Social Explorative Attention based Recommendation for Content Distribution Platforms

Wenyi Xiao · Huan Zhao · Haojie Pan · Yangqiu Song · Vincent W. Zheng · Qiang Yang

Received: 23 February 2020 / Accepted: 2 December 2020

Abstract In modern social media platforms, an effective content recommendation should benefit both creators to bring genuine benefits to them and consumers to help them get really interesting content. To address the limitations of existing methods for social recommendation, we propose Social Explorative Attention Network (SEAN), a social recommendation framework that uses a personalized content recommendation model to encourage personal interests driven recommendation. SEAN has two versions: (1) SEAN-END2END allows user's attention vector to attend their personalized interested points in the documents. (2) SEAN-KEYWORD extracts keywords from users' historical readings to capture their long-term interests. It is much faster than the first version, more suitable for practical usage, while SEAN-END2END is more effective. Both versions allow the personalization factors to attend to users'

Wenyi Xiao HKUST, Hong Kong, China E-mail: wxiaoae@cse.ust.hk

Huan Zhao 4Paradigm Inc., China E-mail: zhaohuan@4paradigm.com

Haojie Pan HKUST, Hong Kong, China E-mail: hpanad@cse.ust.hk

Yangqiu Song HKUST, Hong Kong, China Peng Cheng Laboratory, Shenzhen, China E-mail: yqsong@cse.ust.hk

Vincent W. Zheng WeBank, China E-mail: vincentz@webank.com

Qiang Yang WeBank, China E-mail: qiangyang@webank.com higher-order friends on the social network to improve the accuracy and diversity of recommendation results. Constructing two datasets in two languages, English and Spanish, from a popular decentralized content distribution platform, Steemit, we compare SEAN models with state-of-the-art collaborative filtering (CF) and content based recommendation approaches. Experimental results demonstrate the effectiveness of SEAN in terms of both Gini coefficients for recommendation equality and F1 scores for recommendation accuracy.

Keywords Content Recommendation \cdot Social Recommendation \cdot Social Attention Networks \cdot Monte Carlo Tree Search

1 Introduction

With the prevalence of online social platforms, content recommendation has developed as a promising direction that leverages some side information such as social connections among users to enhance recommendation performance. When applying to modern content distribution platforms, such as Facebook and Steemit¹, an effective content recommendation algorithm should consider both content creators to bring genuine benefits to them and content consumers to help them get really interesting contents. While more accurate recommendation can improve the consumers' reading experience, it is regarded as a healthier content distribution ecosystem that encourages individuals to create contents. However, existing recommendation algorithms still lack consideration for balancing both content creators and consumers.

In the literature, content recommendation methods can be content based or collaborative filtering (CF) based ones. Content based methods (Wang et al. 2017, 2018b) memorize historical reading/watching content of a user and predict his/her future reading/watching content based on features or similarities of both contents. Such approaches emphasize particular topics for a user and may not be able to encourage diversity of recommendation results unless a content consumer actively searches for new topics. On the other hand, CF is considered as a complementary technique for content recommendation (Das et al. 2007) as it recommends based on similar users' clicking behaviors in the whole platform. CF usually optimizes based on global behavior information so that the platform will attract more clicks or reading actions. Despite the recommending effectiveness, CF will produce unintended Matthew's Effects ("The Rich Get Richer") (Rigney 2010) which will hurt small/new content creators who may not be able to attract attentions. Although most of the traditional CF based methods are often called personalized recommendation (Das et al. 2007: Liu et al. 2010) and can be generalized to social networks using social regularization (Jamali and Ester 2010; Ma et al. 2011; Ye et al. 2012;

¹ Steemit (https://steemit.com/) is a blockchain based social media and decentralized content distribution platform for consumers and creators to earn Steemit tokens by playing with the platform and interacting with others. It is regarded as a more effective content distribution ecosystem that allows small content creators to share their creative contents while protecting the copyright without any intermediaries.



Fig. 1 Comparison of Gini coefficients of different algorithms for 368 days using Steemit social media. Gini coefficient is computed over the distribution of recommendation impression numbers of content creators. We compared content based DKN (Wang et al. 2018b) (F1=42.85), NCF (He et al. 2017) (F1=42.14), and our algorithm SEAN with (F1=47.69) and without any social information (F1=42.40).

Yang et al. 2012; Zhao et al. 2017; Sun et al. 2018; Chen et al. 2019a), they are in nature looking at global information and cannot solve this problem.

A few RSs (Salganik et al. 2006; Abeliuk et al. 2017) analyze the Matthew's Effect. However, they are still CF based recommendation and the recommendation strategies are relatively simple, e.g., using popularity (Salganik et al. 2006) or quality (Abeliuk et al. 2017; Berbeglia and Hentenryck 2017) of the content. Moreover, one possible way to reduce Matthew's Effect is to use mechanism design approaches in game theory (Berbeglia and Hentenryck 2017). In fact, the developed strategy (e.g., introducing randomness in recommendation (Berbeglia and Hentenryck 2017)) may hurt consumer's satisfaction as the recommended content may not be related to a consumer's interests, but such effect has not been considered and discussed yet. As shown in Figure 1, we use the Gini coefficient over the distribution of recommendation impression numbers for content creators on Steemit social media to demonstrate Matthew's Effect, as the Gini coefficient is usually used to measure inequality and large Gini values mean eventually a small number of content creators dominating the content consumption. We also use the F1 score of prediction to evaluate content consumer's satisfaction. From the Figure 1, we can see that the state-of-the-art CF based approach NCF (He et al. 2017) encourages inequality more than content based approach DKN (Wang et al. 2018b) although their F1 scores are comparable to each other. Thus, a natural question is can we have a content recommendation algorithm that can further benefit content consumers while not hurting creators?

In this article, we propose the social explorative attention network (SEAN) to consider both creators and consumers. For *creator equality*, we use a content based approach. However, as a traditional content based approach, DKN encodes the new incoming document into a unique vector (the same vector that will be compared with all users) and uses this vector to attend to a user's historically read documents. In this way, popular content will still tend to be

selected by the final prediction classifier regardless of the user's interests. Different from DKN, we use user-dependent vectors to attend to related words and sentences in a new incoming document. In this way, we compress a user's interests into contextual vectors and use the user-dependent document representation vector to feed the final prediction classifier. This is more compatible with the personalized nature of content recommendation that can benefit small creators, as long as the created content is of the consumer's interests. As shown in Figure 1, our model without social information can achieve comparable F1 while significantly reduce the Gini coefficient. A natural way to further encourage diversity and improve creators' equality is to introduce more randomness as suggested by the mechanism design approach (Berbeglia and Hentenryck 2017). We demonstrate this by randomly exploring other consumers' interests. However, this mechanism will hurt prediction accuracy which reflects the consumer's satisfaction.

To improve the *consumer satisfaction*, as the training data for each user's context vectors may be too small and cannot train well for users having less reading history, we design two variants of SEAN, SEAN-END2END and SEAN-KEYWORD, to address it. In SEAN-END2END, we allow a user's context vectors to "attend" to friends' context vectors and fetch back friends' reading interests and prediction knowledge by aggregating their context vectors. In SEAN-KEYWORD, we extract keywords and use a word-level attention module to attend to the user's interested words in both documents and users' historical readings. Nonetheless, even a user has many one-hop friends, friends sharing a similar topic of interest may not be enough. Therefore, we consider the user's higher-order friends. An extreme case is that we go over all *n*-hop friends, and we can likely reach all connected users in a social network when n is large. Apparently, it will be too expensive to explore. To remedy this, we develop a social exploration mechanism based on Monte Carlo Tree Search (MCTS) (Silver et al. 2016). This will be more effective to attend to higher-order friends. In particular, by using MCTS, we can achieve a good balance of finding *n*-hop friends with similar interests and exploring friends with some randomness for more diverse interests in a social network. As shown in Figure 1, introducing social information will significantly improve the F1 score but also increase the Gini coefficient. Therefore, in the experiments, we systematically study how different hyper-parameters of SEAN, including social information, can affect both prediction accuracy and the Gini coefficient.

Our contributions can be highlighted as follows:

- To the best of our knowledge, this is the first study on comprehensive exploring of news recommendation with social connections on decentralized platforms, where the creator equality is much more important than traditional content distribution platforms. In particular, we use the Gini coefficient to measure the inequality of content creators based on recommendation impressions.
- We propose a novel social explorative attention based recommendation model, SEAN, to use a user's personal reading history and go beyond

personal data to explore the user's higher-order friends. We compare two variants, SEAN-END2END and SEAN-KEYWORD, by considering both effectiveness and efficiency.

- We construct two datasets of different languages from a popular decentralized content distribution platform, Steemit. By conducting extensive experiments, we demonstrate the superiority of our model over existing state-ofthe-art recommendation approaches, including CF and content based ones, in terms of benefiting both consumers and creators.

Preliminary results of this manuscript have been reported in (Xiao et al. 2019), published as a conference paper in SIGKDD 2019, where MCTS is designed to explore high-order friends for the target user, and then a personalized hierarchical attention network is proposed for news recommendation. In this full version, we propose and highlight the content modeling and friend selection for the user, which provides a comprehensive solution for fusing social information for RS on decentralized platforms. We conduct extensive research on social exploration strategy designing, content modeling, which constitutes a novel and effective user-item interaction model for more industry-driven applications. Hence, the contributions of this work lie in broader domains. To be specific, for social exploration, besides MCTS, we propose to use ϵ -Greedy as another strategy to balance exploitation and exploration, introduced in Section 3.2. For content modeling, we propose two versions of SEAN, which have different interactions among documents and users: (1) The end-to-end version (SEAN-END2END) uses user-dependent vectors to attend to related words and sentences in a new incoming document, introduced in Section 4. This is the original model(SEAN) introduced in (Xiao et al. 2019). (2) The keyword version (SEAN-KEYWORD), introduced in Section 5, first extracts keywords, and then a use word-level attention module to attend to user's interested words in both newly incoming documents and users' historical readings dynamically. Furthermore, for incorporating social information, we propose both dynamic attention and static attention (various similarity functions) for weighing the influence of users' friends. SEAN-END2END is more effective while SEAN-KEYWORD runs much faster. Additional experiments are performed, to support the increased components in Section 7.5, 7.6, 7.7, 7.8.3 and 7.9, respectively. Finally, we give a comprehensive review on related works in Section 8 and point out some potential research for the future works in Section 9.

We release the code and datasets for researchers to validate the reported results and conduct further research. The code and data are available at https://github.com/HKUST-KnowComp/Social-Explorative-Attention-Networks.

2 Overview

In this section, we first give the descriptions of the problem we plan to solve and list the key notations used in the article. Then we give a brief introduction of model frameworks.

Notation	Meaning			
$\mathcal{G} = (\mathcal{U}, \mathcal{E})$	social graph \mathcal{G} , set of users \mathcal{U} , set of edges \mathcal{E}			
$Q_t(v)$	exploitation reward of node v at day t			
$U_t(v)$	exploration score of node v at day t			
B	beam width			
L	search depth $/ \#$ of selected friends			
λ	trade-off constant			
ϵ	probability to take a random action in ϵ -Greedy			
α	PageRank value for SPR & DPR			
K	number of kernels in CNN			
g	window size in CNN			
r	filter size in CNN			
h	user embedding size & hidden size			
D	the dimension of word embedding			
m	# of keywords to represent user			
n	# of keywords to represent document			

Table 1 Key notations in this article.

2.1 Problem Formulation

The recommendation task is to predict whether a target user u will click a given document d. Here we use the textual document recommendation as an example for content recommendation. We assume that we have a social graph $\mathcal{G} = (\mathcal{U}, \mathcal{E})$, where \mathcal{U} is the set of users, and \mathcal{E} is the set of edges, representing the social connection between two users. Our goal is to learn a prediction function $p = \mathcal{F}(u, d, \theta)$, where p represents the probability that user u will click a given document d, and θ denotes the model parameters of function \mathcal{F} . For the sake of clear presentation, we list the key notations used in this article in Table 1.

2.2 Overall Framework

In this article, we propose to use a personalized model to perform content recommendation, as personalization will encourage the model to find more relevant contents and less affected by the global information about popularity and social influence. Then we socialize it to make the personalized factors be able to attend to friends' information, which will further balance the randomness factor to improve the creator equality and the relatedness factor to improve the consumer satisfaction. To enable the attention over attention mechanism to use more information, we propose to explore a user's higher-order friends. We call our recommendation model Social Explorative Attention Network (SEAN). And the details of two verions of SEAN, SEAN-END2END and SEAN-KEYWORD, will be introduced in the following subsection.

For the whole procedure, we first initialize paths for users by randomly selecting users in the social graph and set users' explored times as the times they selected as friends. For each user in the *t*-th day training, we first explore B sets of friends by MCTS strategy and update the explored times for these selected friends, consequently updating the exploration values $U_t(v)$ of them. These B sets of friends are incorporated with the user as input to the RS model. Then we update $Q_t(u)$ of the target user. The updated results are used for the t + 1-th day training. We show the algorithm for the model training in Algorithm 1. Besides MCTS, we also propose to use ϵ -Greedy as another strategy to balance exploitation and exploration and select high-order friends.

Algorithm 1 SEAN.

	7
1:	for $t = 1, 2, \cdots$ do
2:	$\mathbf{for} u\in\mathcal{U} \mathbf{do}$
3:	$\{\mathcal{F}_i(u)\}_{b=1}^B = \text{SelectFriends}(u, B, L)$
4:	for $b = 1, 2, \cdots, B$ do
5:	Train SEAN with $\mathcal{F}_b(u)$ for u ;
6:	for $v \in \mathcal{F}_b(u)$ do
7:	$N_t(v) \leftarrow N_t(v) + \frac{1}{B};$
8:	Update $U_t(v)$ according to Eq. (2);
9:	end for
10:	Update $Q_t(u)$ for user u according to Section 3.1.1;
11:	end for
12:	end for
13:	end for

In the remaining part of this paper, we first introduce the social exploration in Section 3, and then the two content modeling models in Section 4 and Section 5, respectively. Further in Section 7 extensive experiments are conducted to demonstrate the effectiveness of SEAN in terms of recommendation equality and accuracy. Finally we review related work in Section 8 and conclude this work in Section 9.

3 Friend Selection

In this section, we introduce two approaches to select relative high-order friends to enhance the effectiveness of our recommender system. In section 3.1, we demonstrate how we use MCTS for friends selection, and then further enhance MCTS with beam search. In section 3.2, ϵ -Greedy is introduced as another strategy to select high-order friends comparing to MCTS.

3.1 Selecting Friends with MCTS

Monte Carlo Tree Search (MCTS) (Silver et al. 2016) is a stochastic search algorithm to find an optimal solution in the decision space. It models an agent that simultaneously attempts to acquire new knowledge (called "exploration") and optimize the decisions based on existing knowledge (called "exploration"). MCTS uses the upper confidence bounds one (UCB1) (Kocsis and Szepesvári

2006) value to determine the next move a from a viewpoint of multi-armed bandit problem. The selection strategy is defined by:

$$a = \underset{v}{\operatorname{argmax}} \{ Q_t(v) + \lambda \cdot U_t(v) \}, \tag{1}$$

where $Q_t(v)$ denotes the empirical mean exploitation reward of node v at time t and $U_t(v)$ is the utility to explore node v. This equation clearly expresses the exploration-exploitation trade-off: while the first term of the sum tends to exploit the seemingly optimal arm, the second term of the sum tends to explore less pulled arms. λ is used to balance the two terms.

Here we explain how MCTS guides to generate a path with a fixed number of search depth L, regarded as L friends of u by walking through the social graph. We denote the target user u as the starting node c_0 and denote $c_l, l \in$ [0, L] as the l-th node added for u in the path. On day t at search step l, the node c_{l+1} is retrieved from the neighbors of node c_l . We calculate the score for each neighbor according to Eq. (1) and choose the neighbor with the maximum score as the (l+1)-th friend of the user u. The design of calculating the values from exploitation and exploration are mentioned below.

3.1.1 Exploitation

On day t, we compute the $Q_t(v)$ to get the exploitation reward of neighbor node v. In our scenarios, we want to select those as friends who can improve the recommending performance as much as they can. In this work, we design four exploitation strategies to select friends for maximizing $Q_t(v)$: the average F1 from RS model (SEAN-RS-F1), static PageRank value from social network (SEAN-SPR), dynamic PageRank value from activity network (SEAN-DPR), as well as the actual payout each user earned in blockchain platforms (SEAN-Payout).

- 1. **SEAN-RS-F1.** We regard the average F1 evaluated based on our RS model of each neighbor node v up to time t as the exploitation reward $Q_t(v)$. This is based on the assumption that a user who has been well-learned by the RS model is reliable and could be exploited as a friend for the target user u in the future. In this way, the RS prediction results can guide the friend exploration process, and in turn, the friend exploration process provides useful friends to help enrich the target user's representation.
- 2. **SEAN-SPR.** The second way is to use the PageRank value of v obtained from social network as exploitation reward $Q_t(v)$. In (Xiang et al. 2013), Xiang et.al. explicitly connect PageRank with social influence model and show that authority is equivalent to influence under their framework. Thus, we assume that a node with high PageRank value in the social network is influential and should be exploited as a friend for the target user u.
- 3. **SEAN-DPR.** On social media platforms, each user can not only make activities on the documents (as a consumer) but also create documents (as creator). We build a dynamic activity network and calculate the PageRank



Fig. 2 MCTS for social exploration. We illustrate our MCTS based strategy with this example. We set the beam width and search depth to 2. The node '1' represents the target user. We initialize 2 paths according to the beam width and add "1" to each path. In the Selection step, we calculate the scores by Eq. (1) of '1' is neighbors ('2', '3', '4', '5') and select two nodes with the largest scores ('3', '4') and add them to each path. In the Expansion step, we calculate the scores for the neighbors of both "3" and "4", and again select two nodes ('7', '8') among all the neighbors. In the Evaluation step, the two generated path ('1' \rightarrow '3' \rightarrow '7' \rightarrow '1' \rightarrow '4' \rightarrow '8') are input to RS model. In the same way, we get the paths for other nodes. In the Backup step, we update $Q_t(1)$ from the result at Evaluation and $U_t(3), U_t(4), U_t(7), U_t(8)$ from Selection & Expansion.

values of nodes. Compared with the social network, the edges in the activity network are the consumer-creator connection.

4. **SEAN-Payout.** In some blockchain based social platforms, the platform would give some rewards, i.e., bitcoin, to those users who help distribute the documents, i.e., post or forward a document in the platform. We regard the payout that a user gains as the value of his/her exploitation value $Q_t(v)$.

3.1.2 Exploration

We design the exploration mechanism to get the explored reward $U_t(v)$ for friend v as follows:

$$U_t(v) = \sqrt{\frac{\log N_t(c_l)}{N_t(v) + 1}},$$
(2)

where $N_t(c_l)$, $N_t(v)$ denote as the times that the current node c_l at search step l and the neighbor node v have been selected as friends up to day t, respectively. The goal of the exploration is to select the nodes who have less been explored in the past.

3.1.3 Obtaining Multi-paths with Beam Search

If we want to find higher-order friends, we can greedily select the next node with a maximum score from the neighbors of the current node at search step l. In this way, we would get a path of higher-order friends. If we want to find more than one path, it is time-consuming to get a globally optimal set of paths. Therefore, we combine MCTS with beam search (Koehn 2004) to balance the optimality and completeness. At search step l, we choose the neighbors

Algorithm 2 SelectFriends(u, B, L). **Input:** target user u, beam width B, path length L**Output:** $\left\{\mathcal{F}_b(u)\right\}_{b=1}^B$ 1: Initialization: $\{\mathcal{F}_{b}(u)\}_{b=1}^{B}: \mathcal{F}_{b}(u)$ records the *b*-th path starting from *u*; UCB1(v): UCB1 score for user *v* according to Eq. (1); $\{T_b(u)\}_{b=1}^{B}$: $T_b(u)$ records the sum of UCB1 scores of the b-th path for user u during beam search; Δ_b : the neighbours of the tail node of the path $\mathcal{F}_b(u)$ during beam search; while $k = 0, 1, 2, \cdots$, L: do 2: $\mathcal{H} = \bigcup_{b=1}^{B} \{ UCB1(v) + T_b(u), v \in \Delta_b \};$ while $b = 1, 2, \cdots, B$: do 3: 4: $v = \operatorname{argmax}_{v} \mathcal{H}$; 5: 6: $\mathcal{F}_b(u) \leftarrow \mathcal{F}_b(u) \bigcup v;$ 7: $\mathcal{H} \leftarrow \mathcal{H} \setminus (UCB1(v) + T_b(u));$ end while 8: 9: end while

with largest B scores from Eq. (1) and these B nodes are selected for further expansion. Here B is the beam width. In this way, we generate B paths for the target user u. For training and testing, we obtain B prediction results by using each path and u and compute the average of these results to get the final prediction. We give a concrete example of MCTS for social exploration, shown in Figure 2. Besides, we give the whole procedure on how to select friends based on beam MCTS in Algorithm 2.

3.2 Selecting Friends with ϵ -Greedy

 ϵ -Greedy is the most popular and the simplest method to balance exploration and exploitation by taking the best action most of the time but do random exploration occasionally. We give a detailed introduction on how to incorporate this idea into the friend selection process in section 3.2.1 and discuss the motivation and strength&weakness of using ϵ -Greedy in section 3.2.2.

3.2.1 Algorithm of ϵ -Greedy

In detail, for each step, with a probability ϵ we randomly select the friends without any bias. Meanwhile, with probability $1 - \epsilon$ we select friends with higher exploitation value $Q_t(v)$ based on Section 3.1.1. To obtain B different paths, we follow the work in (Grover and Leskovec 2016), and the details are given in Algorithm 3.

3.2.2 Discussion on ϵ -Greedy

When settings ϵ to 0, it is a fully exploitative choice, thus the random walk process easily gets stuck to a finite set of vertices. It may not be good enough to explore the full graph. In practice, keeping a vaguely explorative/stochastic

Algorithm 5 Selectifiends (u, D, L, ϵ) .	
Input: target user u , $\#$ of path B , path length L , ϵ	
Output: $\{\mathcal{F}_b(u)\}_{b=1}^B$.	

Almonithms 2 Soloot Drive da(a)

```
1: Initialization:
       \{\mathcal{F}_b(u)\}_{b=1}^B: \mathcal{F}_b(u) records the b-th path starting from u; \Delta_b:the neighbours of the tail node of the path \mathcal{F}_b(u) during beam search;
 2:
      while b = 1, 2, \cdots, B: do
 3:
           while k = 1, 2, \dots, L: do
               if probability < \epsilon: then
 4:
 5:
                    \mathcal{H} = \{Q(v)\}, v \in \Delta_b;
 6:
                    v = \operatorname{argmax}_{v} \mathcal{H};
 7
               else
 8:
                    v = \text{Random } \Delta_b;
 9:
               end if
10:
                \mathcal{F}_b(u) \leftarrow \mathcal{F}_b(u) \bigcup v;
11:
           end while
12: end while
```

DI

element in its policy (like a tiny amount of ϵ) allows it to get out of such states. Compared to MCTS, ϵ is simple and straightforward to optimize.

Compared to MCTS, ϵ is simple and straightforward to optimize. Despite its simplicity, ϵ -Greedy often yields pretty good results in some reinforcement learning tasks (Mnih et al. 2015). However, ϵ -Greedy is less efficient to explore the most relative neighbors than MCTS since the exploration process in MCTS has supervising signals, consequently decreasing the searching time.

4 SEAN-END2END

SEAN-END2END is an end-to-end framework to obtain better representation for users in social media with two modifications of recommendation: personalization and socialization. The overview architecture is shown in Figure 3, and it contains three major components:

- 1. Document Representation: we adopt the hierarchical attention networks (Yang et al. 2016). In the word level, the attention is used to select useful words to construct features for sentence representations.
- 2. User Representation: we construct user representation vectors (word and sentence levels) themselves as attentions to his/her friends' representation vectors, which is essentially an attention over attention model.
- 3. Output Layer: predicting the possibility that the candidate document will be clicked by the target user.

4.1 Socialized Document Representation

Assume that a document has I sentences and each sentence contains J words. w_{ij} represents the *j*-th word in sentence s_i with the indices $i \in [0, I]$ and $j \in [0, J]$. We use a hierarchical architecture to learn the document representation.



Fig. 3 The architecture of SEAN. The left side is a social exploration module that explores high-order friends for the RS system on the ride side. These friends are incorporated with the user to build the user's representation vector in word and sentence levels, respectively. The right side is a hierarchical architecture from CNN layer to encode words to GRU layer to encode sentences in the document. The user's representation vectors are used to attend to important words and sentences in the candidate document.

4.1.1 Word Level Personalization

We use pre-trained word embeddings for words and fix them during training. The embedding of each word can be calculated as $\mathbf{w}_{ij} = \mathbf{W}_e \mathbf{e}_{w_{ij}}, i \in [0, I], j \in [0, J]$, where \mathbf{W}_e is the embedding matrix for all words and $\mathbf{e}_{w_{ij}}$ is a onehot vector to select one word embedding vector \mathbf{w}_{ij} for w_{ij} . We concatenate all word embeddings in a sentence to form a sentence matrix $\mathbf{W}_i \in \mathbb{R}^{D \times J}$ for sentence s_i , where D is the dimension size of the embedding vector of each word. We then use a convolutional neural network (CNN) (Kim 2014) to represent sentences in the document. Here, we apply a convolution operation on \mathbf{W}_i with a kernel $\mathbf{K}_k \in \mathbb{R}^{g \times r \times D}$, $k \in [0, K]$ among K kernels of width gand filter size r to obtain the feature \mathbf{f}^k :

$$\mathbf{F}_{ij}^{k} = \operatorname{relu}\left(\mathbf{W}_{i}\left[*, j: j+g-1\right] \odot \mathbf{K}_{k} + \mathbf{b}_{k}\right),\tag{3}$$

where $j \in [1, J - g + 1]$ is the iteration index of convolution, $\mathbf{f}_{ij}^k \in \mathbb{R}^r$ is regarded as *j*-th CNN feature by the *k*-th kernel K_k , and the bias vector \mathbf{b}_k in *i*-th sentence.

We then feed each CNN features \mathbf{f}_{ij}^k to a non-linear layer, parameterized by a global weight matrix $\mathbf{W}_w \in \mathbb{R}^{h \times r}$ to get a hidden representation $\mathbf{f}_{ij}^{k'}$. h and r are pre-defined dimensions of hidden vectors. We measure the importance of the word towards the target user as the similarity of $\mathbf{f}_{ij}^{k'}$ and the wordlevel user's socialized representation vector \mathbf{x}_w (which will be introduced in Section 4.2). The sentence representation vector \mathbf{s}_i^k by CNN with kernel size k is computed as a weighted sum based on the soft attention weights:

$$\mathbf{f}_{ij}^{k'} = \tanh(\mathbf{W}_w \mathbf{f}_{ij}^k + \mathbf{b}_w),\tag{4}$$

$$\alpha_{ij} = \text{Softmax}(\mathbf{x}_w^\top \mathbf{f}_{ij}^{k\,\prime}),\tag{5}$$

$$\mathbf{s}_{i}^{k} = \sum_{j} \alpha_{ij} \mathbf{f}_{ij}^{k},\tag{6}$$

where the superscript \cdot^{\top} represents the vector or matrix transpose. All representation vector \mathbf{s}_i^k are concatenated together and taken as the sentence embedding \mathbf{s}_i for sentence s_i as: $\mathbf{s}_i = [\mathbf{s}_i^1, \mathbf{s}_i^2, ..., \mathbf{s}_i^K]$.

4.1.2 Sentence Level Personalization

At the sentence level, we use a bidirectional Gated Recurrent Unit network (BiGRU) (Bahdanau et al. 2015) to compose a sequence of sentence vectors into a document vector. The BiGRU encodes the sentences from two directions:

$$\mathbf{h}_{i} = \overrightarrow{GRU}(\mathbf{s}_{i}) || \overleftarrow{GRU}(\mathbf{s}_{i}).$$
(7)

After getting \mathbf{h}_i for sentence s_i , we use the sentence level user representation vector \mathbf{x}_s to extract relevant sentences that are interested by the target user and get a final document representation \mathbf{d} by soft attention mechanism similar to sentence representation. We omit the details of equations due to the lack of space and the similarity with the word level computation. As shown in the right side of Figure 3, we have two layers of feature extraction networks. This architecture is inspired by (Yang et al. 2016) since it is better for long document modeling. In our model, we use CNN instead of RNN for word-level since in practice we found that CNN is faster, more robust, and less easy to overfitting on our datasets. Moreover, different from (Yang et al. 2016), we use socialized user representation vectors instead of unified representation vectors for attending to words and sentences.

4.2 Socialized User Representation

We denote \mathbf{e}_u as a one-hot vector of user u and retrieve the word level user representation \mathbf{u}_w from a trainable embedding matrix $\mathbf{A} \in \mathbb{R}^{h \times |\mathcal{U}|}$ by using $\mathbf{A}\mathbf{e}_u$, where h is the size of attention vectors. We can get the user's sentencelevel representation by another trainable embedding matrix \mathbf{A}' in the same way. We design a social attention module to enrich a user's representation by incorporating his/her friends' representations. Let $\mathbf{y}_i \in \mathbb{R}^h, i \in \{1, 2, ...\}$ be his/her friends' word-level representation vectors, and denote $\mathbf{y}_0 = u_w$. The attention mechanism produces a representation \mathbf{x}_w as a weighted sum of the representations vectors $\mathbf{y}_j, j \in \{0, 1, 2, ...\}$ via

$$\alpha_j = \text{Softmax}(\text{LeakyReLU}(\mathbf{w}^\top [\mathbf{W}_y \mathbf{u}_w || \mathbf{W}_y \mathbf{y}_j])), \tag{8}$$

$$\mathbf{x}_w = \sum_j \alpha_j \mathbf{W}_y \mathbf{y}_j,\tag{9}$$

where $\mathbf{W}_y \in \mathbb{R}^{h \times h}$ is a shared linear transformation and || is the concatenation operation. The attention mechanism is a single-layer feedforward neural network, parametrized by a weight vector $\mathbf{w} \in \mathbb{R}^{2h}$, and applying the LeakyReLU nonlinearity.

Similarly, we can get the sentence-level user representation \mathbf{x}_s by the attention of high-order friends' representation vectors.

4.3 Prediction and Learning

Finally, we use a dense layer to predict the probability that the target user u will read the candidate document d:

$$p = \text{Sigmoid}(\mathbf{w}_{q}^{\top}\mathbf{d} + \mathbf{b}), \tag{10}$$

where $\mathbf{w}_g \in \mathbb{R}^{2h}$ is a global trainable weight vector trained by all the samples. **d** is the document representation vector obtained from Section 4.1.

Due to the nature of the implicit feedback and the task of item recommendation, we adopt the binary cross-entropy loss to train our model:

$$\mathcal{L}(\theta) = -\frac{1}{M} \sum_{m=1}^{M} \left[y_m \log(p_m) + (1 - y_m) \log(1 - p_m) \right],$$
(11)

where m is the index of a sample, M is the total number of training samples, $y_m \in \{0,1\}$ is the label, and θ denotes the set of model parameters. The negative samples are formed from the documents that the target user does not make response to while his/her friends make.

During training and testing, we train the model with the data of past t days and test it with the data on (t+1)-th day. The model dynamically adapts to new data day-by-day.

5 SEAN-KEYWORD

In this section, we introduce the SEAN-KEYWORD model. The overview architecture of SEAN-KEYWORD is shown in Figure 4. SEAN-KEYWORD is a content-based model for click-through rate (CTR) prediction, which takes a candidate document, i.e., the article, and a user' clicked history as inputs, and outputs the probability of the user clicking the document. It is a much faster and simpler approach for personalized representations for user and document.



Fig. 4 The overview architecture of SEAN-KEYWORD. Firstly, we use TF-IDF to extract keywords to represent the candidate document and the user, respectively. Then, we apply a word attention layer for the representations of the user and the document. For the social attention layer, the model aggregates all the information from both the user and his/her friends by learning the knowledge contained in the social graph. Finally, the representation is used via the logistic regression to compute the clicking probability.

5.1 Keywords Extraction

We first use TF-IDF to obtain several keywords from the user's reading history to capture his/her long-term interests coarsely. Since the target user is represented by all his/her reading history, it is less practical to use LSTM (Okura et al. 2017) or CNN (Wang et al. 2017) to embed all the documents the users read in textual order due to the computational and space cost. Similarly, we use TF-IDF to rank different words in a document and then use the top words in the ranking list as the representation of the document. Such representations of the document and the user will be used as inputs for the later deep learning model.

5.2 Word Attention Layer

The word attention layer is to learn representations of a user, his/her friends, and a candidate document, respectively. We use pre-trained word embeddings as the representation of each word and fix them during training. Let $X = \{\mathbf{x}_1, ..., \mathbf{x}_m\}$ be word embeddings of keywords extracted from historical documents read by the target user u, where m is the number of keywords to represent u. $\mathbf{x}_k \in \mathbb{R}^D$ with D is the pre-defined dimension of each embedding vector.

An attention mechanism (Bahdanau et al. 2015) is used to extract words that are important to the user's interests as well as the meaning of the document. Specifically, we define the latent representation \mathbf{v} of a user as:

$$\mathbf{h}_k = \tanh(\mathbf{W}_w \mathbf{x}_k + \mathbf{b}_w),\tag{12}$$

$$\mathbf{a}_k = \text{Softmax}(\mathbf{u}_w^\top \mathbf{h}_k),\tag{13}$$

$$r = \sum_{k} \mathbf{a}_{k} \mathbf{h}_{k},\tag{14}$$

where the input word embedding \mathbf{x}_k is put through a non-linear layer, parameterized by a global weight matrix $\mathbf{W}_w \in \mathbb{R}^{h \times d}$ to get a hidden representation \mathbf{h}_k . h is the predefined dimension of each hidden vector. \mathbf{W}_w and \mathbf{b}_w are updated day-by-day by all training samples. The context vector \mathbf{u}_w is randomly initialized and learned during the training process and also updated day-byday. Moreover, the vector \mathbf{u}_w is applied to the user and his/her friends, which can be regarded as a virtual query, querying informative words to the user. Then \mathbf{v} is computed as a weighted sum of word embeddings based on the weights, denoted as importance degrees of words interacted with the context vector \mathbf{u}_w . We do the same process among users' friends in parallel.

Similarly, we use another word attention module to obtain the representation of the candidate document, denoted as \mathbf{q} .

5.3 Social Attention Layer

Now we introduce how we aggregate the friends' influences to enrich the representation of a given user.

The input to this layer is a set of users' representation vectors obtained from previous layers, including the given user's and his/her friends'. The set of users' friends' representations is denoted as $V = {\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_L}$. Here, L is the number of the given user's selected friends, $\mathbf{v}_j \in \mathbb{R}^h$. \mathbf{v}_j means the representation of friend f_j of user u. The output is a social-aware user's representation $\hat{\mathbf{v}}$, which incorporates the information from the given user's friends. In the following steps, we present two social attention approaches, static attention, and dynamic attention, to build a social-aware representation $\hat{\mathbf{v}}$ for the user u.

5.3.1 Static Attention

The static attention based social-aware representation often performs a sum of weighted representations from friends as a general representation, while the weights are computed by certain similarity functions towards each representation of the user's friends and the user himself/herself (McPherson et al. 2001; Ma et al. 2011).

δ	Description		
Cosine	$f\left(x,y\right) = \frac{xy^{T}}{\ x\ \ \ y\ }$		
Polynomial	$f(x,y) = (\gamma x y^{T} + c)^d$		
Sigmoid	$f(x,y) = \tanh\left(\gamma x y^{T} + c\right)$		
RBF	$f(x, y) = \exp\left(-\gamma \ x - y\ ^2\right)$		
Euclidean	$f(x,y) = \frac{1}{1+ x-y }$		
Exponential	$f(x,y) = \exp\left(-\gamma \left\ x - y\right\ _{1}\right)$		
Manhattan	$f(x,y) = \frac{1}{1+\ x-y\ _1}$		
GESD	$f(x,y) = \frac{1}{1+\ x-y\ } \cdot \frac{1}{1+exp[-\alpha(xy]+c)]}$		
AESD	$f(x,y) = \frac{1}{1+\ x-y\ } + \frac{1}{1+exp[-\gamma(xy^{\intercal}+c)]}$		

Table 2 The descriptions of different similarity functions.

Then the general representation from his/her friends is added to the user representation \mathbf{v} . The equation is as follows:

$$\hat{\mathbf{v}} = \mathbf{v} + \sum_{j=1}^{L} \operatorname{Softmax}(\delta(\mathbf{v}, \mathbf{v}_j)) \mathbf{v}_j,$$
(15)

where δ is the similarity functions. In this article, we give nine similarity functions, described in Table 2, to calculate the influence of different friends on user u.

5.3.2 Dynamic Attention

Inspired by graph attention networks (Veličković et al. 2018), we present a novel method to dynamically calculate the importance of each friend to the given user.

Firstly, a shared two-layer network is applied to compute the attention \mathcal{A} on the user's embedding **v** with each friend's **v**_j considering the document's embedding **q**:

$$e_j = \mathcal{A}(\mathbf{v}, \mathbf{v}_j, \mathbf{q}) = \mathbf{u}_v^{\top} \operatorname{ReLU}(\mathbf{W}_1 \mathbf{v} + \mathbf{W}_2 \mathbf{v}_j + \mathbf{W}_3 \mathbf{q} + \mathbf{b}), \quad (16)$$

where $\mathbf{W}_1 \in \mathbb{R}^{h \times h}, \mathbf{W}_1 \in \mathbb{R}^{h \times h}, \mathbf{W}_1 \in \mathbb{R}^{h \times h}, \mathbf{b} \in \mathbb{R}, \mathbf{u}_v \in \mathbb{R}^h$ are model parameters, h denotes the dimension of attention network and RELU is a nonlinear activation function. e_j indicates the informative degree of the friend to the user.

Secondly, we put the attention coefficients through a softmax function to get a normalized importance by

$$a_j = \frac{\exp(e_j)}{\sum_{k \in V} \exp(e_k)}.$$
(17)

Thirdly, we obtain the friend's general representation \mathbf{f} by processing a weighted sum of these friends. Then it is added to the user's individual representation \mathbf{v} to get a social-aware user's representation $\hat{\mathbf{v}}$:

$$\mathbf{f} = \sum_{j \in V} a_j \mathbf{v}_j,\tag{18}$$

$$\hat{\mathbf{v}} = \mathbf{v} + \mathbf{f}.\tag{19}$$

After all these operations, we obtain the representation of document \mathbf{q} and the social-aware representation of user $\hat{\mathbf{v}}$ as:

$$\mathbf{o} = \mathbf{q} \parallel \hat{\mathbf{v}}.\tag{20}$$

Finally, we use a sigmoid function to predict the probability that the target user u will read the candidate document d over the model. For SEAN-KEYWORD, we also adopt the binary cross-entropy loss to train the whole model, the same as SEAN-END2END.

6 Complexity Analysis and Discussion

In this section, we give the complexity analysis on two variants of SEAN and discussing the differences and similarities of the two models.

6.1 Complexity Analysis

6.1.1 For SEAN-END2END

For the word level, the time complexity is linear to the number of tokens in the training data set, which is $M \cdot I \cdot J$, where M is the number of training samples, I is the maximum number of sentences, and J is the maximum number of tokens in a sentence. It is also linear to the number of kernels K, the numbers of hidden vectors h, the filter size r, and the convolutional window size g. Since we use fixed-sized word embeddings, the large number of words do not contribute to our time cost. For the sentence level, the time cost of the GRU layer is linear to the maximum number of sentences I and the number of parameters in the GRU cell P_{GRU} . For both word-level and sentence-level attentions, the cost is linear to the square number of trials of attention B. Note that the number of selected friends L and the times of trials of attention B are the same as the search depth L and beam width B introduced in Section 3. Moreover, for the fully connected layer, the parameter is linear to h. Therefore, the overall time complexity is $O(M \cdot I \cdot J \cdot (K \cdot g \cdot r \cdot D + h \cdot r \cdot K + h^2 \cdot L \cdot B) + M \cdot I \cdot (P_{GRU} + h^2 \cdot L \cdot B))$.

6.1.2 For SEAN-KEYWORD

For the word attention layer to represent the target user, the time cost is linear to the number of tokens in the training data set, which is $O(M \cdot m \cdot L \cdot D \cdot h)$, where M is the number of training samples, m is the number of keywords for the target user, L is the number of friends for the target user, D is the dimension of word embeddings, h is the dimension of the hidden vector. For the social attention layer, the time cost is linear to the number of attended friends L and the square number of hidden dimension h^2 , which is $O(M \cdot L \cdot h^2)$. To represent the candidate document, the time complexity is linear to the number of training samples as well as the keywords extracted for the candidate document, which is $O(M \cdot n \cdot D \cdot h)$, where n is the number of keywords for candidate document. Therefore, the overall complexity is $O(M \cdot m \cdot L \cdot D \cdot h + M \cdot L \cdot h^2 + M \cdot n \cdot D \cdot h)$.

6.2 Discussion on the two variants of SEAN

SEAN-END2END and SEAN-KEYWORD are named according to the techniques of using attentions among users and documents. In this section, we provide a unified view of the two versions of SEAN and show that their major difference lies in whether we use a deep learning based method to extract user's interested points in the documents. Recall that SEAN-END2END is an end-to-end model that combines the three processes, pointing users' interested points, constructing personalized document representation, and predicting users' clicking. However, SEAN-KEYWORD uses non-deep learning based methods to first find keywords coarsely, a much faster and scalable approach to attending users' interests, then applies the deep learning based approaches for representation and prediction. In general, SEAN-END2END achieves higher effectiveness while SEAN-KEYWORD achieves higher efficiency.

7 Experiments

In this section, we present our experimental results. We firstly introduce dataset description, evaluation metrics, baseline comparison, and experimental settings. Then we show the performance comparisons between our models and baselines, followed by extensive study of our models, including different social exploration methods, ablation study, hyper-parameters, and scalability analysis.

7.1 Dataset Description

We build two datasets, Steemit-English and Steemit-Spanish from the decentralized social platform, Steemit. Steemit is a blogging and social networking platform that uses the Steem blockchain to reward creators and consumers.

	${\bf Steemit-English}$	${\bf Steemit-Spanish}$
Duration (days)	370	126
# Consumers	7,242	1,396
# Creators	44,220	4,073
# Relations	273,942	25,593
# Documents	177, 134	14,843
Avg. word per document	290	509
# Logs	220,909	20,893
# Samples	684,752	92,236

Table 3 Statistics of the two datasets.

Most of the modern content distribution platforms are already using recommendation systems to recommend contents to users, which can be biased if we collected data from them for our evaluation. Different from them, the contents and user clicks are not manipulated by the Steemit platform. We retrieve the commenting activities of users (consumers) from June 2nd, 2017 to July 6th, 2018. Two datasets are constructed based on social communities using English and Spanish respectively.

We form a sample as a triplet with three elements: a given user, a document, and a label 1/0. We form the positive samples by the documents in which users have made comments. We treat messages that users' friends have made responses but the users themselves do not as negative samples. Since we collect users' activity information from their comment logs, it is natural that the number of users who made comments on this platform is not too much. The statistics of the two datasets are shown in Table 3.

7.2 Evaluation Metrics

To evaluate the recommendation quality of the proposed approach, we use the following metrics: Area under the Curve of ROC (AUC) and F1 for consumer satisfaction and the Gini coefficient for creator equality, where Gini coefficient is defined as:

$$\text{Gini} = \frac{\sum_{i=1}^{n} (2i - n - 1)x_i}{n \sum_{i=1}^{n} x_i},$$
(21)

where, x is an observed value, n is the number of values observed, and i is the rank of values in ascending order. To measure the performance of models considering both creators and consumers, we calculate the harmonic mean of F1 and (1-Gini), denoted as C&C:

$$C\&C = \frac{2 \times (1 - \text{Gini}) \times \text{F1}}{(1 - \text{Gini}) + \text{F1}}.$$
(22)

Since we train and test day-by-day, we compare a model's quality by the average of each metric during the whole period. For AUC, F1, and C&C, the larger, the better. For Gini, the smaller, the better.

7.3 Baselines

We compare our model with following baselines.

LR (Van den Oord et al. 2013) is the simplest word-based model for CTR prediction. We use TF-IDF to extract keywords for a user's clicked historical documents and the new incoming document and feed them to a logistic regression model to predict the label.

LibFM (Rendle 2012) is a state-of-the-art feature-based factorization model and widely used in CTR prediction. In this article, we use the same features as LR and feed them to LibFM. LibFM treats a user's features and a document's features separately for the factorization.

DKN (Wang et al. 2018b) learns representations of documents and users. In DKN, it obtains a set of embedding vectors for a user's clicked historical documents. Then an attention is applied to automatically match the candidate document to each piece of his/her clicked documents, and aggregate them with different weights. Here, we only use DKN's base model without the knowledge graph information.

NCF (He et al. 2017) is short for Neural network based Collaborative Filtering. It is a deep model for recommender systems that uses a multi-layer perceptron (MLP) to learn the user—item interaction function. It ignores the content of news and uses the comment counting information as input.

SAMN (Chen et al. 2019a), Social Attentional Memory Network, is a collaborative filtering model that employs the attention-based memory module to attend to a user's one-hop friends' vectors. The attention adaptively measures the social influence strength among friends.

SEAN-END2END & SEAN-KEYWORD, are two versions of SEAN proposed in the article. If without any clarification, we use F1 score as the exploitation value.

7.4 Experimental Settings

For our framework, we use pre-trained word embeddings for the document and fix them during training for both two versions of SEAN. In SEAN-END2END, for the word-level representation in the CNN layer, the filter number is set as 50 for each of the window sizes ranging from 1 to 6. The hidden vector size is set to 128 for both GRU layers and dense layers. The beam width B is set to 3 and λ is set to 1. The search depth L is set to 10. We train the model for 6 epochs every day.

In SEAN-KEYWORD, We set the number of keywords retrieved by TF-IDF to 90 to represent the document, and 200 to depict the user's reading history on both Steemit-English and Steemit-Spanish. The size of the hidden vector is set to 64 for all dense layers. The epoch is set to 5 for everyday training.

The key parameter settings for baselines are as follows. For the keyword extraction in LR and LibFM, we set the number of keywords for document and

user's historical readings as 20 and 90. For DKN, the length of the document embedding is set to 200. Due to the limitation of memory, we use a user's latest 10 clicked documents to represent the user. For the CF based methods, NCF and SAMN, the embedding size of users and the documents are all set to 128.

The above settings are for fair consideration. Each experiment is repeated five times, and we report the average and standard deviation as results. The data every day is split to 9:1 for training and validation. We train the model from the data of past t days and test it by using the data on t + 1-th day. The whole model is implemented on Keras 2.0 with Tensorflow 1.12 as the backend, based on CUDA 7.5 using a single GPU, GeForce GTX 1080 with 8GB VRAM.

7.5 Comparison with Baselines

Table 4 reports the results on Steemit-English and Steemit-Spanish datasets. For consumers, SEAN-END2END improves F1 by above 5 percentage points and AUC by near 3 percentage points compared with the best content-based model DKN on Steemit-English and improves F1 by 1.7 percentage point and AUC by near 3 percentage point on Steemit-Spanish. This proves that our model can best consider consumer's interests and recommend the most interesting contents to them. LR and LibFM perform much worse because these two models ignore the word order information and consequently generate worse document and user representations. Moreover, compared with CF-based models (NCF and SAMN), SEAN-END2END can also outperform them significantly. This result shows that CF methods cannot work well in this recommendation scenario since the documents on Steemit is highly time-sensitive, and the content should be considered for the recommendation. Besides, from the comparison with the SAMN, we can see that our strategy to incorporate social information is more effective than SAMN.

For creators, the Gini coefficients of the content-based models are smaller than those of the CF-based models. The result proves our aforementioned claim that CF methods are more likely to suffer from Matthew's effect since CFbased models intend to use global behavioral information to promote popular documents on the social platform. The Gini coefficients of SEAN-END2END and DKN are comparable, which shows that under the premise of the quality of recommendation for consumers, our algorithm can also encourage creator's equality which may further encourage creators to stay on the platform to keep publishing their innovative contents.

From the harmonic mean C&C results, we observe that SEAN-END2END performs best on both datasets. This demonstrates that the social exploration mechanism can have a good balance on optimizing between consumer satisfaction and creator equality.

For SEAN-KEYWORD, it is worse than DKN and SEAN-END2END for both consumers (AUC, F1 score) and creators (Gini coefficients). However, it

Dataset	Model	AUC	$\mathbf{F1}$	Gini	C&C
	NCF	$52.83 {\pm} 0.13$	42.14 ± 0.21	$66.04 {\pm} 0.25$	$37.71 {\pm} 0.22$
	SAMN	53.05 ± 0.35	42.28 ± 0.45	$65.98 {\pm} 0.21$	$37.80 {\pm} 0.28$
	LR	$52.89 {\pm} 0.07$	34.50 ± 0.11	62.89 ± 0.11	$35.86 {\pm} 0.11$
English	LibFM	50.01 ± 0.12	40.43 ± 0.22	66.42 ± 0.13	$36.79 {\pm} 0.16$
-	DKN	62.71 ± 0.22	42.85 ± 0.45	62.29 ± 0.26	40.22 ± 0.33
	SEAN-KEYWORD	55.59 ± 0.39	42.96 ± 0.45	64.00 ± 0.25	39.17 ± 0.32
	SEAN-END2END	$65.57 {\pm} 0.17$	$47.69 {\pm} 0.46$	$61.78 {\pm} 0.24$	$42.43{\pm}0.33$
	NCF	50.46 ± 0.21	35.02 ± 0.26	58.13 ± 0.34	38.14 ± 0.29
	SAMN	51.10 ± 0.24	35.24 ± 0.31	58.29 ± 0.32	$38.20 {\pm} 0.31$
	LR	53.15 ± 0.06	36.50 ± 0.29	$55.84 {\pm} 0.09$	39.97 ± 0.14
Spanish	LibFM	47.71 ± 0.30	22.37 ± 0.33	56.50 ± 0.21	29.55 ± 0.26
*	DKN	57.02 ± 0.39	41.27 ± 0.45	$53.98{\pm}0.25$	43.52 ± 0.32
	SEAN-KEYWORD	55.83 ± 0.29	41.04 ± 0.34	58.19 ± 0.23	41.42 ± 0.32
	SEAN-END2END	$59.98{\pm}0.34$	$42.99 {\pm} 0.37$	$53.99 {\pm} 0.23$	$44.46{\pm}0.28$

Table 4 Comparison of different methods on Steemit. The best results are highlighted inboldface.

outperforms NCF, SAMN, LR, and LibFM. The reason would be that both DKN and SEAN-END2END use more complicated components (LSTM, CNN) to encode the document. Comparing two SEAN models, it indicates that an end-to-end model to get user's interests for the final prediction can get better performance than first retrieving user's keywords then prediction.

7.6 Different Strategies in Social Exploration

In the experiment, we evaluate the performance of each exploitation-exploration method using the Steemit-English dataset. Since the social exploration strategy is used in SEAN-END2END and SEAN-KEYWORD without any difference, we only show the results with SEAN-END2END for simplicity. "Random Select" is the model that randomly selects a set of users on the social platform as the target user's friends. "Random Walk" is the model that uses a stochastic process, moving from a node to another adjacent node. These two models are both using a random based strategy to explore. As shown in Table 5, MCTS based models have better F1 than random based models, because the exploitation mechanism can help the model find more relevant friends. SEAN-RS-F1 performs the best on F1 because this model tends to explore friends of higher quality continuously by directly using the recommendation feedbacks. The F1 performance of SEAN-SPR and SEAN-DPR are compatible, while both are worse than the others since SEAN-SPR uses the static social network and SEAN-DPR only uses the daily comment network formed by consumer-creator connections, both missing some information. Moreover, MCTS based models also outperform random based models on C&C, which indicates that our model can improve the recommendation quality for consumers even though slightly hurts the equality. Specifically, SEAN-Payout has the highest C&C which indicates that using payout, the rewards given by Steemit as the exploitation value, is more suitable to select friends on this platform. Meanwhile, it further verifies the decentralized nature of this platform.

Models	F1	Gini	C&C
Random Select	$42.48 {\pm} 0.38$	$59.13{\pm}0.22$	$41.09 {\pm} 0.28$
Random Walk	$45.05 {\pm} 0.39$	$60.98 {\pm} 0.09$	$41.77 {\pm} 0.20$
SEAN-RS-F1	$47.69{\pm}0.46$	$61.78 {\pm} 0.24$	$42.43 {\pm} 0.33$
SEAN-SPR	$45.99 {\pm} 0.35$	$60.90 {\pm} 0.32$	42.27 ± 0.33
SEAN-DPR	$45.96 {\pm} 0.44$	$60.98 {\pm} 0.22$	42.21 ± 0.29
SEAN-Payout	$46.26 {\pm} 0.36$	$60.65 {\pm} 0.40$	$42.53 {\pm} 0.37$
ϵ -Greedy	$43.95 {\pm} 0.42$	$59.63 {\pm} 0.19$	$42.08 {\pm} 0.20$

Table 5 Comparison of social exploration methods.

Fig. 5 Time Comparison between SEAN-END2END and SEAN-KEYWORD.



 ϵ -Greedy is SEAN with the strategy ϵ -Greedy mentioned in Section 3.2. Compared with MCTS based models, it performs worse on F1 while better on Gini. It is because that ϵ -Greedy has a certain probability to randomly select from neighbors, a more random way to explore new friends. The C&C shows that MCTS is a smarter and more dynamic way to balance exploitation and exploration compared with ϵ -Greedy.

7.7 Compare Running Time of Two SEAN Models

We give the running time comparison of two SEAN models. As shown in Figure 5, the running time of SEAN-KEYWORD is much faster than SEAN-END2END since there are no complex modules in SEAN-KEYWORD, e.g., CNN or LSTM, expect for simple attention modules (word attention to social attention). Another reason should be that using top-K keywords is efficient than an end-to-end manner to point out users' reading keywords. In other words, SEAN-KEYWORD is more suitable to handle very large-scale data in real-world industrial scenarios.

Table 6 Comparison of different variants for socialization.

7.8 Model Ablation Study

7.8.1 Variants of Socialization

We compare variants of the socialization component in SEAN in terms of the following aspects to demonstrate the efficacy of the framework design: the use of social connections, the use of social attention, the use of friend exploration. Same as in Section 7.6, we only conduct experiments with SEAN-END2END for simplicity, and the results are shown in Table 6.

For the consumer side, we can conclude as follows.

- Without any social information means that we are using a pure personalization model for each user. This will decrease F1 by 5 percentage points. This confirms the efficacy of using social information in the SEAN-END2END.
- We also replace social attention with simple averaging friends' representation vectors. This results in a loss of F1 by near 3 percentage points. In other words, it demonstrates the effectiveness of weighing different social influences from friends on recommendation performance.
- We use each user's first-order (one-hop) friends for socialization. This is also worse than SEAN-END2END with exploring high-order friends, which proves the importance of exploring friends for recommendation.

For the creator side, the Gini coefficient of SEAN-END2END w/o social is the lowest one, followed by SEAN-END2END with one-hop friends. The reason is that without using any social information, users are not influenced by other users' reading histories, thus cutting off the spread of popular documents. Besides, using the first-order connections is worse than using high-order social information.

7.8.2 Variants of SEAN-END2END

We further compare different components in the hierarchical document representation of our SEAN-END2END model to demonstrate the efficacy of the RS model design. The results are shown in Table 7. Specifically, we test how CNN for word-level and GRU for sentence-level encoding affect the performance. The usage of CNN and GRU brings about 2 percentage points to gain on F1 respectively. Without using GRU and CNN decreases F1 by more than 3 percentage points. For the creator side, for models without GRU and/or CNN components, Gini drops within 2 percentage points while F1 also drops. The

Variants	F1	Gini	C&C	
w/o CNN	$45.25 {\pm} 0.22$	$59.98{\pm}0.26$	$42.58 {\pm} 0.21$	
w/o GRU	$45.07 {\pm} 0.31$	$60.47 {\pm} 0.14$	$42.12 {\pm} 0.27$	
w/o CNN & GRU	$44.08 {\pm} 0.26$	$60.06 {\pm} 0.20$	$41.91 {\pm} 0.25$	
SEAN-END2END	$47.69{\pm}0.46$	$61.78 {\pm} 0.24$	$42.43{\pm}0.33$	

Table 7Comparison of different variants of SEAN-END2END.

Table 8 Comparison of various similarity functions in SEAN-KEYWORD.

Attention Type Similarity Function		Parameter	F1	Gini	C&C
Dynamic	Self-Attention	-	42.96	64.00	39.17
	Cosine	-	42.35	63.70	39.09
	Polynomial	$\gamma = 0.5, d = 2, c = 1,$	42.61	63.66	39.23
	Sigmoid	$\gamma = 0.5, c = 1$	42.80	63.68	39.29
	RBF	$\gamma = 0.5$	34.10	61.75	36.06
Static	Euclidean	-	34.24	62.00	36.02
	Exponential	$\gamma = 0.5$	34.21	61.85	36.07
	Manhattan	-	35.20	62.11	36.50
	GESD	$\gamma = 0.5, c = 0.1$	35.02	61.97	36.46
	AESD	$\gamma=0.5, c=0.1$	33.71	61.86	35.79

best result of C&C indicates that our model can obviously improve consumers' satisfaction without hurting equality too much.

7.8.3 Variants of SEAN-KEYWORD

We conduct experiments on a variety of social attention approaches on Steemit-English. As results shown in Table 8, the dynamic attention can get the best F1 and the RBF similarity can get the best Gini. Besides, RBF achieves the best performance on C&C. In General, the results of dynamic attention, cosine similarity, polynomial similarity, sigmoid similarity are comparable and better than others, which indicate they are more suitable to compute the influence of different friends.

7.9 Hyper-parameter Sensitivity

SEAN involves many hyper-parameters, including SEAN-END2END and SEAN-KEYWORD. In the following experiments, except for the parameter being tested, all other parameters are set as introduced in Section 7.4 if we do not point out. The parameter sensitivity is done by using the samples from Steemit-English during the first 100 days.

7.9.1 For Friend Selection

We test the hyper-parameters in MCTS. Since the friend selection parts in SEAN-END2END and SEAN-KEYWORD are the same, we test only on SEAN-END2END for simplicity.

- Search Depth L. We test the influence of search depth L for four proposed models with L = 5, 10, 15, 20, 25. The results are shown in Figure 7(a). Given the best settings shown in Section 7.4, changing L from 5 to 25 does not affect both F1 and Gini a lot compared to the beam width B. This may indicate that given the Steemit network and the prediction F1 score, using a small number of friends can already cover most of the friends to explore while increasing B will force the exploration to find more neighbors.
- Beam width B. We investigate the influence of the beam width B (number of paths) by setting B ranging from 2 to 10. The results are shown in Figure 7(b). We can see that F1 increases as the beam width grows since there are more selected friends that are helpful for the user. While with a continuing increase of B, F1 tends to be flat since the overlapping of friends selected from each path also increases. Meanwhile, Gini continuously increases when the beam width increases, and thus C&C appears to be best only when B is 4.
- **Trade-off constant** λ . The choice of the trade-off constant λ is set to be $\lambda \in \{0.01, 0.1, 1, 10, 100\}$. We can see in Figure 7(c), the best F1 is at $\lambda = 1$ for all approaches. It indicates that both exploration and exploitation are important to better select friends. Besides, Gini is less influenced by λ .
- ϵ -Greedy. We also test the influence of different ϵ for using ϵ -Greedy algorithm in SEAN-END2END to explore new friends. Here, we set ϵ with $\{0.1, 0.3, 0.5, 0.7, 0.9\}$. The results are shown in Figure 7. The best C&C is obtained when $\epsilon = 0.7$. When $\epsilon = 0.1$, the F1 score is much worse since it has less probability to explore new friends, further demonstrating the importance of balancing exploitation and exploration.

7.9.2 For SEAN-END2END

- Hidden Size and User Embedding Size h. We first investigate how the hidden size h affect the performance by testing h in set $\{20, 50, 70, 100\}$. The results are shown in Figure 9(a), from which we can observe that all four models obtain best F1 when h = 128. Changing h does not affect too much Gini scores. The trend of C&C is similar to the trend of F1, also getting the best result when h = 128.
- The number of kernels K and sizes of filters r. We investigate the number of kernels K and the choice of filter sizes r for CNN in SEAN. As shown in Figure 9(b), the F1 score generally increases as the number of kernels K with different convolutional windows g gets larger, since more kernels can capture long-distance patterns in sentences. Due to the limitation of time and memory, we do not further enlarge the K. Meanwhile, the influence of K on Gini is smaller than on F1. SEAN-F1-RS performs best on F1 while performs worst on Gini. We can get the best C&C for all the proposed models except SEAN-RS-F1 when K = 6. Likewise, we can observe similar rules for the filter size r, shown in Figure 9(c): a small



Fig. 6 Hyper-parameter sensitivity analysis on Steemit-English for Socialization.

Fig. 7 Hyper-parameter sensitivity analysis on Steemit-English for ϵ -Greedy.



filter size cannot capture more local patterns in sentences, while a large filter size may easily suffer from overfitting. The Gini increases with the increasing of filter size r. The best C&C results are obtained when r = 50.

7.9.3 For SEAN-KEYWORD

- # of user keywords m. We test m in $\{50, 100, 150, 200, 250\}$ for SEAN-KEYWORD on Steemit-English for 100 days. From Figure 9(d), we can see that F1 trend of m is that too few or too many words can hurt the F1 of the proposed SEAN-KEYWORD model since fewer words limit the expressive power while more words can include more noises.



Fig. 8 Hyper-parameter sensitivity analysis on Steemit-English for SEAN-END2END.

- # of document keywords n. The number of document keyword n is set to $\{50, 100, 150, 200, 250\}$ for SEAN-KEYWORD. From Figure 9(e), we can see that the influences of n on F1 are limited. The reason might the that when n > 90, these keywords contain nearly the entire information of each document. Besides, we can obtain the highest F1 and C&C at r = 90, despite the gap is quite small.

7.10 Scalability Analysis

The training complexity of SEAN-END2END and SEAN-KEYWORD are both dominated by the search depth L and the beam width B as explained in Section 6.1. Since we use the same strategy of social exploration for two models, we only present the scalability analysis on SEAN-END2END for simplicity. As shown in Figure 10, the time consumption of SEAN-END2END with four different exploitation methods is almost linear to L and B.

8 Related Work

8.1 Content Recommendation

For content recommendation, both content based approaches and CF based approaches have been studied. Content based approaches use a user's histor-



Fig. 9 Hyper-parameter sensitivity analysis on Steemit-English for SEAN-KEYWORD.



Fig. 10 Running time w.r.t. L and B.

ical reading contents to represent a user, so they are naturally personalized models. In (Huang et al. 2013), multi-layer neural networks are used to learn embeddings for a given query and related documents. In (Okura et al. 2017), an end-to-end embedding-based method was proposed to use distributed representations for recommendation. (Wang et al. 2018b,a) apply a knowledge graph to extract latent knowledge-level connections among contents for better exploration. Moreover, in (Wu et al. 2019a), the authors propose a neural news recommendation model with personalized attention. CF based approaches, including traditional ones (Das et al. 2007; Hu et al. 2008; Koren 2008; Koren et al. 2009; Liu et al. 2010; Rendle 2012; Lee et al. 2016; Zhao et al. 2017) and deep learning based models (Wang et al. 2015; Covington et al. 2016; He et al. 2017; Li and She 2017; Sun et al. 2018; Chen et al. 2019a), are also usu-

ally called personalized recommendation, as they consider a user's personal preference based on the user's behaviors or actions on the platform.

8.2 Social Recommendation

In the last decade, social connections have been proved helpful for modeling users' preferences and improving the performance of recommendation. Most of these approaches are based on the social homophily theory (McPherson et al. 2001) that people with similar preferences tend to be connected as friends. Therefore, related works to learn users' preferences in RS tend to focus on exploiting social relations to alleviate data sparsity (Jiang et al. 2012). These models could be roughly summarized into the following two categories: classical social recommendation and graph neural networks (GNN) (Wu et al. 2019b) in RS. The former ones only use first-order neighbors, most of which perform social regularization in both CF models (Ma et al. 2011; Wang et al. 2013; Hu et al. 2015; Zhao et al. 2017, 2019) and content-based models (Jiang et al. 2012). However, social regularization still assumes static social influences between a user and his/her friends. Recent works also apply social attention (Chen et al. 2019a; Song et al. 2019; Wang et al. 2019b; Xu et al. 2019) to calculate different influence strength among neighbors.

GNN-based models applied in RS often perform message passing to aggregate the neighborhood information, such that the K-th order social information is captured with K iterations/layers. DiffNet (Wu et al. 2019b) proposes to use influence propagation structure to model the recursive dynamic social diffusion. GraphRec (Fan et al. 2019) combines user embedding propagation with attention networks to jointly capture interactions and opinions in the user-item graph. The main difference between our model and others with high-order neighbors is that SEAN designs several new strategies based on random walk to incorporate high-order friends while GNN-based models have to incorporate all neighbors layer-by-layer, which spend much more computational space. Besides, SEAN considers both consumer satisfaction and creator equality to select the most relative friends while GNN-based approaches only consider the similarity between the target user and his/her neighbors.

8.3 Exploitation-Exploration

The exploration-exploitation trade-off is a popular problem which happens in some situations where the model repeatedly learns to make decisions with uncertain pay-offs. This idea is not only highly applied in reinforcement learning (Mnih et al. 2015), but also in other scenarios such as online advertising (Li et al. 2010b).

Exploitation-Exploration of items is also a hot topic in RS field (Joachims et al. 1997; Radlinski et al. 2008; Li et al. 2010a; Wang et al. 2014; Liebman et al. 2017; Zheng et al. 2018; McInerney et al. 2018; Chen et al. 2019b). Exploring more items can introduce more diversity in recommendation results.

However, they still only use the click-through rate (CTR) to evaluate their models. That means most of them still focus on optimizing the performance of recommendation, which only benefits consumers and the platform. Moreover, they are still working on traditional user-item based collaborative filtering settings. There is a lack of studies on content recommendation and focusing on the creators of the contents. To our knowledge, we are the first work that considers using the Gini coefficient combined with F1 as a core metric to evaluate different recommendation algorithms. Some existing work, such as (Salganik et al. 2006), has used the Gini coefficient to evaluate Matthew's Effect of a RS. However, they have not simultaneously considered recommendation performance. In addition to recommendation algorithms, a market-based mechanism design, e.g., (Wei et al. 2005), is considered as an orthogonal perspective to improve an RS. The related studies to improve the item equality have been shown in (Abeliuk et al. 2017; Berbeglia and Hentenryck 2017).

8.4 Attention Mechanisms

The attention mechanisms have been proved effective in various tasks such as image classification (Xu et al. 2015), language model, machine comprehension (Luong et al. 2015), and machine translation (Bahdanau et al. 2015), due to its reasonable assumption that human only focus on selective parts of the whole perception space according to specific tasks. Recently, a hierarchical attention model (Yang et al. 2016) is proposed to capture patterns of feeds from the word level to the sentence-level and then to the whole documents for the task of document classification. Besides, (Veličković et al. 2018) presents graph attention networks (GATs) to attend over the friends' features of every node in the citation network. In the field of recommendation, (Chen et al. 2017) introduces both component-level and item-level attention into a CF framework for the multimedia recommendation. (Xiao et al. 2017) improves factorization machine (FM) by learning the importance of different feature interactions via a neural attention network. KGAT (Wang et al. 2019a) explicitly models the high-order relations in collaborative knowledge graph by using graph attention networks.

9 Conclusions and Future Work

In this article, we present a social recommendation model, SEAN, which goes beyond personalization by exploring higher-order friends in a social network to help content recommendation. In the model design and the exploration design, we consider the effects for both content creators and consumers on the social media platform. This can benefit the platform to attract more innovative content creators and encourage more interactions between the creators and consumers. We use datasets derived from a decentralized content distribution platform, Steemit, to evaluate our proposed framework. Experimental results show that we can improve both the creator's equality and consumer's satisfaction in content recommendation.

For future work, we plan to design a more fine-grained approach to select the higher-order friends for users. Specifically, a complete end-to-end model that combines the friends selecting and the recommendation process. In addition, we notice that our method (and all literature) focuses on modeling single side information in recommendation, i.e., social connections. Therefore, another interesting direction of future work is to investigate how to design an algorithm to well combine various side information to improve recommendation.

Acknowledgements

The authors of this paper were supported by NSFC (U20B2053), Hong Kong RGC including Early Career Scheme (ECS, No. 26206717), General Research Fund (GRF, No. 16211520), and Research Impact Fund (RIF, No. R6020-19), and WeBank-HKUST Joint Lab. This article was partially done when the first author was an intern at WeBank AI Department. We also thank the anonymous reviewers for their valuable comments and suggestions that help improve the quality of this manuscript.

References

- Abeliuk A, Berbeglia G, Hentenryck PV, Hogg T, Lerman K (2017) Taming the unpredictability of cultural markets with social influence. In: Proceedings of the 26th International Conference on World Wide Web
- Bahdanau D, Cho K, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: Proceedings of the 3rd International Conference on Learning Representations
- Berbeglia F, Hentenryck PV (2017) Taming the matthew effect in online markets with social influence. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence
- Chen C, Zhang M, Liu Y, Ma S (2019a) Social attentional memory network: Modeling aspect-and friend-level differences in recommendation. In: Proceedings of the 12th ACM International Conference on Web Search and Data Mining
- Chen H, Dai X, Cai H, Zhang W, Wang X, Tang R, Zhang Y, Yu Y (2019b) Large-scale interactive recommendation with tree-structured policy gradient. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence
- Chen J, Zhang H, He X, Nie L, Liu W, Chua TS (2017) Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In: Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval

- Covington P, Adams J, Sargin E (2016) Deep neural networks for youtube recommendations. In: Proceedings of the 10th ACM Conference on Recommender Systems
- Das AS, Datar M, Garg A, Rajaram S (2007) Google news personalization: scalable online collaborative filtering. In: Proceedings of the 16th International Conference on World Wide Web
- Fan W, Ma Y, Li Q, He Y, Zhao E, Tang J, Yin D (2019) Graph neural networks for social recommendation. In: Proceedings of the 28th International Conference on World Wide Web
- Grover A, Leskovec J (2016) node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
- He X, Liao L, Zhang H, Nie L, Hu X, Chua TS (2017) Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web
- Hu GN, Dai XY, Song Y, Huang S, Chen J (2015) A synthetic approach for recommendation: Combining ratings, social relations, and reviews. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence
- Hu Y, Koren Y, Volinsky C (2008) Collaborative filtering for implicit feedback datasets. In: Proceedings of the 8th IEEE International Conference on Data Mining
- Huang PS, He X, Gao J, Deng L, Acero A, Heck L (2013) Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management
- Jamali M, Ester M (2010) A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the 4th ACM conference on Recommender systems
- Jiang M, Cui P, Liu R, Yang Q, Wang F, Zhu W, Yang S (2012) Social contextual recommendation. In: Proceedings of the 21st ACM international conference on Information and knowledge management
- Joachims T, Freitag D, Mitchell TM (1997) Web watcher: A tour guide for the world wide web. In: Proceedings of the 15th International Joint Conference on Artificial Intelligence
- Kim Y (2014) Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing
- Kocsis L, Szepesvári C (2006) Bandit based monte-carlo planning. In: European conference on machine learning
- Koehn P (2004) Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In: Conference of the Association for Machine Translation in the Americas
- Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining

- Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. Computer 42(8)
- Lee J, Kim S, Lebanon G, Singer Y, Bengio S (2016) Llorma: local low-rank matrix approximation. Journal of Machine Learning Research 17(1):442–465
- Li L, Chu W, Langford J, Schapire RE (2010a) A contextual-bandit approach to personalized news article recommendation. In: Proceedings of the 19th international conference on World Wide Web
- Li W, Wang X, Zhang R, Cui Y, Mao J, Jin R (2010b) Exploitation and exploration in a performance based contextual advertising system. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining
- Li X, She J (2017) Collaborative variational autoencoder for recommender systems. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining
- Liebman E, Khandelwal P, Saar-Tsechansky M, Stone P (2017) Designing better playlists with monte carlo tree search. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence
- Liu J, Dolan P, Pedersen ER (2010) Personalized news recommendation based on click behavior. In: Proceedings of the 15th international conference on Intelligent user interfaces
- Luong T, Pham H, Manning CD (2015) Effective approaches to attentionbased neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing
- Ma H, Zhou D, Liu C, Lyu MR, King I (2011) Recommender systems with social regularization. In: Proceedings of the 4th ACM international conference on Web search and data mining
- McInerney J, Lacker B, Hansen S, Higley K, Bouchard H, Gruson A, Mehrotra R (2018) Explore, exploit, and explain: personalizing explainable recommendations with bandits. In: Proceedings of the 12th ACM Conference on Recommender Systems
- McPherson M, Smith-Lovin L, Cook JM (2001) Birds of a feather: Homophily. Annu Rev Sociol 27:415–44
- Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, et al. (2015) Human-level control through deep reinforcement learning. nature 518(7540):529–533
- Okura S, Tagami Y, Ono S, Tajima A (2017) Embedding-based news recommendation for millions of users. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
- Van den Oord A, Dieleman S, Schrauwen B (2013) Deep content-based music recommendation. In: Proceedings of the 27th Conference on Neural Information Processing Systems
- Radlinski F, Kleinberg R, Joachims T (2008) Learning diverse rankings with multi-armed bandits. In: Proceedings of the 25th international conference on Machine learning
- Rendle S (2012) Factorization machines with libfm. ACM Transactions on Intelligent Systems and Technology 3(3):57

- Rigney D (ed) (2010) The Matthew Effect, How Advantage Begets Further Advantage. Columbia University Press
- Salganik MJ, Dodds PS, Watts DJ (2006) Experimental Study of Inequality and Unpredictability in an Artificial Cultural Market. Science 311(5762):854–856
- Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D (2016) Mastering the game of go with deep neural networks and tree search. Nature 529:484–503
- Song W, Xiao Z, Wang Y, Charlin L, Zhang M, Tang J (2019) Session-based social recommendation via dynamic graph attention networks. In: Proceedings of the 12th ACM International Conference on Web Search and Data Mining
- Sun P, Wu L, Wang M (2018) Attentive recurrent social recommendation. In: Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval
- Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2018) Graph Attention Networks. In: Proceedings of the 6th International Conference on Learning Representations
- Wang H, Chen B, Li WJ (2013) Collaborative topic regression with social regularization for tag recommendation. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence
- Wang H, Wang N, Yeung DY (2015) Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
- Wang H, Zhang F, Wang J, Zhao M, Li W, Xie X, Guo M (2018a) Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management
- Wang H, Zhang F, Xie X, Guo M (2018b) Dkn: Deep knowledge-aware network for news recommendation. In: Proceedings of the 27th International Conference on World Wide Web
- Wang X, Wang Y, Hsu D, Wang Y (2014) Exploration in interactive personalized music recommendation: a reinforcement learning approach. ACM Transactions on Multimedia Computing, Communications, and Applications 11(1):7
- Wang X, Yu L, Ren K, Tao G, Zhang W, Yu Y, Wang J (2017) Dynamic attention deep model for article recommendation by learning human editors' demonstration. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
- Wang X, He X, Cao Y, Liu M, Chua TS (2019a) Kgat: Knowledge graph attention network for recommendation. In: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining

- Wang X, Zhu W, Liu C (2019b) Social recommendation with optimal limited attention. In: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
- Wei YZ, Moreau L, Jennings NR (2005) A market-based approach to recommender systems. ACM Trans Inf Syst 23(3):227–266
- Wu C, Wu F, An M, Huang J, Huang Y, Xie X (2019a) Npa: Neural news recommendation with personalized attention. In: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
- Wu L, Sun P, Fu Y, Hong R, Wang X, Wang M (2019b) A neural influence diffusion model for social recommendation. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval
- Xiang B, Liu Q, Chen E, Xiong H, Zheng Y, Yang Y (2013) Pagerank with priors: An influence propagation perspective. In: Proceedings of the 33rd International Joint Conference on Artificial Intelligence
- Xiao J, Ye H, He X, Zhang H, Wu F, Chua TS (2017) Attentional factorization machines: learning the weight of feature interactions via attention networks. In: Proceedings of the 37th International Joint Conference on Artificial Intelligence
- Xiao W, Zhao H, Pan H, Song Y, Zheng VW, Yang Q (2019) Beyond personalization: Social content recommendation for creator equality and consumer satisfaction. In: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
- Xu F, Lian J, Han Z, Li Y, Xu Y, Xie X (2019) Relation-aware graph convolutional networks for agent-initiated social e-commerce recommendation. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management
- Xu K, Ba J, Kiros R, Cho K, Courville A, Salakhudinov R, Zemel R, Bengio Y (2015) Show, attend and tell: Neural image caption generation with visual attention. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning
- Yang X, Steck H, Liu Y (2012) Circle-based recommendation in online social networks. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining
- Yang Z, Yang D, Dyer C, He X, Smola A, Hovy E (2016) Hierarchical attention networks for document classification. In: Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies
- Ye M, Liu X, Lee WC (2012) Exploring social influence for recommendation: a generative model approach. In: Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval
- Zhao H, Yao Q, Kwok JT, Lee DL (2017) Collaborative filtering with social local models. In: 2017 IEEE International Conference on Data Mining (ICDM)

- Zhao H, Zhou Y, Song Y, Lee DL (2019) Motif enhanced recommendation over heterogeneous information network. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management
- Zheng G, Zhang F, Zheng Z, Xiang Y, Yuan NJ, Xie X, Li Z (2018) Drn: A deep reinforcement learning framework for news recommendation. In: Proceedings of the 27th International Conference on World Wide Web