where $\mathbf{x}_k^t$ represents the concatenated feature vector of user embedding $\theta_{u_i^t}$ and arm embedding $\theta_{a_k^t}$, $\mathbf{A}_k := \mathbf{O}_k^T \mathbf{O}_k + \mathbf{I}$, $\mathbf{O}_k \in \mathbb{R}^{m \times d}$ is a design matrix in trial $t$, whose rows correspond to $m$ training inputs (*e.g.*, $m$ contexts that are observed previously for arm $a_k^t$), $\mathbf{w}_k$ denotes the learnable weight vector of arm $\mathbf{a}_k^t$, $y_k^t$ represents the predicted reward of arm $a_k^t$ in trail $t$, and $\alpha$ is the coefficient to balance the exploration and exploitation.

However, several significant limitations exist in the LinUCB based solutions: (i) it assumes that reward expectation of an arm is linear in the contextual feature vector. As a result, LinUCB cannot deal with the complex non-linear interaction structures between users and items in real-world applications. (ii) The LinUCB methods often count on a sufficient amount of positive and negative user-item interaction data, but fail to model the implicit feedback of item candidates. It follows that these methods may not comprehensively capture the latent interaction structures between users and items. (iii) The success of most existing LinUCB models rely on the various features from both users and items. Many practical scenarios, however, only partial features of users or items could be obtained for analysis at the training time. To overcome the above limitations, we propose to explicitly explore the cross-dimensional signals from multi-modal user-item interactions and partial external contextual features in advancing the online personalized recommendation task.

# 3 METHODOLOGY

In this section, we present the details of GRC framework, which pursues a full neural treatment of reward mapping function modeling to accurately predict the rewards between users and candidate arms. The overall framework is shown in Figure 1.

## 3.1 Neural Contextual Bandit Framework

In this work, we propose an instantiation of GRC adopting a multi-layer perceptron (MLP) module to endow the bandit algorithm of modelling non-linear structure of user-item interactions. In particular, in trail $t$, we first concatenate the embedding vector of user $\theta_{u_i^t}$ and arm $\theta_{a_k^t}$ as $\mathbf{x}_k^t$, and then feed the concatenated representation vector into the MLP module. The output of the final layer in MLP is the reward expectation $e_k^t$. By doing so, the above incorporation unifies the strengths of dynamics of contextual bandit algorithm and non-linearity of MLP for modelling time-evolving user-item latent structures. Formally, we present MLP as:

$$\begin{aligned} \mathbf{z}_1 &= \phi_1(\mathbf{W}_1 \mathbf{z}_0 + \mathbf{b}_1), \\ &\cdots \\ z_L &= \phi_L(\mathbf{W}_L \mathbf{z}_{L-1} + \mathbf{b}_L) \\ \hat{y} &= \mathbf{W}_o \mathbf{z}_L + \mathbf{b}_o, \end{aligned} \tag{3}$$

where $L$ is the number of hidden layers (indexed by $l$). For the $l$ layer, $\phi_n$, $\mathbf{W}_n$ and $\mathbf{b}_n$ represent the activation function (e.g., ReLU or tanh) of MLP layers and learnable parameters. We take the contextual vector $\mathbf{x}_k^t$ as the input of MLP (i.e., $\mathbf{z}_0 = \mathbf{x}_k^t$), the reward expectation is formally represented as $e_k^t = \text{MLP}(\mathbf{x}_k^t)$.

By doing so, we mitigate the limitation of existing contextual bandit techniques with the assumptions, *i.e.*, linear or simple non-linear payoff in estimating the uncertainty of reward deviation, However, directly deriving the corresponding upper confidence bound for uncertainty estimation remains as a daunting task, since the context information is provided in a dynamic environment and is not highly correlated with previous states and actions.

To address the aforementioned challenge, we apply dropout layers to learn the reward mapping function by unifying the strengths of neural network models and stochastic modeling [6]. Particularly, to supercharge our model with arbitrary depth and non-linearities, we apply dropout before every weight layer, which is shown to be mathematically equivalent to an approximation to the probabilistic deep Gaussian process [7]. After our model iterating to convergence, uncertainty estimates can be extracted from dropout neural networks. In particular, we sample $N$ times from Bernoulli$(N, p^l)$ distribution of network configurations for each layer $l$, and obtain its corresponding parameters $\{\mathcal{W}^1, \cdots, \mathcal{W}^N\}$. Here $\mathcal{W}^N = \{\mathbf{W}_1^N, \cdots, \mathbf{W}_L^N\}$ are the $L$ weight matrices sampled

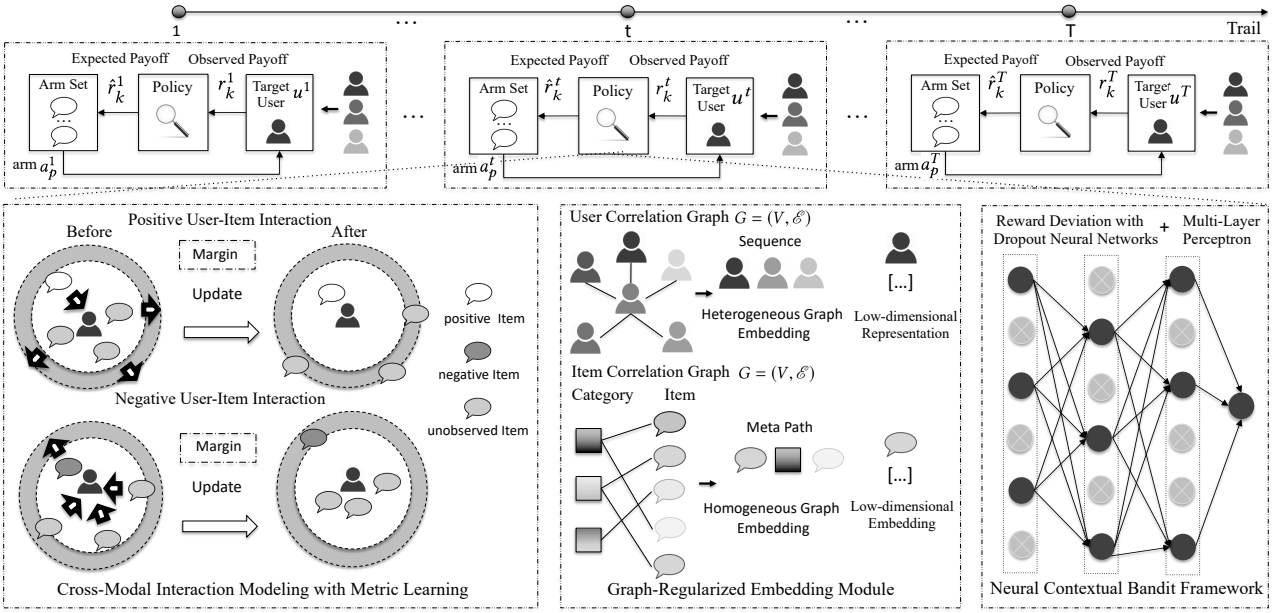



**Figure 1: The Graph-Regularized Cross-Modal Learning (GRC) Framework. GRC endows the contextual bandit architecture with the complex level of non-linearities, with the integration of a multi-layer perceptron and dropout mechanism. To accurately select an arm to users for recommendation, GRC carefully investigate the inter-dependencies among positive, negative and unobserved user-item interactions based on a deep metric learning framework. To alleviate the data incompleteness and sparseness issue, a graph-regularized embedding module is introduced to effectively transfer knowledge from ancillary features in guiding the cross-modal behavior learning.**

in $t$-th iteration. Thereafter, we can formally evaluate the Monte Carlo estimates with the input variables as:

$$\overline{e_k^t} \approx \frac{1}{N}\sum_{n=1}^{N} e_k^{(t,n)} = \frac{1}{N}\sum_{n=1}^{N} \text{MLP}^{(n)}(\mathbf{x}_k^t), \tag{4}$$

where $\text{MLP}^{(n)}$ represents the MLP with parameter set $\mathcal{W}^n$. Similarly, we can evaluate the second moment of input variables in Monte Carlo estimation process as follows:

$$d_k^t \approx \tau^{-1} + \frac{1}{N}\sum_{n=1}^{N}[(e_k^{(t,n)})^2 - (\overline{e_k^t})^2], \tag{5}$$

where $\tau$ is the model precision, which is defined as $\tau := \frac{pl^2}{2N\lambda}$ [28]. The collected results of stochastic forward passes through the model, and can be incorporated into our neural network model which is trained with dropout mechanism.

## 3.2 Cross-Modal Interaction Modeling with Metric Learning

Another challenge of existing contextual bandit algorithm is how to sufficiently explore user-item interactions, very few positive or negative rewards based on clicks are observed in return, with other unselected items in the candidate pool completely ignored. To address this challenge, we augment our neural contextual bandit framework to carefully investigate the inter-dependencies across multi-modal user-item interactions (*i.e.*, positive, negative and unobserved) with metric learning [24]. The basic idea of metric learning is to learn a distance metric to make similar input pairs closer to each other and make dissimilar input pairs further apart.

To model the latent relations between users and items, we apply triangle inequality relation structures to capture the dependencies among positive, negative and other unselected candidate user-item interactions. This is based on the assumption that users are more likely to be correlated with their interested items than uninterested ones [15]. In particular, given the preference representation $\theta_{u_i^t}$ of a user $i$, the representations of clicked items are expected be closer to $\theta_{u_i^t}$ than the representations of the unselected items in the feature space, while the representations of unselected items should be closer to $\theta_{u_i^t}$ than the representations of the unclicked items. In this way, we incorporate positive, negative interactions as well as the implicit feedback of those unselected candidate items. Formally, the metric learning module consists of two key steps:

(i) If the selected arm $a_p^t$ receives the positive feedback from user $i$ at trail $t$ (user $i$ is interested in item $a_p^t$) in trail $t$, our GRC will guide the representation learning process to make the embedding vectors of user $\theta_{u_i^t}$ and arm $\theta_{a_p^t}$ closer to each other, and the embedding vectors of user $\theta_{u_i^t}$ and unchosen arms (*i.e.*, $\theta_{a_p^t} \in \mathcal{A}^t \setminus \theta_{a_k^t}$) further apart.

(ii) If we observe the negative feedback for the selected arm $a_p^t$ from user $i$ at timestamp $t$ (user $i$ is uninterested in item $a_p^t$), the learned embedding vectors of user $\theta_{u_i^t}$ will be more different from

that of uninterested arm $\theta_{a_p^t}$, and be closer to embedding vector of unchosen arms (increase their probabilities to be selected in the next trail $t+1$).

We define the loss function of our metric learning module as:

$$\mathcal{L}_{Metric} = \sum_t \Big( \sum_{a_k^t \in \mathcal{A}_{neg}^t} r_p^t [m + \|\theta_{u_i^t} - \theta_{a_p^t}\|_2^2 - \|\theta_{u_i^t} - \theta_{a_k^t}\|_2^2]_+ + (1 - r_p^t)[m + \|\theta_{u_i^t} - \theta_{a_k^t}\|_2^2 - \|\theta_{u_i^t} - \theta_{a_p^t}\|_2^2]_+ \Big), \quad (6)$$

where $\mathcal{A}_{neg}^t$ is sampled from the rest candidate arm in pool $\mathcal{A}^t$. In addition, $\|\cdot\|_2$ denotes the 2-norm. Note that the feature vector and embedding vector share the same dimension size. $[\cdot]_+ = \max(\cdot, 0)$ is the standard hinge loss, and $m$ indicates a positive margin value.

## 3.3 Graph-Regularized Embedding Module

In real-world online recommendation scenarios, the contextual features of users or items are often incomplete. To transfer the knowledge from external sources and address the challenge of data incompleteness and sparseness, we develop a graph-regularized embedding module to bridge user behavior modeling with correlation graph embedding, such that the external knowledge of users and items can be leveraged to guide the cross-modal embedding and jointly alleviate data incompleteness and sparseness issues. The key idea of our graph-regularized embedding module is to learn latent representations of incoming users or items by leveraging their explicit connections with existing ones, such as users' social network and categorical dependencies between items. We first define the following inputs to our module:

Definition 1. ***Correlation Graph*** $G$. *Given relations between users or items from external knowledge, we define a correlation graph* $G = (\mathcal{V}, \mathcal{E})$ *in which* $\mathcal{V}$ *and* $\mathcal{E}$ *represents the set of users or items, and their relations, respectively. Particularly, for users, each node and each edge in the correlation graph* $G$ *represents the individual user and their social relationships. For items, each node and each edge in* $G$ *indicates the individual item and their categorical dependencies.*

In the our graph construction process, an edge between node $i$ and $i'$ is added when there exists social connection between $i$ and $i'$. Furthermore, since item-category bipartite graphs are heterogeneous in nature, involving diversity of node types (*i.e.*, items and categories), we construct a heterogeneous item correlation graph. In particular, each node in the heterogeneous item correlation graph $G$ represents the individual item/category. Each link in $G$ indicates the node relationship. Upon the above definitions, to learn feature representations for users in homogeneous graph $G$, we leverage node2vec [11] framework which is a neighborhood sampling strategy to smoothly interpolate between BFS and DFS. To learn a low-dimensional vector representation for each node in the heterogeneous item correlation graph $G$, we utilize metapath2vec [4] framework to preserve both structural and semantic correlations of graph $G$. In this work, we leverage random walk to treat network structures as the equivalent of sentences, which generates the input to our embedding module. We further define the random walk path in our model as:

Definition 2. ***Random Walk Path.*** *The random walk path in homogeneous user correlation graph is defined as a node sequence* $P = \{..., v_i, ...\}$ *wherein the node* $v_i$ *in the walk is randomly selected from the neighbors of its predecessor* $v_{i-1}$. *In heterogeneous item-category correlation graph, the random walk path with the meta-path generation scheme was defined as those in [4]. We define our graph-constrained embedding learning process as:*

$$\mathcal{L}_{Graph} = -\sum_{P \in \mathcal{S}(P)} \Big( \sum_{(v*, v) \in P} \log(\sigma(\theta_v^T \theta_{v^*})) + \sum_{v'} \mathbb{E}_{v' \sim \text{Dist}(v')} \log(\sigma(-\theta_{v'}^T \theta_{v^*})) \Big), \quad (7)$$

where $\sigma$ represents Sigmoid activation function, $v$ and $v'$ denotes the neighborhood context and non-neighborhood nodes of center node $v^*$ on random walk path $P$, and $\mathcal{S}(P)$ is the path set. We define *RandomWalk*$(G, a_k^t)$ function (defined in [4, 11]) to represent the random walk process on $G$ which consider $a_k^t$ as the starting point. By deriving the graph-constrained embedding, the embedding vectors of newly emerged users or items are initialized in a more reasonable way than random initialization.

**Algorithm 1:** The training process of GRC model.

**Require:** batch size $b_{size}$
1: Randomly initialize parameters
2: TrainBatch=[]
3: **for all** trail $t$ **do**
4: Observe features of all arms $\mathcal{A}^t$
5: **for all** arm $a_k^t$ in the candidate pool $\mathcal{A}^t$ **do**
6: **if** arm $a_k^t$ is new **then**
7: Path=RandomWalk(G, $a_k^t$)
8: initialize the embedding vector of $\theta_{a_k^t}$ by Eq. (7)
9: **end if**
10: calculate $\hat{r}_k^t = \overline{e_k^t} + \alpha d_k^t$ according to Eq. (5)
11: **end for**
12: Choose arm $a_p^t = \arg\max_{a_k^t \in \mathcal{A}^t} \hat{r}_k^t$ and observe a real-valued reward $r_p^t$
13: Sample a set of negative samples $\mathcal{A}_{neg}^t$ from the rest candidates in pool $\mathcal{A}^t$
14: TrainBatch.append($u^t, a_p^t, r_p^t, [..., \hat{r}_k^t, ...], \mathcal{A}_{neg}^t$)
15: **if** length(TrainBatch)==$b_{size}$ **then**
16: calculate the loss by Eq. (9)
17: update all parameters by Adam Optimization
18: TrainBatch=[]
19: **end if**
20: **end for**

## 3.4 The Learning Process of GRC Framework

As we introduced in Section 2, the objective of the arm selection strategy in online learning scenarios is to derive the value of $\hat{r}_k^t$ which denotes the estimated reward of $k$-th arm candidate in trail $t$. In general, online recommendation can be considered as a personalized ranking task. To this end, we learn the parameters of our GRC with a ranking-aware objective, *i.e.*, all arms with positive feedback should be ranked higher than the arms with the ones with negative feedback. We generate the ranking-aware objective

with the integration of pointwise [16] and listwise loss [37], which is more beneficial for personalized ranking task. In the training process of GRC, if arm $a_k^t$ is observed positively interacted with the target user, we will assign a higher ranking score to it. Otherwise, the ranking score of arm $a_k^t$ will be set to be lower than other aims. Therefore, our GRC framework aims to predict the relative orders between user-item interactions, instead of inferring their absolute scores as optimized in pointwise loss. To maximize the likelihood for the ranking score vector, we define our loss function as:

$$\mathcal{L}_{Payoff} = -\sum_t r_p^t \ \log(\frac{e_p^t}{\sum_k e_k^t}) + (1 - r_p^t) \ \log(1 - \frac{e_p^t}{\sum_k e_k^t}). \quad (8)$$

By integrating the loss function of our triple relation in Equation 6, we define our designed joint objective function as:

$$\mathcal{L}_{joint} = \mathcal{L}_{Payoff} + \lambda \mathcal{L}_{Metric} + \mathcal{L}_{Graph}, \quad (9)$$

where $\lambda$ is the coefficient to control the weight of the term for metric learning module. The GRC can be learned by minimizing the above loss function between the observed user-item interactions and the estimated reward. We denote the batch size as $b_{\text{size}}$ and solve the above optimization problem using the Adam optimizer. Based on its sub-steps, we could obtain the total expected payoff $E(\sum_{t=1}^{T} r_p^t)$ from previous $T$ trails, where $a_p^t$ is the selected arm which maximizes the expected payoff in trail $t$.

Algorithm 1 summarizes the training process of *GRC*. In each iteration of trail $t$. we could observe features of all arms $\mathcal{A}^t$ for each arm $a^t$ and conduct random walks on graph $G$ if the arm is new to us, otherwise, we update based on Equations 3 and 5. After that, we choose an arm $a_p^t$ which generates the largest reward. In addition, we also choose a set of other arms $\mathcal{A}_{neg}^t$. $a_p^t$ together with $\mathcal{A}_{neg}^t$ are add into the training batch for later optimization. If we accumulate enough arms in the training batch, we calculate the loss based on Eqn. (9) and update the model parameters by Adam optimization. We repeat this process for all trails to learn the hidden parameters.

# 4 EVALUATION

In this work, we performed extensive experiments on three real-world datasets, including two benchmark datasets (*i.e.*, Delicious and LastFM datasets) and a large-scale click stream data (*i.e.*, Ads dataset) from a major commercial search engine. We also compared *GRC* with several state-of-the-art baselines. Particularly, our experiments aim to answer the following research questions:

- **Q1**: How is the performance of our *GRC* in online personalized recommendation tasks as compared to state-of-the-art methods?
- **Q2**: How is the performance of *GRC* variants with different combinations of key components in the joint framework?
- **Q3**: How is the performance of *GRC* with different configurations of loss function?
- **Q4**: How does our *GRC* work for online personalized recommendation task with different exploration coefficient (*i.e.*, $\alpha$) and arm set size (*i.e.*, $K$)?
- **Q5**: How do the key hyperparameter settings impact *GRC*'s performance?

In the following subsections, we first present the experimental settings and then answer the above research questions in turn.

## 4.1 Experimental Settings

*4.1.1* **Data Description.** In our experiments, we evaluate the model performance on three different types of datasets: (i) social bookmarking web service data–Delicious; (ii) music streaming service data–LastFM; (iii) a large-scale ads click stream data from a major search engine. The statistics of the three datasets are summarized in Table 2 and details are shown as follows:

- **The LastFM Data**. The LastFM dataset has been widely used as a benchmark in evaluating the performance of bandit algorithms [35]. This dataset contains 1,892 users and 17,632 artists (items), which was collected from a music streaming service website [1]. In this dataset, we generated payoffs of recommendation candidates for each user by leveraging the information of his/her "interested artists". In particular, in each trail $t$, the payoff $r_k^t$ is set to 1 if the selected user $u_t$ listened to any music from artist $a_k^t$. Otherwise, $r_k^t = 0$.
- **The Delicious Data**. The Delicious data is another benchmark dataset for bandit algorithm performance validation [33]. 1,861 users and 69,226 URLs (items) were included in this dataset. The payoffs between users and items are generated based on the bookmark behavior information in this dataset. Specifically, we set the payoff $r_k^t = 1$ if the user bookmarked URL $a_k^t$ and $r_k^t = 0$ otherwise in each trail $t$.
- **Ads Click Stream Data**. We collected a large-scale, proprietary search ads click stream data from Yahoo Gemini platform, which offers an online search and native advertising. Search advertising is a multi-billion dollar industry where advertisers promote their products to users by having search engines display their advertisements (ads) on contextually relevant search results pages. In a commercial search engine, a large number of new ads will be continuously introduced to the system leading to a massive cold start problem, which provides us a good opportunity to investigate the performance of our *GRC* in real-world recommendation scenarios.

*4.1.2* **Data Pre-processing.** To evaluate our *GRC* framework in the settings which fit the contextual bandit problem, we conducted the following data pre-processing steps as follows:

**Pre-processing of LastFM and Delicious Datasets**. Following the same experimental settings in [36], we first constructed the TF-IDF feature vector of each item (arm) using its associated tags, and then reduced the dimension of the generated feature vectors via the Principle Component Analysis (PCA) technique. Specifically, we selected the first 25 principle components to generate the context feature vectors in both two datasets, *i.e.*, feature dimension is 25.

[1] http:www.last.fm

**Table 2: The Statistics of Datasets**

| | #Users | #URLs | #Tags | #Bookmarks | #Relations |
|---|---|---|---|---|---|
| Delicious | 1, 867 | 69, 226 | 53, 388 | 104, 799 | 7, 668 |
| | #Users | #Artists | #Tags | #Listened | #Relations |
| LastFM | 1, 892 | 17, 632 | 11, 946 | 437, 594 | 15, 329 |
| | #Queries | #Ads | #Categories | #Clicks | #Impressions |
| Ads Click | 3, 111, 569 | 18, 197 | 1, 000 | 2, 256, 380 | 72, 415, 447 |

Then, we set the size of candidate arm set $K$ as 100 and generate it as follows: we chose one arm from the set of nonzero payoff items based on the global observations in the datasets and selected the remaining 99 ones from the same set of zero-payoff items randomly.

**Pre-processing of Ads Click Stream Data**. In our prorietary, large-scale search-ads click-stream data, we follow the same experimental settings in [39] and build the ads pool by randomly selecting 100 ads ($K = 100$) from the entire set of ads. This dataset contains 72.4 million impression records and 2.2 million click records that represent the users' response to the ads shown on the large number of queries. Each impression record represents an ad shown to the user on the search result page of a particular search query. If the user clicks on the shown ad, a click event happens and is logged accordingly. Click attribution is done at the event level, which is typically a unique identifier for each ad impression and its corresponding click (if a click event happens). We set the dimension of the embedding feature vectors of queries as 300 and generate the embedding vectors for the whole queries rather than individual query terms, achieving the aggregation level query embedding to capture the semantics [10].

*4.1.3* **Evaluation Protocols and Metrics.** In our experiments, we evaluated the performance of all compared algorithms using the unbiased offline evaluation protocol proposed in [19]. In particular, in each trail $t$, the interaction events between user $u_t$ and the selected arm $a_k^t$ by each method will be evaluated using ground truth information generated from historical user-item interaction events. In the training process, our *GRC* will update the model parameters based on the output by a particular approach and move forward to the next trail $(t + 1)$.

We evaluated the model performance by using the following two widely used metrics in online personalized recommendation: *Cumulative Rewards (CR)* and *Click Through Rate (CTR)* [35] (Note that a higher CC and CTR score indicates better model performance):

- **Cumulative Rewards (CR):** it represents the cumulative value of each observed reward $r_k^t$ from all previous trails (refer to Eq 2 for mathematical definition).
- **Click Through Rate (CTR):** it indicates the ratio of clicks on recommended items divided by the number of recommendations. We computed the average CTR in every 5000 trails based on the aforementioned unbiased offline evaluation protocol.

*4.1.4* **Baseline Methods.** In our evaluations, we compare the performance *GRC* against three types of baselines: i) conventional contextual bandit methods (*i.e.*, *GCLUB*[21], *LinUCB*[20], *UCBPMF*[27], *CLUB*[9], *PTS*[9]); ii) bandit algorithms with neural network architectures (*i.e.*, *NN*[5], *DW*[3]); iii) online learning framework for recommendation system (*i.e.*, *eALS* [13]). Since there is no direct contextual multi-armed bandit scenarios, it is not fair to compare directly against second and third type of baselines. Instead, we integrate them with $\epsilon$-greedy bandit framework which serves as a generic bandit model for online personalized recommendations as suggested in [31].

*4.1.5* **Reproducibility and Parameter Settings.** We summarized the parameter settings of *GRC* in our experiments in Table 3. In addition, we vary each of key parameters in *GRC* and fix others to examine the parameter sensitivity. We implemented our framework based on TensorFlow and chose Adam [17] as our optimizer to learn the model parameters. The hyperparameter settings are optimized with the grid search strategy [11]. For all bandit algorithm, we used the same explore coefficient $\alpha$. For all neural network based methods, we use the same parameters as *GRC* which are listed in Table 3. In addition, the parameters of all baselines have been carefully tuned to the best performance using the grid search strategy.

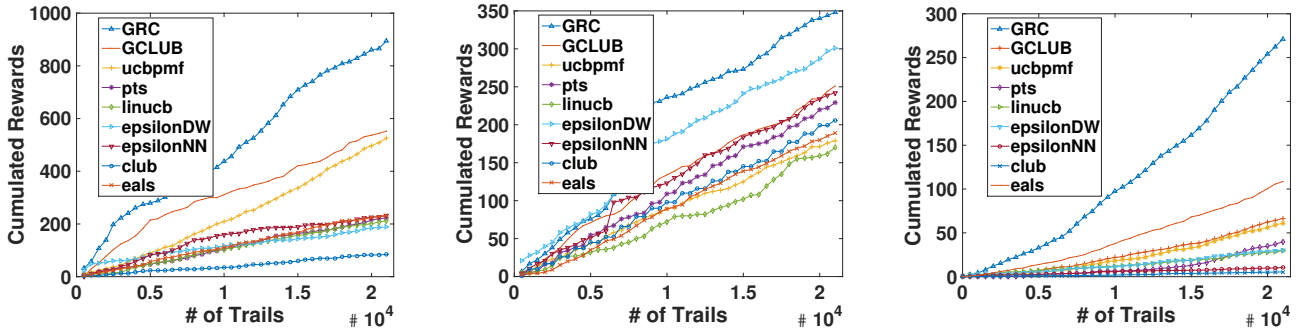
**Table 3: Parameter Settings**

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| # Negative Samples | 8 | Exploration Parameter $\alpha$ | 0.3 |
| # Hidden Layers | 4 | Dropout Ratio | 0.4 |
| Margin | 1 | Metric Learning Ratio $\lambda$ | 0.1 |
| Batch Size | 8 | Learning Rate | 0.001 |

## 4.2 Performance Comparison (Q1)

We evaluated the performance of all compared algorithms on three different type of datasets (*i.e.*, the LastFM, Delicious and Ads Click Stream Data), and reported the evaluation results in Figure 2 and Figure 3 in terms of *CR* and *CRT*, respectively. Due to space limit, we only reported the evaluation results in terms of CTR on ads click stream data and similar results could be observed for other datasets. In our experiments, we set the size of candidate arm set $K$ as 100. All the methods are executed up to 21, 000 iterations on each dataset. Based on the experimental results, similar trend can be obtained with more trails (*i.e.*, $> 21,000$). 21, 000 is only tentatively set for illustrative purpose. From the evaluation results, we summarize several key observations as follows:

*First and foremost*, we can observe that *GRC* consistently outperforms other competitive baselines on all datasets over different trials. Specifically, *GRC* achieves significant improvements over the best performing baseline in terms of CR and CTR on each dataset. This set of results clearly demonstrate that *GRC* achieves the state-of-the-art performance for online recommendations. It is important to note that the performance gain between *GRC* and other baselines does increase when the number of trails increases in most cases, which suggests that our *GRC* is more efficient and effective in capturing users' dynamic preferences at the early stage of exploration. In general, the above observations justify the efficacy of *GRC* which collectively models users' various response interactions and eventually helps to select the most likely recommendations/clicks.




(a) LastFM Data (b) Delicious Data (c) Ads Click Stream Data

**Figure 2: Performance comparison of all algorithms on three real-world datasets in terms of *CR*.**

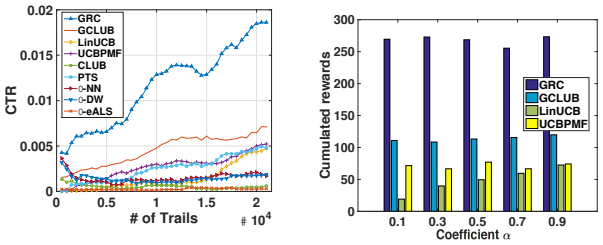
*Second*, the large performance gap between *GRC* and conventional contextual bandit algorithms (*i.e.*, LinUCB, UCBPMF, CLUB and PTS) clearly shows the limitation of those approaches: (i) the inner product, which simply combines the multiplication of latent features linearly, may not be sufficient to capture the complex structure of dynamic user-item interactions; (ii) these models do not consider the imbalance issue and produce suboptimal performance with limited amount of positive user-item interaction data in real-world online personalized recommender systems. Furthermore, another interesting observation is that our *GRC* method could achieve more significant performance improvement over other baselines in Ads click stream data as compared to LastFM and Delicious data. This is due to that modeling cross-modal imbalanced user-ads interactions is more effective than modeling user-music and user-book interactions in LastFM and Delicious data.

*Third*, as compared to neural network based bandit algorithms that also apply the deep learning techniques but in different ways, *GRC* exhibits significant performance improvement. This is remarkable, since *GRC* implies the potential of improving existing neural bandit algorithms in learning better user/item latent representations, by designing an integration framework of a graph-regularized embedding module and metric learning technique.

## 4.3 Model Ablation Study of *GRC* (Q2)

In addition to comparing *GRC* with state-of-the-art techniques, we also aim to get a better understanding of the proposed framework and evaluate the key components of *GRC*. Particularly, we aim to answer the following question: whether each key learning component plays a crucial role in the joint representation learning model *GRC*? Hence, in our evaluation, we consider three model variants of *GRC*:

- **Efficacy of the graph-regularized cross-modal learning model** (*GRC*-n): To make a fair comparison, we design the variant only with the neural contextual bandit architecture to model the non-linear structure of user-item interactions, *i.e.*, without the cooperation between graph-regularized embedding module and metric learning technique.




(a) (b)

**Figure 3: (a) Performance comparison of all algorithms on Ads Click Stream Data in terms of *CTR*; (b) Performance comparison v.s. values of coefficient $\alpha$ which balances the exploration and exploitation.**

- **Efficacy of cross-modal interaction learning framework** (*GRC*-g): To investigate the effect of our developed cross-modal interaction learning framework based on metric learning, we proposed another simplified version of *GRC* without the component of cross-modal user-item interaction modeling.

- **Efficacy of graph-regularized embedding module** (*GRC*-m): To show the effect of our model in learning robust representations for users/items using their the external knowledge under data incompleteness, we proposed a simplified version of *GRC* without the graph-regularized embedding module in encoding the dependencies between users or items.

We reported the evaluation results in Table 4. We could notice that the full version of *GRC* achieves the best performance in all cases, which suggests: (i) our graph-regularized embedding module can utilize external knowledge (*i.e.*, explicit user/item connections) to effectively capture contextual signals of users and items with

**Table 4: Model ablation study of *GRC* in terms of *CR* and *CTR* on three datasets.**

| Data Source | LastFM | | Delicious | | Ads Click | |
|---|---|---|---|---|---|---|
| Metric | CR | CTR | CR | CTR | CR | CTR |
| *GRC*-n | 256 | 1.04% | 196 | 0.94% | 31 | 0.14% |
| *GRC*-g | 319 | 1.44% | 227 | 1.16% | 32 | 0.12% |
| *GRC*-m | 422 | **2.60%** | 225 | 1.00% | 251 | 1.71% |
| *GRC* | **894** | 2.26% | **348** | **1.18%** | **272** | **1.82%** |

**Table 5: Mathematical definitions of different loss functions.**

| Loss | Definition |
|---|---|
| MSE | $\sum_t \|r_p^t - E_p^t\|_2^2$ |
| CE | $-\sum_t \left(r_p^t \log(E_p^t) + (1 - r_p^t) \log(1 - E_p^t)\right)$ |
| NS | $-\sum_t r_p^t \left(\log(\sigma(E_p^t)) + \sum_{k'} \mathbb{E}_{k' \sim \text{Dist}(a)} \log(\sigma(-E_{k'}^t))\right)$ |

incomplete feature information. (ii) When the metric learning technique is applied in the embedding learning process to model the inter-dependencies between multi-modal user-item interactions, the performance for online personalized recommendation is improved. (iii) The efficacy of *GRC* in modeling complex interdependencies across multi-modal interactions.

**Table 6: Loss function configuration effect of *GRC* in terms of *CR* and *CTR* on three datasets.**

| Data Source | LastFM | | Delicious | | Ads Click | |
|---|---|---|---|---|---|---|
| Metric | CR | CTR | CR | CTR | CR | CTR |
| *GRC*-mse | 556 | 2.04% | 212 | 1.14% | 131 | 1.12% |
| *GRC*-ce | 220 | 1.06% | 236 | 1.05% | 208 | 1.72% |
| *GRC*-neg | 883 | **3.36%** | 227 | 1.03% | 162 | 1.24% |
| *GRC* | **894** | 2.26% | **348** | **1.18%** | **272** | **1.82%** |

## 4.4 Effect of Loss Function Configuration in *GRC* Framework (Q3)

To show the effect of loss function configuration in our developed *GRC*, we replaced our designed loss function with three variants:

- *GRC*-mse: with the loss function based on mean squared error [32].
- *GRC*-ce: with the loss function based on cross entropy [12].
- *GRC*-neg: with the loss function using negative sampling technique [26].

Specifically, *GRC*-mse and *GRC*-ce correspond to the loss function generation with pointwise method, and *GRC*-neg represents the loss function generation with listwise method. The mathematical definitions of different types of loss function are presented in Table 5. We can observe that *GRC* achieves better performance as compared to other variants in most evaluation cases. The above empirical evidence provides support for our motivation of designing our loss function which integrates the metric learning with the contextual bandit framework, to alleviate the data imbalance issue.

## 4.5 Effect of Exploration Coefficient $\alpha$ and Arm Set Size $K$ (Q4)

**Effect of Exploration Coefficient $\alpha$.** The exploration coefficient $\alpha$ plays a pivotal role in balancing the exploration and exploitation in bandit algorithm. We further performed experiments to investigate the effect of $\alpha$ in *GRC* and other representative baselines (*i.e.*, GCLUB, LinUCB and UCBPMF) which involve the same coefficient parameter. Figure 3(b) shows the evaluation results as measured by CR on the ads click stream data. According to the results, *GRC* is not strictly sensitive to parameter $\alpha$ and is able to consistently reach high performance under different parameter choices, further suggesting the robustness of our *GRC* in the trade-off between the exploration and exploitation in the online recommendation.

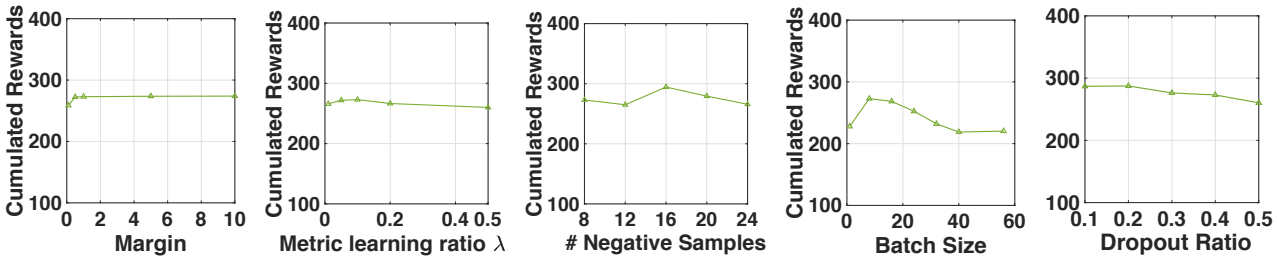
**Effect of Arm Set Size $K$.** To further investigate the robustness and generalization ability of different bandit algorithms in practical online recommendation scenario, we examined the scenarios with different sizes of the arm set (*i.e.*, $K$). As shown in Table 7, we varied the value of $K$ from 100 to 300 and reported the accumulated rewards at the 21, 000-th trails. We can notice that *GRC* consistently outperforms all compared methods over different size of arm set. Moreover, another interesting observation is that *GRC* is more robust to the large value of $K$, *i.e.*, the accumulated rewards of *GRC* decreases much slower than all compared baselines and finally achieve similar performance with smaller size of arm set. This reflects the strong expressiveness and generalization of our *GRC* since increasing the number of candidate arms does not lead to suboptimal performance. Hence, our proposed *GRC* has remarkable generalization ability, and thus is very suitable for practical use in large-scale online recommender systems.

**Table 7: Performance comparisons of different methods in ads dataset in terms of arm set size $K$.**

| Value of $K$ | 100 | | 200 | | 300 | |
|---|---|---|---|---|---|---|
| Metric | CR | CTR | CR | CTR | CR | CTR |
| GCLUB | 108 | 0.71% | 96 | 0.58% | 73 | 0.44% |
| LinUCB | 39 | 0.46% | 23 | 0.24% | 20 | 0.12% |
| UCBPMF | 66 | 0.52% | 62 | 0.34% | 56 | 0.32% |
| CLUB | 10 | 0.04% | 5 | 0.02% | 3 | 0.05% |
| PTS | 61 | 0.48% | 47 | 0.33% | 48 | 0.27% |
| $\epsilon$-NN | 30 | 0.17% | 10 | 0.03% | 6 | 0.03% |
| $\epsilon$-DW | 29 | 0.21% | 17 | 0.12% | 10 | 0.03% |
| $\epsilon$-eALS | 10 | 0.03% | 3 | 0.02% | 3 | 0.02% |
| *GRC* | **272** | **1.82%** | **208** | **1.26%** | **172** | **1.08%** |

## 4.6 Hyperparameter Sensitivity Studies (Q5)

The *GRC* model involves several parameters (*e.g.*, Metric Learning Ratio $\lambda$, Dropout Ratio and # of Negative Samples). To investigate the robustness of *GRC* framework, we examine how the different




(a) Margin (b) Metric Learning Ratio $\lambda$ (c) # of Negative Samples (d) Batch Size (e) Dropout Ratio

**Figure 4: Hyperparameter sensitivity studies of *GRC* on Ads Click Stream Dataset**

choices of parameters affect the performance of *GRC* on Ads click stream data. Except for the parameter being tested, we set other parameters at the default values (see Table 3).

Figure 4 shows the evaluation results on ads click stream data (measured by CR) as a function of one selected parameter when fixing others. Overall, we observe that *GRC* is not strictly sensitive to these parameters, which further demonstrates the robustness of our proposed framework. In particular, we can observe that *GRC* achieves better performance with the increase of batch size, but model performance decreases with a larger value of batch size. The reason is that a smaller batch size will update model parameters in a more timely manner. We set batch size as 8 in our experiments due to the consideration of the trade-off between the effectiveness and computational cost. Additionally, different from the low impact of margin and metric learning ratio, the dropout ratio is negatively correlated with the model performance as shown in Figure 4(e).

## 5 RELATED WORK

**Contextual Bandit Algorithms.** Online learning is one of the fundamental challenges in recommendation systems. Prior works have made significant advances to develop various contextual bandit algorithms for online personalized recommendations [8, 14, 18, 20, 25, 33, 39, 40]. Specifically, the early pioneer work by Li *et al.* [20] applied the contextual-bandit approach to the personalized recommendation problem with the assumption of the expected reward is linear with respect to context. Many follow-up works extend the basic LinUCB framework by considering historical data reuse [14], latent feature learning [34] and time varying contextual scenario [34].

Motivated by the unprecedented achievements of deep recommendation techniques, many recent works followed the idea of integrating transfer learning and neural network architecture [25, 29, 30]. For example, Liu *et al.* [30] applied neural contextual multi-armed bandits to online learning of response selection in retrieval-based dialog models. Different from existing contextual bandit methods, we propose to not only leverage the neural network architecture to capture the dynamic, non-linear, user-item interactions; but also enhance online personalized recommendations via the integration of metric learning and graph-regularized embedding module.

**Collaborative Contextual Bandit Framework.** Collaborative Filtering (CF) has been widely applied to various recommendation systems [2, 12, 23]. There exist several attempts on combining contextual bandit models with collaborative filtering techniques [1, 22, 36]. For example, Wu *et al.* [36] proposed a collaborative bandit algorithm using the adjacency graph among users for online updating settings. Li *et al.* [22] took into account the collaborative effects in bandit algorithms by grouping users based on their predefined relations. However, these approaches are computationally expensive and require repeated offline training using pre-obtained network information, which is unscalable and inefficient in the training process. Therefore, it is very difficult to adapt these models to real-time online recommendations. To address this issue, our GRC enables an effective online learning process to capture time-evolving user-item interactions in a timely manner.

## 6 CONCLUSION

In this paper, we studied the contextual bandit problem for online personalized recommendations. We proposed a graph-regularized cross-modal learning framework that collectively addresses the critical challenges of modeling complex non-linear user-item interactions. In particular, we comprehensively utilize both positive, negative user-item interactions as well as unrecommended items via metric learning. An extensive set of experiments on three real-world datasets (*i.e.*, two public benchmark datasets along with a proprietary, large-scale advertising dataset from a major search engine) demonstrate that GRC achieves significant performance improvements over the state-of-the-art baselines. For future work, we plan to further extend GRC to other sequence modeling tasks for online recommender systems and online advertising scenarios, *e.g.*, cold-start ads conversion estimation, personalized recommendation for ads campaign and ads re-engagement.

## ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation (NSF) grants CNS-1629914.

## REFERENCES

[1] Nicolo Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. 2013. A gang of bandits. In *NIPS*. 737–745.
[2] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *SIGIR*. ACM, 335–344.
[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *RecSys*. ACM, 7–10.

[4] Yuxiao Dong, Nitesh V Chawla, et al. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD*. ACM, 135–144.

[5] Stefan Faußer and Friedhelm Schwenker. 2015. Neural network ensembles in reinforcement learning. *Neural Processing Letters* 41, 1 (2015), 55–69.

[6] Yarin Gal. 2016. Uncertainty in deep learning. *University of Cambridge* (2016).

[7] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML*. 1050–1059.

[8] Claudio Gentile, Shuai Li, Purushottam Kar, Alexandros Karatzoglou, Giovanni Zappella, and Evans Etrue. 2017. On context-dependent clustering of bandits. In *ICML*. JMLR. org, 1253–1262.

[9] Claudio Gentile, Shuai Li, and Giovanni Zappella. 2014. Online clustering of bandits. In *ICML*. 757–765.

[10] Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, Fabrizio Silvestri, and Narayan Bhamidipati. 2015. Context- and content-aware embeddings for query rewriting in sponsored search. In *SIGIR*. ACM, 383–392.

[11] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*. ACM, 855–864.

[12] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. ACM, 173–182.

[13] Xiangnan He, Hanwang Zhang, et al. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *SIGIR*. ACM, 549–558.

[14] Katja Hofmann, Anne Schuth, et al. 2013. Reusing historical interaction data for faster online learning to rank for IR. In *WSDM*. ACM, 183–192.

[15] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative metric learning. In *WWW*. ACM, 193–201.

[16] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Recsys*. ACM, 79–86.

[17] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Nathan Korda, Balázs Szörényi, and Li Shuai. 2016. Distributed clustering of linear bandits in peer to peer networks. In *JMLR*, Vol. 48. International Machine Learning Societ, 1301–1309.

[19] Lihong Li, Wei Chu, et al. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *WSDM*. ACM, 297–306.

[20] Lihong Li, Wei Chu, John Langford, et al. 2010. A contextual-bandit approach to personalized news article recommendation. In *WWW*. ACM, 661–670.

[21] Shuai Li, Claudio Gentile, and Alexandros Karatzoglou. 2016. Graph clustering bandits for recommendation. *arXiv preprint arXiv:1605.00596* (2016).

[22] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. 2016. Collaborative filtering bandits. In *SIGIR*. ACM, 539–548.

[23] Xiaopeng Li and James She. 2017. Collaborative Variational Autoencoder for Recommender Systems. In *KDD*. ACM, 305–314.

[24] Shengcai Liao, Yang Hu, and Xiangyu Zhu. 2015. Person re-identification by local maximal occurrence representation and metric learning. In *CVPR*. 2197–2206.

[25] Bo Liu, Ying Wei, Yu Zhang, Zhixian Yan, and Qiang Yang. 2018. Transferable contextual bandit for cross-domain recommendation. In *AAAI*.

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, et al. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 3111–3119.

[27] Atsuyoshi Nakamura. 2015. A ucb-like strategy of collaborative filtering. In *ACML*. 315–329.

[28] Valentin Peretroukhin, Lee Clement, and Jonathan Kelly. 2017. Reducing drift in visual odometry by inferring sun direction using a bayesian convolutional neural network. In *ICRA*. IEEE, 2035–2042.

[29] Carlos Riquelme, George Tucker, and Jasper Snoek. 2018. Deep Bayesian Bandits Showdown: An Empirical Comparison of Bayesian Deep Networks for Thompson Sampling. *ICLR* (2018).

[30] Suvash Sedhain, Aditya Krishna Menon, et al. 2018. Customized nonlinear bandits for online response selection in neural conversation models. In *AAAI*.

[31] Michel Tokic. 2010. Adaptive $\varepsilon$-greedy exploration in reinforcement learning based on value differences. In *AAAI*. 203–210.

[32] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *NIPS*. 2643–2651.

[33] Huazheng Wang, Qingyun Wu, and Hongning Wang. 2016. Learning hidden features for contextual bandits. In *CIKM*. ACM, 1633–1642.

[34] Huazheng Wang, Qingyun Wu, and Hongning Wang. 2016. Learning hidden features for contextual bandits. In *CIKM*. ACM, 1633–1642.

[35] Qingyun Wu, Naveen Iyer, and Hongning Wang. 2018. Learning Contextual Bandits in a Non-stationary Environment. In *SIGIR*.

[36] Qingyun Wu, Huazheng Wang, Quanquan Gu, and Hongning Wang. 2016. Contextual bandits in a collaborative environment. In *SIGIR*. ACM, 529–538.

[37] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: theory and algorithm. In *ICML*. ACM, 1192–1199.

[38] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *WSDM*. ACM, 283–292.

[39] Chunqiu Zeng, Qing Wang, et al. 2016. Online context-aware recommendation with time varying multi-armed bandit. In *KDD*. ACM, 2025–2034.

[40] Guanjie Zheng, Fuzheng Zhang, et al. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *WWW*. ACM, 167–176.