BanditProp: Bandit Selection of Review Properties for Effective Recommendation

XI WANG, University of Glasgow, UK
IADH OUNIS, University of Glasgow, UK
CRAIG MACDONALD, University of Glasgow, UK

Many recent recommendation systems leverage the large quantity of reviews placed by users on items. However, it is both challenging and important to accurately measure the usefulness of such reviews for effective recommendation. In particular, users have been shown to exhibit distinct preferences over different types of reviews (e.g. preferring longer vs. shorter or recent vs. old reviews), indicating that users might differ in their viewpoints on what makes the reviews useful. Yet, there have been limited studies that account for the personalised usefulness of reviews when estimating the users' preferences. In this paper, we propose a novel neural model, called BanditProp, which addresses this gap in the literature. It first models reviews according to both their content and associated properties (e.g. length, sentiment and recency). Thereafter, it constructs a multi-task learning (MTL) framework to model the reviews' content encoded with various properties. In such an MTL framework, each task corresponds to producing recommendations focusing on an individual property. Next, we address the selection of the features from reviews with different review properties as a bandit problem using multinomial rewards. We propose a neural contextual bandit algorithm (i.e. ConvBandit) and examine its effectiveness in comparison to eight existing bandit algorithms in addressing the bandit problem. Our extensive experiments on two well-known Amazon and Yelp datasets show that BanditProp can significantly outperform one classic and six existing state-of-the-art recommendation baselines. Moreover, BanditProp using ConvBandit consistently outperforms the use of other bandit algorithms over the two used datasets. In particular, we experimentally demonstrate the effectiveness of our proposed customised multinomial rewards in comparison to binary rewards, when addressing our bandit problem.

CCS Concepts: • **Information systems → Recommender systems**; **Personalization**.

Additional Key Words and Phrases: recommendation systems, bandit search, user behaviour modelling, review property

1 INTRODUCTION

Recent recommendation studies [16, 23, 44] have shown the value of identifying those useful reviews that better capture the users' preferences as well as the items' characteristics in order to improve the recommendation performance. In particular, the properties of reviews, such as their length, recency and associated ratings have been shown to improve review-based recommendation systems [19, 56]. Moreover, Wang et al. [56] investigated the effectiveness of predicting the reviews' usefulness with different individual review properties. They showed that various review properties have a varying effectiveness in identifying useful reviews in downstream recommendation tasks. Such results demonstrate the necessity of predicting which review property allows to better predict the reviews' usefulness.

Authors' addresses: Xi Wang, University of Glasgow, Glasgow, UK, x.wang.6@research.gla.ac.uk; Iadh Ounis, University of Glasgow, Glasgow, UK, iadh.ounis@glasgow.ac.uk; Craig Macdonald, University of Glasgow, Glasgow, UK, craig.macdonald@glasgow.ac.uk.

1559-1131/2022/7-ART $15.00
https://doi.org/10.1145/3532859

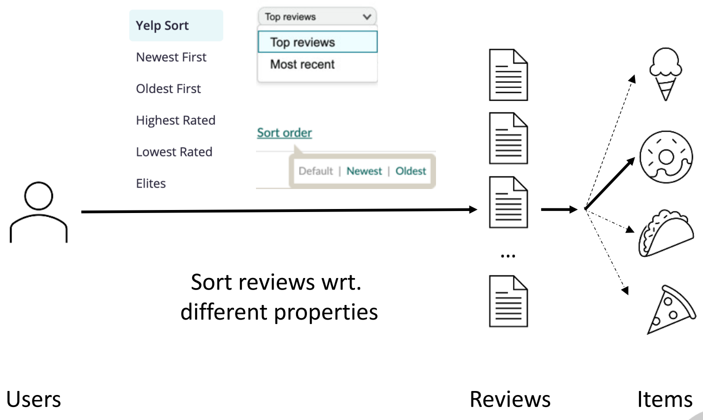


Fig. 1. A pipeline of user-review-item interactions with examples of review sorting interfaces from Yelp (left), Amazon (top right), and Goodreads (bottom right).

In this paper, we also argue that if a recommendation system is able to accurately predict the properties of the useful reviews (e.g. length, recency) that better capture a given user's preferences then the overall recommendation performance will be improved. This argument is motivated by how users interact with items on various e-commerce websites and the review interfaces of various platforms as illustrated in Figure 1. Indeed, reviews are typically sorted according to various available review properties on e-commence websites, and users tend to browse and read many reviews about their targeted items before making their decisions. Our main argument in this paper is that a recommendation model can benefit from capturing the users' preferences from their interactions with different types of reviews, thereby improving its recommendation performance. The users' preferences for reviews exhibiting different properties can be captured from the users' engagement with such reviews [15]. For example, the Users' Adoption of Information (UAoI) framework [51] leverages the users' behaviour to classify users into central and peripheral users. To support their decision making, the central users tend to leverage insightful and in-depth information, while the peripheral users prefer to use simple and straightforward information instead. Such differences among users support the premise of our argument in this paper, namely that it is beneficial for a recommendation model to learn the users' behaviours or preferences from their engagement with different types of reviews. However, to the best of our knowledge, very few studies have investigated the effectiveness of leveraging such users' differences in a recommendation model. One such a study is by Wang et al. [56], who applied a dot-product attention mechanism to learn the importance of review properties for a user. The dot-product attention mechanism linearly combines different parts of the input to compute a representation, so as to enhance the corresponding model's performance [53]. Yet, as mentioned above, following the UAoI framework, different types of users prefer reviews exhibiting different properties. However, the attention mechanism cannot leverage such users' differences to capture the usefulness of reviews.

In this paper, we argue that, instead of using an attention mechanism, it is advantageous to characterise the users' selection of various review properties in estimating the usefulness of reviews as a multi-armed bandit problem. Indeed, the identification of those review properties that are most preferred by a given user can be characterised as an exploration vs. exploitation dilemma [4]. Typically, a contextual bandit search algorithm relies on the context to estimate the rewards for selecting different arms. To capture the users' differences, we propose to leverage the users' posted reviews, since they convey rich preferences information [3, 21], as the context used by the contextual bandit search algorithm, so as to effectively estimate the value of each review property for a given user. By using the users' posted reviews as context, we also address the limitation of the dot-product attention mechanism.

Accordingly, we propose a novel deep neural network (NN) model (called BanditProp) to address the introduced bandit problem and the recommendation task. In particular, BanditProp is a hard parameter sharing-based multi-task learning (MTL) NN model [42]. In this MTL model, each task corresponds to the use of reviews encoded with one particular review property to model the reviews. As a consequence, with k review properties, BanditProp has k sub-networks to extract and generate the feature vectors of reviews corresponding to every used review property. Using the generated feature vectors through the modelling of reviews encoded with k review properties, for a user-item pair, the model generates k scores that estimate the user's preferences on the item given a particular review property. To the best of our knowledge, our study is the first to investigate the use of both MTL and bandit algorithms to estimate the users' preferences on using various review properties for examining the reviews' usefulness, so as to enhance the performance of a recommendation system. Our aim is to capture the users' most preferred review property in identifying a given review's usefulness, so as to maximise the model's recommendation performance.

The main contributions of this paper are as follows:

(1) We propose a new MTL recommendation model, BanditProp, which considers the features extracted from reviews with different review properties as multiple tasks to simultaneously address the recommendation task.

(2) We propose to convert the selection of the properties of the useful reviews that better capture the users' preferences into a bandit problem and integrate various bandit algorithms into BanditProp to predict the users' preferred review properties.

(3) We explore various bandit algorithms and further leverage the users' posted reviews as the contextual information of a newly proposed neural contextual bandit algorithm called ConvBandit. ConvBandit is designed to estimate the payoffs of selecting the k review scores and their corresponding review properties so as to enhance the effectiveness of estimating the users' preferred review properties when making recommendations.

(4) We thoroughly evaluate the effectiveness of our proposed BanditProp model in comparison to one classic and six state-of-the-art recommendation approaches on two public datasets (i.e. the Yelp and Amazon datasets).

(5) We further experimentally demonstrate the effectiveness of ConvBandit in comparison to various existing bandit approaches in addressing our bandit problem.

2 RELATED WORK

In this section, we briefly describe the three bodies of related work, namely review-based, multi-task learning-based and Bandit Problem-based recommendation.

(1) Review-based Recommendation: Users' posted reviews can provide useful evidence for recommendation systems. Indeed, many previous studies [9, 61, 64] have investigated the benefit to recommendation performance of leveraging reviews to capture the users' preferences and items' characteristics. Two typical approaches are the review aspect-based and review body-based recommendation strategies, respectively. The review aspect-based recommenders [5, 19, 28, 30] aim to predict users' preferences on the extracted aspects of the items to estimate the users' preferences on items in a more fine-grained manner. However, their recommendation performances are dependent on the performance of the extracted aspects. The definition of the aspect, the used tool to extract the aspects and the strategy in applying the aspects in a recommendation system have indeed a marked impact on the resulting recommendation performances [11, 16]. On the other hand, the body-based recommenders convert the whole textual content of reviews into latent vectors. Their performances have been improved through benefiting from recent developments in neural language modelling techniques [56].

Therefore, in this paper, we opt to use an approach that considers using a review body-based strategy when addressing the recommendation task. However, different from the aforementioned prior approaches, we consider further enhancing the performances of review body-based recommendation approaches by additionally encoding the properties of the reviews (e.g. length or recency of reviews). In particular, instead of learning the users' preferences from the items' aspects, we leverage the learned users' preferences on the used review properties to capture those useful reviews that enhance personalised recommendation.

(2) Multi-task learning (MTL)-based Recommendation: Many studies have investigated the use of MTL to improve the performances of recommendation models. MTL is essentially a transfer learning mechanism that aims to leverage the shared information between several related tasks to improve a given model's generalisation performances [8]. A number of previous approaches [10, 20, 32] have indeed aimed to aggregate several recommendation-related tasks to improve the recommendation results. Examples included aggregating rating prediction and explainable recommendation tasks [32], context-aware entity search and recommendation [20] or rating prediction on target users as well as their friends [37]. Differently from the existing MTL-based recommendation studies, we propose to consider the learning of the users' preferences and items' characteristics using reviews encoded with their various properties as multiple tasks (i.e. the additional encoding of a given single review property into the review's body corresponds to a single task) so as to construct a novel MTL model.

(3) Bandit Problem-based Recommendation: Many studies [7, 29, 57] proposed to formulate the recommendation task as a bandit problem. In a common setup [29, 45, 57], each user is considered as an agent that selects arms (i.e. items) and evaluates the corresponding payoffs in generating effective recommendations. However, as highlighted by [12], in a multi-armed bandit problem, the difficulty of finding the optimal arm increases as the number of arms increases. As a consequence, under the aforementioned common setup, which considers items as arms, the recommenders' performances will be limited when applied to a large dataset with numerous items. On the other hand, instead of predicting the users' preferences on items, bandit algorithms can also be used to address other exploration vs. exploitation dilemmas - indeed, Javier et al. [46] considered users as arms instead of items in their proposed neighbour selection scheme. Our approach is different from previous bandit-based recommendation approaches in that we consider the small number of review properties as arms in our proposed bandit problem instead of either items or users. Hence, given that the existing bandit-based recommendation approaches do not address the use of review properties, while considering users or items as arms, they do not constitute suitable baselines for our proposed ConvBandit bandit approach. To the best of our knowledge, we are the first study that leverages the users' posted reviews as context to estimate the users' preferences on review properties so as to examine the usefulness of reviews.

3 METHODOLOGY

In this section, we first present the review-based recommendation task and the commonly available properties associated to the reviews (Section 3.1). Next, we describe the review modelling in our proposed BanditProp multi-task learning recommendation model (Section 3.2). We also introduce how BanditProp estimates the users' preferences on the learned features from using different review properties as a bandit problem (Section 3.3). In particular, we describe various bandit algorithms used to address the bandit problem.

3.1 Problem Statement

We focus on addressing a recommendation task, which aims at effectively estimating the users' preferences and recommending items of interest by leveraging the users' posted reviews and their properties. Essentially, a recommendation task involves two main entities: the set of users $U = \{u_1, u_2, ..., u_N\}$ with size N and the set of items $I = \{i_1, i_2, ..., i_M\}$ with size M. To address this task, we leverage the reviews posted by users on items to learn their preferences. For a given user u or item i, there is a set of corresponding reviews C_u or C_i. Furthermore, each review c can be additionally encoded with k review properties $\mathbf{P} = \{P_1, P_2, ..., P_k\}$ to depict the usefulness of reviews from different perspectives. In particular, for a given review property, e.g. P_k, such a review c posted

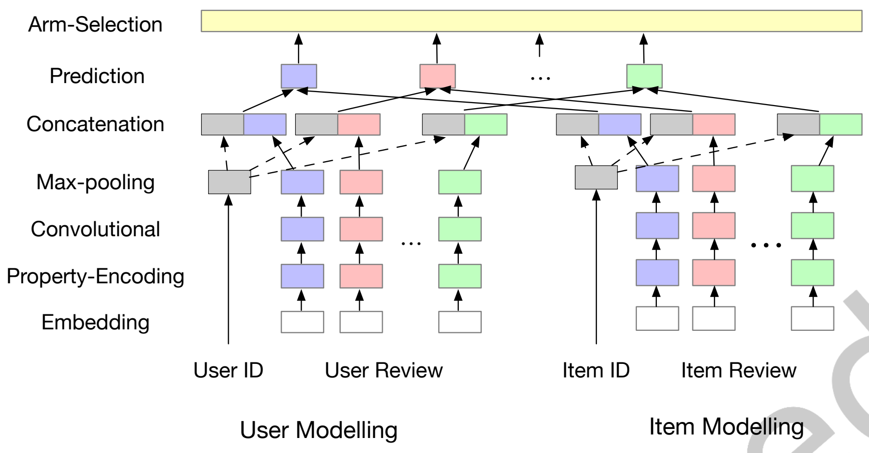


Fig. 2. The structure of the BanditProp model

by user u or on item i, has an associated property score $P^u_{k,c}$ or $P^i_{k,c}$. Note that we map the property scores into scalars in the range of [0..1] through an adequate function in Section 3.2.

3.2 Review Modelling in BanditProp

To address the introduced recommendation task, we propose a neural network model (i.e. BanditProp) that learns the users' preferences on items from reviews additionally encoded with their properties. Figure 2 presents the architecture of BanditProp. Such architecture involves two duplicated neural networks to learn the features of the user and the item, respectively. We use the user modelling neural network as an example to illustrate the network structure. In the BanditProp's input data, for a given user, we have their ID and associated reviews. Each type of input is learned by using a separate network. To use the user ID, we leverage the one-hot encoding technique to encode the general features of a given user. On the other hand, in the review modelling network, we construct k parallel neural networks to model the reviews, which are additionally encoded with k different review properties. This architecture can also be seen as a multi-task learning network that uses the shared user/item embeddings among k different review property modelling networks. In the following, we further describe the details of the review modelling neural networks.

3.2.1 Review Modelling Networks. In the review modelling networks of BanditProp, there are k parallel neural networks that learn features from the reviews additionally encoded with k different review properties. We follow [56] and use six commonly available review properties (age, length, rating, Polar_Senti, Helpful and Prob_Helpful that introduced in Table 1) to describe the reviews from different aspects. The network that models the reviews additionally encoded with each of the six review properties consists of five layers from the embedding to the concatenation layer. In the following, since the parallel review property modelling networks have the same structure, we use the network that focuses on the use of the length property to illustrate the five layers of the networks.

First, in the embedding layer, we convert each review into the embedding vector X by using the pre-trained BERT model [13], which is a widely used language modelling approach. Next, in the property-encoding layer, we embed the scores of the length property of reviews into the converted review embedding vectors. For example, for a given review c posted by user u and the length property P_k, we calculate the length property score $p^u_{k,c}$ for

Properties	**Descriptions**
Recency	The min-max normalised number of days d since a review has been posted.
Length	The min-max normalise number of words that are included in a review.
Rating	The min-max normalised rating associated with the review.
Polar_Senti	A CNN classifier [24] is used to calculate the corresponding probabilities of the positive reviews being actually positive or the negative reviews being negative.
Helpful	The min-max normalised number of helpful votes for a given review.
Prob_Helpful	The estimated probability of a review being helpful by using a review helpfulness classifier [55].

Table 1. Summary of the review properties.

review c as described in Table 1. This review property score indicates the importance of review c according to the length property P_k. In particular, according to the description of the length property, a longer review will obtain a higher length property score than a shorter review. Afterwards, the integration of the review property scores to the review embedding vector X is done by the dot-product operation, which is formalised as follows:

$$O_{u,P_k} = [X_1 \cdot p^u_{k,1}, X_2 \cdot p^u_{k,2}, ..., X_{|C_u|} \cdot p^u_{k,|C_u|}] \tag{1}$$

where P_k refers to the k_{th} property and the property scores are applied to user u's $|C_u|$ reviews. As a consequence, BanditProp can extract and learn features from the score-embedded latent vectors for each of the given review property (e.g. the length property). Moreover, by using k review properties, BanditProp can comprehensively capture features from reviews using different review properties. After the property encoding layer, the review property-encoded latent vector O_{u,P_k} is used as input to the convolutional layer and then the max-pooling operation is applied in the next layer. Note that, in the convolutional layer, given the textual nature of the users' posted reviews, we apply the one-dimensional convolutional operator to capture the features from the reviews. Next, we concatenate the learned embedding vector from the review modelling network and the embedding of the corresponding user into a latent vector. This latent vector indicates the learned feature from the user by looking at both their general preferences on items and their learned preferences from the reviews using a specific review property (e.g. length of reviews). Afterwards, for a given user u or item i with k review properties, we obtain k resulting latent vectors ($O'_{u,P}$ and $O'_{i,P}$, $P \in \mathbf{P}$) for both user u and item i. We denote the parameters included in these review modelling networks as ϕ where ϕ_{P_k} indicates the parameters in the network that models the k_{th} review property P_k.

Next, for the review modelling network of the reviews additionally encoded with a given review property P_k, we apply the dot-product operation on the learned latent vectors between user u and item i (i.e. $o''_{u,i,k} = O'_{u,P_k} \cdot O'_{i,P_k}$) to conduct users' preference estimation. This results in k scores that estimate the user's preferences on items for each of the k review properties.

3.3 Arm Selection Layer in BanditProp

After the prediction layer, we have an arm-selection layer to selectively use the computed scores $o''_{u,i,k}$ from the modelling of k review properties. In this arm selection layer, we propose to consider the selection of the

computed scores $o''_{U,I,k}$ of all users U and items I on the k review properties from the review modelling networks as a bandit problem. The target is to enhance BanditProp's performances by selecting the scores $o''_{U,I,k}$ to estimate users' preferences on items. We formulate such a bandit problem as follows:
• The arms $\mathcal{A}$ correspond to the k review properties $\mathbf{P}$. The selection of different arms is akin to the use of the corresponding review properties.
• The reward $\mathcal{R}$ is calculated based on the users' historical interactions similar to [31, 63]. The reward follows a multinomial distribution and is defined as the ratio of times that the model ranks the positive item higher than the randomly sampled negative item in Γ positive-negative item pairs by using the feature score of a given review property [1]. For example, after conducting an arm selection, if the model ranks the positive item higher than the negative item six times in ten positive-negative item pairs, the reward is 0.6. The resulting rewards' values are the elements of a set $\{0, \frac{1}{\Gamma}, ..., 1\}$. By maximising the expected rewards, the model can better distinguish between the positive and negative items so as to generate the top items for each user. Note that as argued in [2, 40], the multinomial rewards can be seen as more detailed and informative than the binary rewards. Indeed, in Section 6, we experimentally demonstrate the advantage of using multinomial rewards over binary rewards.
• The state $\mathcal{S}$ is the review-based feature vector of a given user. According to the UAoI framework [15, 51], there is a relationship between the users' type and their preferred properties of reviews. In particular, in [56], the authors experimentally showed that there is a potential correlation between the properties of the users' posted reviews and the users' types. Therefore, we propose to leverage the users' posted reviews as context to approximate the users' type so as to support the prediction of the users' preferred review properties. Note that, the contextual bandit algorithms extend the multi-armed bandit algorithms by leveraging the state $\mathcal{S}$ to construct their algorithms and predict the users' preferences on the review properties.

To effectively address the bandit problem within a deep neural network, we propose a customised learning algorithm, which is described in Algorithm 1. In Algorithm 1, we introduce how we separately train the parallel review modelling networks and the bandit approaches. In particular, the BPR loss function [39] is leveraged to update the parameters of each of the review modelling networks that encapsulate the selected review properties. Moreover, to update the parameters of the neural networks, we apply a freezing mechanism, such that the parameters of the review modelling network that encapsulates the selected review property are updated, while the networks that address other review properties are frozen, i.e. unchanged. Note that we use the greedy search as an example in Algorithm 1, but it can be replaced with any other bandit algorithms.

For the bandit approaches that we use as surrogates to the default greedy search algorithm in Algorithm 1, we consider two types of bandit algorithms, namely multi-armed bandit (MAB) and neural contextual bandit algorithms, to address our introduced bandit problem. We now introduce these algorithms:
(1) Multi-Armed Bandit Approaches In general, a multi-armed bandit can be described as a tuple $\langle \mathcal{A}, \mathcal{R} \rangle$ that indicates actions and rewards, respectively. In this paper, we include three types of MAB algorithms, namely greedy search-based [26], UCB-based [22] and Thompson Sampling (TS)-based [43] algorithms. Greedy search-based encapsulate variants such as ϵ-greedy search and decayed ϵ-greedy search [26]. They exploit the averaged historical accumulated payoffs of selecting the corresponding review properties. The corresponding arm selection strategy is formulated as follows:

$$a_t^* = \arg\max_{a \in \mathcal{A}} \left[\frac{1}{N(a)} \sum_{t=0}^{T} r_t(a_t = a) \right] \tag{2}$$

However, the greedy search-based algorithms differ in conducting the exploration of the arm selections. The greedy search solely relies on the exploitation of historical payoffs (i.e. Equation (2)). Note that, we later use the greedy search as the default bandit algorithm in our BanditProp model. Meanwhile, the ϵ-greedy search [26] uses

[1]An item is positive if it has been interacted with by the corresponding user, and it is negative otherwise.

Algorithm 1: Model learning for user u by using the default greedy search for selecting arms (i.e. the set of review properties $\mathbf{P}$). This can be replaced by any other general bandit algorithm (including a contextual bandit algorithm).

```
1  $R_{u,\mathbf{P}} \leftarrow 0$;                                  // Accumulated reward ;
2  $N_{u,\mathbf{P}} \leftarrow 0$;                                  // Count the selection of arms ;
3  $n \leftarrow 0$;
4  repeat
5      // neural network training
6      for $P_k \in \boldsymbol{P}$ do
7          $r_{u,P_k} \leftarrow$ the reward by using property $P_k$;
8          $\mathcal{L}_{u,P_k} \leftarrow \sum_{u,i^+,i^- \in D} ln[\sigma(o''_{u,i^+,P_k} - o''_{u,i^-,P_k})]$;    // calculate the BPR loss [39] using $P_k$;
9          // $D := \{(u,i^+,i^-)|i^+ \in I_u^+ \wedge i^- \in I \setminus I_u^+\}$;
10         Freeze the gradients of $\phi_{\mathbf{P}} - \phi_{P_k}$;
11         Update $\phi_{P_k}$ according to $\mathcal{L}_{u,P_k}$;
12     end
13     // bandit search illustrated with the default greedy search
14     for $i \leftarrow 0$ to $n$ do
15         $P \leftarrow \arg\max(R_{u,P_k}/N_{u,P_k})$;                 // Equation (2);
16         $R_{u,P_k} \leftarrow R_{u,P_k} + r_{u,P_k}$;
17         $N_{u,P_k} \leftarrow N_{u,P_k} + 1$;
18     end
19     $n \leftarrow n + 1$;
20 until Convergence;
```

a fixed exploration factor ϵ to indicate the probability of selecting random arms. On the other hand, the decayed ϵ-greedy (i.e. DE-greedy) search [26] downgrades the value of the exploration factor ϵ over time. In this paper, we use the following decay function: $\epsilon_t = \frac{1}{log(t)}$. Note that by using different decay functions, we can have various DE-greedy search algorithms, which have different rates of decline in the value of ϵ. However, we have experimentally found that there is no significant difference between the performances of these variants when addressing our specific bandit problem. Hence, due to space constraints, we only focus on the DE-greedy search approach.

Apart from the greedy search-based algorithms, we consider another family of bandit algorithms (i.e. UCB search [22]). Different from the greedy search algorithms, the UCB search algorithms estimate the values of arms by capturing the upper confidence bounds of selecting arms. The upper confidence bound $\hat{U}$ is defined as the uncertainty in the payoffs of selecting an arm. If an arm has been frequently selected, it has a lower uncertainty in the payoffs than selecting other arms. Therefore, the system tends to select arms with higher uncertainty in the payoffs to capture the values of selecting these arms. We consider two UCB-based algorithms (i.e. UCB1 and Bayes-UCB [22]). UCB1 can be formulated as follows:

$$a_t^* = \arg\max_{a \in \mathcal{A}} \left[\frac{1}{N(a)} \sum_{t=0}^{T} r_t(a_t = a) + \sqrt{\frac{2\log(T)}{N(a)}} \right] \tag{3}$$

UCB1 estimates the value of arms according to the averaged accumulated rewards (left part of Equation (3)) like the greedy search-based approaches and the uncertainty score (right part of Equation (3)). The Bayes-UCB algorithm leverages the prior knowledge of the arms' values to estimate the posterior values of arms. We assume that the prior knowledge of arms can be modelled by the beta distribution, which includes two shape parameters $\alpha(a)$ and $\beta(a)$. At each iteration, if arm a is selected, $\alpha(a)$ and $\beta(a)$ are updated by $\alpha_{t+1}(a) = \alpha_t(a) + r_{t,a}$ and $\beta_{t+1}(a) = \beta_t(a) + (1 - r_{t,a})$, respectively. The Bayes-UCB estimates the values of arms as:

$$a_t^* = \arg\max_{a \in \mathcal{A}} \left[\frac{\alpha_t(a)}{\alpha_t(a) + \beta_t(a)} + c\sigma\Big(\alpha_t(a), \beta_t(a)\Big) \right] \tag{4}$$

where $c\sigma(\alpha_t(a), \beta_t(a))$ is the standard deviation of the beta distribution and c determines the size of the confidence interval.

Moreover, similar to the Bayes-UCB strategy, we also consider the TS algorithm, which also relies on the beta distribution as the prior knowledge. However, the Bayes-UCB and TS algorithms are different in estimating the values of selecting arms. The TS algorithm follows the idea of probability matching [48], which is a decision strategy that relies on the defined prior distribution. At each time step, the system samples the posterior probability of each arm and selects the arm that maximises the expected reward $\mathbb{E}$. This process can be interpreted with the following equation:

$$a_t^* = \mathbb{E}\left[\arg\max_{a \in \mathcal{A}} \Big(a | Beta(\alpha_t(a), \beta_t(a)) \Big) \right] \tag{5}$$

Furthermore, given that we are using multinomial rewards to examine the value of selecting arms, we also consider Multinomial TS (Mul-TS) [40], a state-of-the-art bandit algorithm that has been designed to address bandit problems using multinomial rewards. At each round of the bandit selection, for each arm a, Mul-TS leverages the Dirichlet distributions $Dir(\beta_0^a, \beta_1^a, .., \beta_\Gamma^a)$ with dimension $\Gamma + 1$ to generate samples L_a. Essentially, L_a refers to the likelihood of obtaining specific values of multinomial rewards after selecting a corresponding arm. After that, the arm selection can be described by the following equation: $a_t^* = \arg\max_{a \in \mathcal{A}} \left[(0, \frac{1}{\Gamma}, \frac{2}{\Gamma}, ..., 1)^\top L_a \right]$. Then, if a reward $\frac{\gamma}{\Gamma}$ is observed, the parameters of the used Dirichlet distribution are updated with $\beta_\gamma^{a_t^*} := \beta_\gamma^{a_t^*} + 1$. On the other hand, by comparing the implementations of the considered MAB approaches in this paper, we observe that the greedy search-based approaches and the UCB1 strategy rely on the accumulated historical rewards to estimate the values of arms, while the Bayes-UCB, TS and Mul-TS algorithms focus on constructing functional estimators to predict the values of arms. Given such a difference, the Bayes-UCB and TS-based approaches require an additional effort in learning the estimators, which makes it challenging for them to effectively estimate the values of arms. However, by comparing TS and Mul-TS, we expect that Mul-TS should outperform TS by capturing the multinomial nature of the reward distribution.

(2) Neural Contextual Bandit Approaches

Neural contextual bandit algorithms extend the MAB algorithms by leveraging the state information $\mathcal{S}$ to support the estimation of the arms' values. Recently proposed neural contextual bandits [41, 60] are considered to be the current state-of-the-art and generally have a good performance in tasks where exploitation vs exploration must be addressed. Therefore, in this paper, we consider the neural contextual bandit approaches, which model the arms, states and rewards (i.e. $\langle \mathcal{A}, \mathcal{S}, \mathcal{R} \rangle$) through a neural network. The action-value function is formulated by the expected reward, which considers both the state and action (i.e. $Q(s, a) = \mathbb{E}(r|s, a)$).

As mentioned above, we propose to use the review-based feature vectors X of the users as the state $\mathcal{S}$. We first construct a linear neural contextual bandit algorithm (LinBandit) [65]. It approximates the value of arms with the following equation:

$$Approx_{LinBandit}^a = W_L^a \sigma\Big(W_{L-1}^a \sigma(...\sigma(W_1[X \oplus X^a])) \Big) \tag{6}$$

where X indicates the feature vector of the reviews posted by a given user. X^a is the review feature vector encoded with the information of arm a. The latter corresponds to the feature vector O_{u,P_k} within Equation (1) with review property P. $\oplus$ refers to the residual connector. In particular, LinBandit assumes that the values of arms can be linearly approximated from the corresponding context or state $\mathcal{S}$. Next, we observe that the review feature vectors X of the arms are extracted from the textual documents. Given that the convolutional operation can effectively capture the local features of documents by modelling local receptive fields and shared weights [23, 24], we postulate that the convolutional operation can outperform the linear operation in better capturing information in the review-based feature vectors X. Therefore, in this paper, we propose another neural contextual bandit algorithm (i.e. ConvBandit) that uses the convolutional operation-based approximation function to estimate the reward of selecting various arms. Its reward approximation function is:

$$Approx^{a}_{ConvBandit} = \sigma\left[f^{a}_{1} * (g^{a}_{1}(\sigma(f^{a}_{0} * g^{a}_{0}[X \oplus X^{a}])))\right] \tag{7}$$

where $\sigma(.)$ and $*$ are the activation function and the convolutional operation, respectively. Note that for both LinBandit and ConvBandit, we first train the approximation functions by minimising Mean Squared Error (MSE) between their predicted values for the given arms and the averaged historical rewards (i.e. $\frac{1}{N(a)}\sum_{t=0}^{T} r_t(a_t = a)$). Next, similar to NeuralUCB [65], while using the neural approaches for the arm value's estimation, we also leverage the upper confidence bound as in Equation (3) to implement the arm exploration. Therefore, the arm selection of both LinBandit and ConvBandit are implemented as follows:

$$a^{*}_{t} = \arg\max_{a \in \mathcal{A}}(Approx^{a} + \sqrt{\frac{2\log(T)}{N(a)}}) \tag{8}$$

4 EXPERIMENTAL SETUP

We first introduce our research questions, which examine the effectiveness of BanditProp. Next, we describe the two used datasets as well as the baseline approaches. Then, we describe in detail the settings used to train and deploy both BanditProp and the baseline approaches. In order to evaluate our proposed BanditProp model, we aim to answer the following research questions:

RQ1: Does BanditProp, which uses a multi-task learning framework with a default greedy search algorithm outperform existing state-of-the-art baseline approaches on the two used datasets?

RQ2: Which multi-armed bandit algorithm performs the best when applied to the BanditProp model on the two used datasets?

RQ3: Do the neural contextual bandit algorithms outperform the multi-armed bandit ones when applied to the BanditProp model?

4.1 Datasets

We follow [56] and use two real-world datasets (i.e. the Yelp (round 13)[2] and the Amazon[3] [18] datasets). They have been used in many review-based recommendation studies [35, 47]. Following the dataset filtering strategy in [56], for the Yelp dataset, we use the users' reviews on the top category of venues (i.e. restaurant). Moreover, for the Amazon dataset, we include the users' reviews from six categories[4]. These two datasets contain rich review properties (e.g. timestamp and helpfulness vote), allowing to capture the users' preferences on the reviews properties under different system interfaces and conditions. We also filter the datasets to alleviate the datasets' sparseness problem as in [44, 56]. The filtering operation results in all the users and items in both datasets having at least 5 associated reviews. After the filtering step, the resulting Yelp and Amazon datasets, have 47k users / 16k items / 551k reviews and 26k users / 16k items / 285k reviews, respectively. In order to evaluate the performances of our proposed approaches and baselines on the two datasets, we further randomly split both datasets into 80% training,

[2]https://www.yelp.com/dataset

[3]http://jmcauley.ucsd.edu/data/amazon/

[4]'amazon instant video', 'automotive', 'grocery and gourmet food', 'musical instruments', 'office products' and 'patio lawn and garden'

10% validation and 10% testing sets. The recommenders' performances are measured on the testing set using the Precision (P@1 and P@5), Recall (R@10), Mean Average Precision (MAP) and Normalised Discounted Cumulative Gains (NDCG@10) metrics. Note that'@#' indicates the cutoff position at '#' for the corresponding metric.

4.2 Baseline Approaches

We evaluate our proposed BanditProp model on the two used datasets in comparison to six strong state-of-the-art baselines, namely:

(1) *BPR-MF [39]* is a classical recommendation model, which estimates the users' preferences on items by leveraging the interactions between the decomposed user-item feature vectors.
(2) *DREAM [59]* is a sequential recommender, which leverages the recency of reviews and estimates the users' preferences on items by considering sequential dependencies in user-item interactions.
(3) *CASER [52]* is a state-of-the-art sequential recommender. It uses the convolutional neural network to model the sequential interaction between a given user and the items.
(4) *JRL [61]* is a heterogeneous recommender that estimates the users' preferences from various types of users' explicit feedback (e.g. reviews and ratings). We implement the JRL model by only using the users' posted reviews.
(5) *DeepCoNN* [64] is a popular review-based recommendation baseline [9, 55]. It uses the interaction between the review feature vectors of the user-item pairs to estimate the users' preferences.
(6) *NARRE* [9] is a state-of-the-art review-based recommender. It ranks the users' interest in items, using both the one-hot embedding and the review feature vectors for a given user-item pair. It also leverages an attention mechanism to observe useful review features.
(7) *RPRS [56]* is a recent review-based recommender, which builds upon the users' adoption of information framework. It uses various review properties to describe the reviews from different perspectives. Then, it models the users' interactions with a given type of reviews so as to estimate the users' tendency in using the review data and to improve the recommendation accuracy.

4.3 Models' Settings

We deploy our proposed BanditProp model[5] and the baseline approaches using the PyTorch framework [38]. For the BanditProp's setup, we leverage the pre-trained BERT model as in [56] and use the representation vector of the '[CLS]' token to convert the review tokens into a 768-sized latent vector. We apply the pre-trained BERT model since BERT and other recent neural language models have been shown to be more effective than classic text feature representation techniques [49, 54, 62] (e.g. bag-of-words [62]) in many text-based tasks, such as document retrieval [33, 54] and text classification [36]. Similarly, for the DeepCoNN, NARRE and RPRS baseline approaches, we also use BERT to compute the review embeddings used as input to these baseline models. Note that the DeepCoNN and NARRE models were originally designed to address the rating prediction task. Hence, in this paper we convert them into ranking-based approaches. To do so, we change their training to use the BPR ranking method [39] by leveraging the calculated scores of the users' preferences on items as input to the BPR loss function. Next, in the property-encoding layer, we compute the scores of the review properties as described in Table 1. For the 'Polar_Senti' and 'Prop_Helpful', we use a trained CNN-based classifier and a review helpfulness classifier, namely NCWS [55], which is a state-of-the-art review helpfulness classifier, to estimate the probabilities of the reviews being sentimentally polarised (i.e. strongly positive or negative) or being helpful. Note that, for the CNN-based sentiment classifier, we train it on 100,000 reviews, which include half positive and half negative reviews that are sampled from the Yelp Challenge dataset round 12. As for the NCWS classifier, we follow the experimental setup in [55] by training two classifiers on the reviews from the Yelp Challenge dataset round 12 and the Amazon

[5]We will release all our code upon the acceptance of this paper.

kindle datasets, respectively. Then, these two classifiers are used to predict the helpfulness property of each of the collected reviews from the same website. In particular, in the ground truth, we consider one review as helpful if it has at least one helpful vote. Furthermore, for the bandit problem tackled in this paper, to generate the multinomial rewards and speed up the training, we follow [58] and set the number of randomly sampled positive-negative item pairs Γ to 10. For the ϵ-greedy search algorithm, as in [14], we set the ϵ to 0.1 to indicate the probability of selecting random arms. On the other hand, in the Bayesian UCB algorithm of Equation (4), we set the value of c to 3 to control the size of the confidence interval. For training the BanditProp model, we apply the early-stopping strategy with a maximum 200 epochs and use the Adam optimiser [25] with a $1e^{-4}$ learning rate. The same strategy is also used to train the baseline approaches. In addition, as we argued in Section 1, the performance of a review-based recommendation system can benefit from the accurate prediction of the properties of the useful reviews that better capture the users' preferences and the items' attributes. Therefore, to evaluate different strategies for estimating the properties of useful reviews, we directly compare the recommendation performances of BanditProp with different bandit algorithms that learn to selectively use various review properties to estimate the usefulness of reviews.

Table 2. Recommendation performances. 1/2/3/4/5 denote a significant difference w.r.t. BPR-MF, DREAM, NARRE, RPRS and Bandit-Prop, respectively, on NDCG@10 according to a paired t-test with the Tukey HSD correction ($p < 0.05$).

Dataset			**Amazon**					**Yelp**				
Model			**P@1**	**P@5**	**R@10**	**MAP**	**NDCG@10**	**P@1**	**P@5**	**R@10**	**MAP**	**NDCG@10**
1	**BPR-MF**	2, 3, 4, 5	0.00538	0.00431	0.03013	0.01118	0.01633	0.01014	0.00656	0.03919	0.01455	0.02176
2	**DREAM**	1, 3, 4, 5	0.00523	0.00358	0.02914	0.01067	0.01506	0.00838	0.00723	0.04692	0.01555	0.02292
–	**CASER**	1, 2, 3, 4, 5	0.00934	0.00720	0.04991	0.02392	0.03150	0.01112	0.00937	0.05710	0.02291	0.03333
–	**DeepCoNN**	2, 3, 4, 5	0.00534	0.00451	0.03431	0.01196	0.01575	0.00549	0.00310	0.01739	0.00726	0.01154
–	**JRL**	1, 2, 3, 4, 5	0.00417	0.00384	0.03102	0.00923	0.01238	0.00437	0.00294	0.01355	0.00615	0.00928
3	**NARRE**	1, 2, 4, 5	0.01753	0.01024	0.05885	0.02797	0.03654	0.01375	0.00994	0.06057	0.02289	0.03324
4	**RPRS**	1, 2, 3, 5	0.02238	0.01270	0.08657	0.03784	0.04966	0.01618	0.01221	0.07611	0.02713	0.04205
5	**BanditProp**	1, 2, 3, 4	**0.02534**	**0.01423**	**0.09481**	**0.04197**	**0.05536**	**0.01760**	**0.01326**	**0.08144**	**0.02952**	**0.04486**

5 RESULTS ANALYSIS

In the following, we evaluate and analyse the performance of the proposed BanditProp by answering our three research questions.

5.1 RQ1: BanditProp Performance Evaluation

Our first research question aims to determine if our BanditProp model – with a default greedy search algorithm – can outperform the existing state-of-the-art recommendation approaches. Table 2 presents the performances of BanditProp with a default greedy search algorithm in comparison to 7 baselines. Following [34, 50], statistical significance differences among the various approaches on the NDCG@10 metric are assessed using the paired t-test as well as the Tukey HSD correction test (p-value < 0.05). Among the baseline approaches, we observe that the DeepCoNN and JRL models, which solely rely on the reviews' text as input, provide less competitive recommendation performances than other approaches. Meanwhile, by using both the user/item embeddings and the review text as input, NARRE can significantly outperform BPR-MF, DeepCoNN and JRL on the two datasets. Similarly, we observe that the performances of sequential recommenders can also be improved by using the user/item embeddings. For example, the CASER model, which uses the user/item embeddings, significantly outperforms DREAM, which solely relies on the sequential pattern of user-item interactions. This result supports the use of the user/item embeddings in BanditProp. Next, we observe that RPRS significantly outperforms all the baselines on two datasets. This result demonstrates that RPRS is effective at capturing the users' preferences by leveraging various review properties, and predicting the users' interests in items. We now turn our attention to the performance of our proposed BanditProp model. The experimental results in Table 2 show that just by using the default greedy search algorithm, BanditProp significantly outperforms all the baseline approaches including RPRS. Given how RPRS and BanditProp differ in their use of review properties, these results also support our argument for developing a model encompassing multi-task learning and a bandit search strategy that captures the users' preferences from their interaction with different types of reviews as we advocated in Section 1 based on the existing review sorting interfaces of various platforms (c.f. Figure 1). Therefore, in answering RQ1, we conclude that our proposed BanditProp model can significantly outperform the existing state-of-the-art recommendation baselines. In particular, we showed the effectiveness of two main components in the BanditProp model, namely (1) the user/item embeddings; (2) the multi-task learning component and the corresponding bandit search strategy-based model architecture, as motivated the need to capture users' behaviour with the review sorting interfaces of various platforms.

Table 3. Performances of the recommendation approaches. ↑ denotes a significant difference w.r.t. RPRS on all ranking metrics according to a paired t-test with the Tukey HSD multiple testing correction ($p < 0.05$). Similarly ○ and ●, respectively, denote significant differences using the same test w.r.t. BanditProp using the default greedy bandit algorithm and ConvBandit.

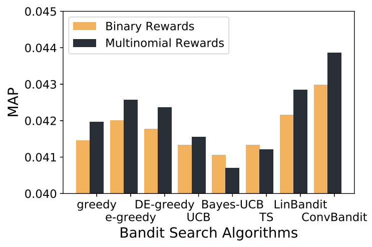
Dataset		**Amazon**					**Yelp**				
Model		**P@1**	**P@5**	**R@10**	**MAP**	**NDCG@10**	**P@1**	**P@5**	**R@10**	**MAP**	**NDCG@10**
–	**RPRS**	0.02238○●	0.01270○●	0.08657○●	0.03784○●	0.04966○●	0.01618○●	0.01221○●	0.07611○●	0.02713○●	0.04205○●
Multi-armed Bandit Algorithms											
↑	**greedy**	0.02534●	0.01423●	0.09481●	0.04197●	0.05536●	0.01760●	0.01326●	0.08144●	0.02952●	0.04486●
↑	**ϵ-greedy**	0.02545●	0.01447●	0.09521●	0.04258○●	0.05603○●	0.01773●	0.01325●	0.08144●	0.02936●	0.04431●
↑	**DE-greedy**	0.02534●	0.01437●	0.09413○●	0.04237●	0.05568●	0.01773●	0.01345●	0.08146●	0.02957●	0.04490●
↑	**UCB**	0.02484●	0.01428●	0.09315○●	0.04156●	0.05488○●	0.01680○●	0.01267○●	0.07943○●	0.02800○●	0.04281○●
–	**Bayes-UCB**	0.02491●	0.01404●	0.09135○●	0.04070○●	0.05416○●	0.01656○●	0.01252○●	0.07859○●	0.02790○●	0.04227○●
–	**TS**	0.02461○●	0.01386○●	0.09193○●	0.04122○●	0.05448○●	0.01637○●	0.01229○●	0.07472○●	0.02725○●	0.04189○●
↑	**Mul-TS**	0.02526●	0.01423●	0.09298○●	0.04190●	0.05473○●	0.01728●	0.01302●	0.07878○●	0.02849○●	0.04311○●
Neural Contextual Bandit Algorithms											
↑	**LinBandit**	0.02572●	0.01476○●	0.09553○●	0.04285○●	0.05603○●	0.01781●	0.01344●	0.08121●	0.02946●	0.04437●
↑	**ConvBandit**	**0.02637**○	**0.01558**○	**0.09675**○	**0.04386**○	**0.05723**○	**0.01825**○	**0.01408**○	**0.08183**○	**0.03015**○	**0.04539**○

5.2 RQ2: Effectiveness of the MAB Algorithms

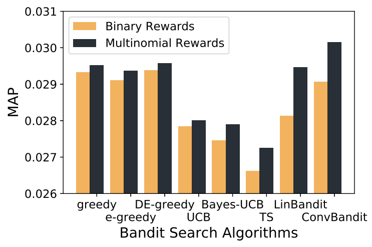
To answer RQ2, we investigate the effectiveness of using various multi-armed bandit (MAB) algorithms when applied to our proposed BanditProp model. In Section 3.3, we introduced three types of MAB algorithms, namely the greedy search-based, UCB-based and TS-based approaches. Recall that the main difference between these MAB algorithms consists in whether a bandit algorithm uses the averaged historical rewards (the greedy search-based and the UCB1 algorithms) or a functional estimator (the Bayes-UCB, TS and Mul-TS algorithms) to approximate the value of arms. Table 3 presents the performances of using these three types of bandit algorithms in our BanditProp model. Note that the algorithm denoted by 'greedy' in Table 3 corresponds to the default bandit greedy search strategy used in the BanditProp model in Table 2. First, we investigate the effectiveness of using the greedy search-based algorithms. We observe that the ϵ-greedy search approach is the best performing among the greedy-search approaches on the Amazon dataset. It also significantly outperforms the default greedy search on MAP and NDCG@10. On the Yelp dataset, the DE-greedy search algorithm outperforms the greedy and ϵ-greedy search algorithms. Such results demonstrate the positive impact of introducing the exploration component into the default greedy search algorithm on the recommendation performances. On the other hand, we also observe that all the greedy search approaches as well as the UCB1 algorithm, which rely on the accumulated historical rewards, significantly outperform the strongest RPRS baseline on all ranking metrics. These results indicate the benefits of leveraging the accumulated historical rewards for estimating the values of arms, so as to make effective recommendations. Next, we evaluate the performances of using the Bayes-UCB and TS-based approaches (namely TS and Mul-TS), which use functional estimators to approximate the values of arms, within the BanditProp model. The reported results in Table 3 show that the Bayes-UCB and TS approaches are significantly outperformed by the default greedy search algorithm on most of the ranking metrics. This observation indicates that it is difficult for the functional estimator-based algorithms to effectively approximate the values of arms and improve the recommendation performances when used within BanditProp. However, by comparing the performances of TS and Mul-TS, the obtained results in Table 3 show that Mul-TS outperforms TS while achieving a significantly indistinguishable performance in comparison to the default greedy search algorithm on many ranking metrics. This indicates the advantage of modelling the multinomial rewards with a Dirichlet distribution instead of the beta distribution used within TS. Therefore, in answering RQ2, we conclude that the MAB algorithms that rely on the accumulated rewards to approximate the values of arms (namely the greedy-based and UCB1 approaches) are more effective than the approaches that use functional estimators (namely the Bayes-UCB and TS-based approaches) in predicting the arms' values within our proposed BanditProp model. Moreover, we also observe the benefit of using a bandit approach (namely Mul-TS) that models the multinomial rewards with the Dirichlet distribution within BanditProp. Thus far, the reported experimental results show that by considering various multi-armed bandit algorithms that selectively use different review properties, so as to capture the usefulness of reviews, BanditProp can indeed yield a significantly improved recommendation performance. However, the impact of using the state information (c.f. Section 3.3) on the selection of the review properties and consequently on the recommendation performance remains to be investigated. Therefore, next, we present and discuss the performances of applying the contextual bandit algorithms to our proposed BanditProp model.

5.3 RQ3: Using the Contextual Information

To answer RQ3, we investigate whether using the state information $\mathcal{S}$, a neural contextual bandit algorithm that encapsulates the features extracted from the reviews (an information not conveyed by the MAB algorithms), can lead to effective recommendation performances along BanditProp. In particular, we examine the performance of our proposed ConvBandit algorithm in comparison to the LinBandit algorithm, which is a state-of-the-art neural contextual bandit algorithm. Recall that in Section 3.3, ConvBandit differs from LinBandit, that used a linear operation, by using the convolutional operation to model the state information $\mathcal{S}$. Table 3 reports the results on the two used datasets. We observe that when used in BanditProp, ConvBandit significantly outperforms LinBandit as well as the multi-armed bandit (MAB) algorithms on all ranking metrics across the two used datasets. The difference between the neural contextual bandit approaches (e.g. ConvBandit and LinBandit) and the MAB algorithms is that the neural contextual bandit algorithms consider the action, state and reward instead of only the action and reward used by the MAB algorithms (as denoted in Section 3.3). The observed effectiveness of ConvBandit indicates the usefulness, for a bandit approach, to leverage the feature vectors of reviews as context to approximate the values of arms within BanditProp. The improved performance using ConvBandit compared to that using the default greedy search algorithm also demonstrates the advantage of considering additional components (i.e. the state information as well as the exploration mechanism introduced in Equation (3)), instead of solely relying on the accumulated rewards. In particular, as previously described in Section 3.3, the Bayes-UCB, TS-based, LinBandit and ConvBandit approaches all use functional estimators to approximate the values of arms. The significantly better performance achieved by ConvBandit, in comparison to both the Bayes-UCB and TS-based approaches, further demonstrates that it is useful to leverage the feature embeddings of reviews as context when estimating the arms' values to improve the BanditProp's performance. Therefore, in answering RQ3, we conclude that ConvBandit significantly outperforms the existing state-of-the-art LinBandit and Mul-TS bandit algorithms as well as various classic bandit algorithms on both the Amazon and Yelp datasets when applied to our proposed BanditProp model. In addition, the significantly better performance of ConvBandit, in comparison



(a) Amazon dataset



(b) Yelp Dataset

Fig. 3. Effectiveness comparison between using the binary rewards or the multinomial rewards on the two datasets.

to other functional estimator-based bandit approaches without the context information, further supports the added value of the used context information within ConvBandit.

6 MULTINOMIAL VERSUS BINARY REWARDS

The strategy used to leverage the collected rewards after selecting the arms plays an important role in addressing a bandit problem [1]. In Section 3.3, we used the multinomial rewards in various bandit algorithms. The multinomial rewards are computed according to the ratios of the correctly ranked positive-negative item pairs. Alternatively, we can also compute binary rewards by applying a conditional function to the multinomial rewards (i.e. the reward is 1 if the multinomial reward is greater than 0.5, 0 otherwise). For the used bandit approaches, we argued that using the multinomial rewards can better capture the values of selecting different arms than the binary rewards. However, in the existing literature, the binary rewards are still frequently used in many bandit approaches for addressing a wide range of bandit problems [6, 17, 27]. Therefore, in this section, we experimentally compare the effectiveness of using either the multinomial or binary rewards in various bandit approaches within BanditProp on two datasets. Figure 3 presents the recommendation performance differences between the use of binary or multinomial rewards for various bandit approaches within BanditProp. Note that, as an illustration, we use the MAP ranking metric when measuring the performance differences of these approaches[6]. We observed similar trends when using other ranking metrics.

In Figure 3, we observe that the multinomial rewards outperform the binary rewards when used within most bandit approaches with the exceptions of Bayes-UCB and TS on Amazon. This shows the effectiveness of using the multinomial rewards to measure the values of arms when addressing our bandit problem. Moreover, for the Bayes-UCB and TS algorithms, the binary rewards do not consistently enhance the performances of both Bayes-UCB and TS on the two used datasets (i.e. the multinomial rewards are a better choice for both approaches on Yelp). Therefore, from the obtained results, we conclude that to address our tackled bandit problem,the multinomial rewards are more effective than the binary rewards when used in various bandit approaches within the BanditProp model.

7 CONCLUSIONS

We proposed a novel recommendation model, BanditProp, which encompasses a multi-task learning and a bandit search strategy to estimate the users' preferences on the review properties thereby enhancing the recommendation performances. We investigated various bandit search algorithms to address the selection of review properties to use for a given user, including a novel neural contextual bandit approach called ConvBandit. Our extensive experiments using two well-known datasets showed that BanditProp significantly and consistently outperforms one classical and six state-of-the-art baselines over the two used datasets. Moreover, we showed that the proposed

[6]Mul-TS is not included, since it is designed to model the multinomial rewards.

ConvBandit algorithm significantly outperforms other strong bandit algorithms. Our obtained results demonstrate that the additional features extracted from the users' posted reviews allow to effectively capture the users' preferences on the review properties for an enhanced recommendation performance. Our experiments also showed the effectiveness of using the multinomial rewards to capture the values of arms, when addressing our tackled bandit problem. As future work, we plan to further evaluate the effectiveness of our proposed BanditProp model in some extreme usage cases, such as when users have few or no reviews (i.e. cold-start vs. full cold-start users) or users with mostly positive or negative reviews.

REFERENCES

[1] Rishabh Agarwal, Chen Liang, Dale Schuurmans, and Mohammad Norouzi. 2019. Learning to generalize from sparse and underspecified rewards. In *Proc. of ICLR*.

[2] Sanae Amani and Christos Thrampoulidis. 2021. UCB-based Algorithms for Multinomial Logistic Regression Bandits. *CoRR* abs/2103.11489 (2021).

[3] Alessia Antelmi. 2019. Towards an Exhaustive Framework for Online Social Networks User Behaviour Modelling. In *Proc. of UMAP*.

[4] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning* 47, 2 (2002).

[5] Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. 2017. Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *Proc. of SIGKDD*.

[6] Djallel Bouneffouf, Irina Rish, Guillermo A. Cecchi, and Raphaël Féraud. 2017. Context Attentive Bandits: Contextual Bandit with Restricted Context. In *Proc. of IJCAI*.

[7] Rocío Cañamares, Marcos Redondo, and Pablo Castells. 2019. Multi-armed recommender system bandit ensembles. In *Proc. of RecSys*.

[8] Rich Caruana. 1993. Multitask Learning: A Knowledge-Based Source of Inductive Bias. In *Proc. of ICML*.

[9] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proc. of WWW*.

[10] Zhongxia Chen, Xiting Wang, Xing Xie, Tong Wu, Guoqing Bu, Yining Wang, and Enhong Chen. 2019. Co-Attentive Multi-Task Learning for Explainable Recommendation.. In *Proc. of IJCAI*.

[11] Zhiyong Cheng, Ying Ding, Xiangnan He, Lei Zhu, Xuemeng Song, and Mohan S Kankanhalli. 2018. Aˆ 3NCF: An Adaptive Aspect Attention Model for Rating Prediction.. In *Proc. of IJCAI*.

[12] Vincent A. Cicirello and Stephen F. Smith. 2005. The Max *K*-Armed Bandit: A New Model of Exploration Applied to Search Heuristic Selection. In *Proc.of AAAI*.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of NAACL-HLT*.

[14] Alexandre dos Santos Mignon and Ricardo Luis de Azevedo da Rocha. 2017. An Adaptive Implementation of ϵ-Greedy in Reinforcement Learning. In *Proc. of ANT*.

[15] Raffaele Filieri and Fraser McLeay. 2014. E-WOM and accommodation: An analysis of the factors that influence travelers' adoption of information from online reviews. *Journal of Travel Research* 53, 1 (2014).

[16] Xinyu Guan, Zhiyong Cheng, Xiangnan He, Yongfeng Zhang, Zhibo Zhu, Qinke Peng, and Tat-Seng Chua. 2019. Attentive Aspect Modeling for Review-Aware Recommendation. *ACM Transactions Information System* 37, 3 (2019).

[17] Dalin Guo and Angela J. Yu. 2018. Why so gloomy? A Bayesian explanation of human pessimism bias in the multi-armed bandit task. In *Proc. of NeurIPS*.

[18] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proc. of WWW*.

[19] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proc. of CIKM*.

[20] Jizhou Huang, Wei Zhang, Yaming Sun, Haifeng Wang, and Ting Liu. 2018. Improving Entity Recommendation with Search Log and Multi-Task Learning. In *Proc. of IJCAI*.

[21] Long Jin, Yang Chen, Tianyi Wang, Pan Hui, and Athanasios V. Vasilakos. 2013. Understanding user behavior in online social networks: a survey. *IEEE Communications Magazine* 51, 9 (2013).

[22] Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. 2012. On Bayesian Upper Confidence Bounds for Bandit Problems. In *Proc. of AISTATS*.

[23] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional matrix factorization for document context-aware recommendation. In *Proc. of RecSys*.

[24] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proc. of EMNLP*.

[25] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proc. of ICLR*.

[26] Volodymyr Kuleshov and Doina Precup. 2000. Algorithms for multi-armed bandit problems. *Machine Learning Research* 1 (2000).

[27] Branislav Kveton, Csaba Szepesvári, Sharan Vaswani, Zheng Wen, Tor Lattimore, and Mohammad Ghavamzadeh. 2019. Garbage In, Reward Out: Bootstrapping Exploration in Multi-Armed Bandits. In *Proc. of ICML*.

[28] Chenliang Li, Xichuan Niu, Xiangyang Luo, Zhenzhong Chen, and Cong Quan. 2019. A Review-Driven Neural Model for Sequential Recommendation. In *Proc. of IJCAI*.

[29] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proc. of WWW*.

[30] Guang Ling, Michael R Lyu, and Irwin King. 2014. Ratings meet reviews, a combined approach to recommend. In *Proc. of RecSys*.

[31] Feng Liu, Ruiming Tang, Huifeng Guo, Xutao Li, Yunming Ye, and Xiuqiang He. 2020. Top-aware reinforcement learning based recommendation. *Neurocomputing* 417 (2020), 255–269.

[32] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Why I like it: multi-task learning for recommendation and explanation. In *Proc. of RecSys*.

[33] Craig Macdonald, Nicola Tonellotto, and Sean MacAvaney. 2021. IR From Bag-of-words to BERT and Beyond through Practical Experiments. In *Proc. of CIKM*.

[34] Jarana Manotumruksa and Emine Yilmaz. 2020. Sequential-based Adversarial Optimisation for Personalised Top-N Item Recommendation. In *Proc. of SIGIR*.

[35] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proc. of RecSys*.

[36] Oren Melamud, Mihaela Bornea, and Ken Barker. 2019. Combining unsupervised pre-training and annotator rationales to improve low-shot text classification. In *Proc. of EMNLP-IJCNLP*.

[37] Xia Ning and George Karypis. 2010. Multi-task learning for recommender system. In *Proc. of ACML*.

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Proc. of NeurIPS*.

[39] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proc. of UAI*.

[40] Charles Riou and Junya Honda. 2020. Bandit Algorithms Based on Thompson Sampling for Bounded Reward Distributions. In *Proc. of ALT*.

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

[42] Sebastian Ruder. 2017. An Overview of Multi-Task Learning in Deep Neural Networks. *CoRR* abs/1706.05098 (2017).

[43] Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. 2018. A Tutorial on Thompson Sampling. *Foundations and Trends in Machine Learning* 11, 1 (2018).

[44] Noveen Sachdeva and Julian J. McAuley. 2020. How Useful are Reviews for Recommendation? A Critical Review and Potential Improvements. In *Proc. of SIGIR*.

[45] Otmane Sakhi, Stephen Bonner, David Rohde, and Flavian Vasile. 2020. BLOB: A Probabilistic Model for Recommendation that Combines Organic and Bandit Signals. In *Proc. of SIGKDD*.

[46] Javier Sanz-Cruzado, Pablo Castells, and Esther López. 2019. A simple multi-armed nearest-neighbor bandit for interactive recommendation. In *Proc. of RecSys*.

[47] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proc. of RecSys*.

[48] David R Shanks, Richard J Tunney, and John D McCarthy. 2002. A re-examination of probability matching and rational choice. *Behavioral Decision Making* 15, 3 (2002).

[49] Zhengxiang Shi, Qiang Zhang, and Aldo Lipani. 2022. StepGame: A New Benchmark for Robust Multi-Hop Spatial Reasoning in Texts. In *Proc. of AAAI*.

[50] Jianing Sun, Yingxue Zhang, Wei Guo, Huifeng Guo, Ruiming Tang, Xiuqiang He, Chen Ma, and Mark Coates. 2020. Neighbor Interaction Aware Graph Convolution Networks for Recommendation. In *Proc. of SIGIR*.

[51] Stephanie Watts Sussman and Wendy Schneier Siegal. 2003. Informational influence in organizations: An integrated approach to knowledge adoption. *Information systems research* 14, 1 (2003).

[52] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proc. of WSDM*.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proc. of NeurIPS*.

[54] Shuai Wang, Shengyao Zhuang, and Guido Zuccon. 2021. Bert-based dense retrievers require interpolation with bm25 for effective passage retrieval. In *Proc. of SIGIR*.

[55] Xi Wang, Iadh Ounis, and Craig MacDonald. 2020. Negative Confidence-Aware Weakly Supervised Binary Classification for Effective Review Helpfulness Classification. In *Proc. of CIKM*.

[56] Xi Wang, Iadh Ounis, and Craig MacDonald. 2021. Leveraging Review Properties for Effective Recommendation. In *Proc. of WWW*.

[57] Xiao Xu, Fang Dong, Yanghua Li, Shaojian He, and Xin Li. 2020. Contextual-Bandit Based Personalized Recommendation with Time-Varying User Interests. In *Proc. of AAAI*.

[58] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge J. Belongie, and Deborah Estrin. 2018. Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *Proc. of RecSys*.

[59] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *Proc. of SIGIR*.

[60] Tom Zahavy and Shie Mannor. 2019. Deep Neural Linear Bandits: Overcoming Catastrophic Forgetting through Likelihood Matching. *CoRR* abs/1901.08612 (2019).

[61] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proc. of CIKM*.

[62] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. 2010. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics* 1, 1 (2010), 43–52.

[63] Xiangyu Zhao, Liang Zhang, Long Xia, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep reinforcement learning for list-wise recommendations. *CoRR* abs/1801.00209 (2018).

[64] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proc. of WSDM*.

[65] Dongruo Zhou, Lihong Li, and Quanquan Gu. 2020. Neural Contextual Bandits with UCB-based Exploration. In *Proc. of ICML*.