Popularity-Enhanced News Recommendation with Multi-View Interest Representation

Jingkun Wang
Peking University
Beijing, China
wjk@pku.edu.cn

Yipu Chen*
Beihang University
Beijing, China
eap@buaa.edu.cn

Zichun Wang
Peking University
Beijing, China
wangzichun_1997@pku.edu.cn

Wen Zhao
Peking University
Beijing, China
zhaowen@pku.edu.cn

ABSTRACT

News recommendation is of vital importance to alleviating information overload. Recent research shows that precise modeling of news content and user interests become critical for news recommendation. Existing methods usually utilize information such as news title, abstract, entities to predict Click Through Rate(CTR) or add some auxiliary tasks to a multi-task learning framework. However, none of them directly consider predicted news popularity and the degree of users' attention to popular news into the CTR prediction results. Meanwhile, multiple interests may arise throughout users' browsing history. Thus it is hard to represent user interests via a single user vector. In this paper, we propose PENR, a Popularity-Enhanced News Recommendation method, which integrates popularity prediction task to improve the performance of the news encoder. News popularity score is predicted and added to the final CTR, while news popularity is utilized to model the degree of users' tendency to follow hot news. Moreover, user interests are modeled from different perspectives via a subspace projection method that assembles the browsing history to multiple subspaces. In this way, we capture users' multi-view interest representations. Experiments on a real-world dataset validate the effectiveness of our PENR approach.

CCS CONCEPTS

• **Information systems → Recommender systems;** • **Computing methodologies → Natural language processing**.

KEYWORDS

News recommendation; Neural network; Multi-view learning

ACM Reference format:
Jingkun Wang, Yipu Chen, Zichun Wang, and Wen Zhao. 2021. Popularity-Enhanced News Recommendation with Multi-View Interest Representation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482462

*Corresponding author.

1 INTRODUCTION

News reading has become an indispensable part of most people's daily life. Many online news service providers, such as MSN News[1], collect news articles from various sources and recommend them to users who spend fragmented time reading popular or exciting news. However, massive news articles are popping up every day, and it is unrealistic for users to read all of them due to the time limit. Therefore, targeting user interest and making personalized news recommendations precisely becomes vital for news service providers to alleviate information overload. Matching user interests and candidate news lies on two key aspects. First, the representation of news needs to be modeled precisely. News usually contains a variety of textual information, such as titles, article bodies, categories, and knowledge graphs which have been more and more introduced into news recommendation. Titles use short and concise expressions to summarize the core information of the whole news, while the main body of the news is more detailed. In addition, the categories of news are also informative, a user usually clicks on the news of the same category out of interest. Various entities in the knowledge graph make the representation of articles richer. For example, celebrities are more attractive to users than ordinary people; two leaders signing an agreement will attract users' attention more quickly if they belong to countries that are hostile to each other. Second, it is crucial to deeply characterize user interests, including modeling interest representations from the user profile and click history. Moreover, the news is updated quickly, so traditional methods [28-30] will suffer from a severe cold start problem.

Existing news recommendation methods are usually based on matching the final user vector with the candidate news vector, so the core problem in these methods is how to learn representations of news and users. For instance, NAML [1] proposes an attentive multi-view learning model to learn unified news representations from titles, bodies, topic categories and applies attention mechanism to select important textual information and informative news. LSTUR [7] uses GRU network that generates

[1] https://www.msn.com/en-us/news

	Historical Browsed News
D1	MGM Resorts sells Circus, Bellagio on Las Vegas Strip
D2	20 Funny Things People in the 1970s Were Totally Guilty of Doing
D3	Subtle Signs You May Have Clogged Arteries
D4	See spectacular fall foliage in these national parks
D5	Singer stands up to heckler who told her to take shirt off
D6	Meghan McCain confronts Trump Jr.: 'You and your family have hurt a lot of people'
	Visited Candidate News
C1	College gymnast **dies** following training accident in Connecticut

Figure 1: Example of one user's click behavior from MSN News. Statistics show that news C1 has got 47003 hits, while there is no relevant news in the browsing history. The user visits C1 just because the news draws high social attention.

short-term representations of users from their recent browsing news to enhance representation of user interests. However, these methods ignore that the popularity of news can directly affect the CTR of news. For example, there is no news about "death" in one's click history, but when a piece of news about "death" appears on the list of candidates, the user will still click on it. In this case, we cannot regard "death" as a particular interest of the user, and it is clicked just because the news draws great social attention. Figure 1 illustrates this challenge with an example. Moreover, a user may have multiple interests according to his/her browsing history. Thus it is hard to represent user interests accurately via a single user vector.

In this paper, we propose PENR, a Popularity-Enhanced News Recommendation method, motivated by the issues mentioned above. We count the number of visits as popularity to each news article at first. Then we predict candidate news popularity score and add it to the final CTR to recommend hot news to users. Meanwhile, we control the proportion of news popularity in CTR by calculating users' attention to hot news. We also model user interests from different perspectives via a subspace projection method that aggregates browsing history to multiple subspaces. In this way, we capture the multi-view interest representation of a user. To summarize, we make the following three contributions:

- We propose a Popularity-Enhanced news recommendation method. We introduce popularity and it ultimately affects the click probability of candidate news. We attempt several approaches and determine the best way to integrate popularity into the model.
- We further propose a method to learn the multi-view interest representation of users, indicating user interests in different aspects. In addition, we match the proposed multi-view interest representation with candidate news through a bilinear function.
- Experiments on a real-world dataset MIND [14] which is an open-source version of MSN news prove that our approach achieves better performance on news recommendation than existing methods.

2 RELATED WORK

News recommendation aims to select news that users may be interested in from a large number of candidate news. In the traditional methods, LibFM [17] uses matrix factorization for recommendation; DeepFM [20] combines deep neural networks and factorization machines; Wide & Deep [19] inputs features into a wide model and a deep model; DSSM [18] inputs word features into a deep model for text matching. However, the previous methods cannot effectively represent news articles and user interests for news recommendation. To solve this problem, NAML [1] proposes an attentive multi-view learning model to learn unified news representations from titles, bodies, and topic categories and applies attention mechanism to select important textual information and informative news. NRMS [2] adds the text content of the news body when obtaining news representation and then uses multi-head self-attention to capture interactions between words and relevance of news.

More and more studies have recently introduced knowledge graphs in the recommendation process to obtain more informative news and user representations. DKN [3] introduces various information of the knowledge graph through a knowledge-aware convolutional neural network with multi-channel and word entity alignment. The news representation they generate contains the potential knowledge-level connections between news. DAN [4] proposes an attention module to enhance user representation. TANR [25] adds a topic prediction task to improve the capability of the news encoder. KRED [6] uses the knowledge graph to enrich the information of entities in the news and introduces the frequency, category, location of the entities to generate the context embedding of the news. MVL [5] models the interactions between different users and news to capture the user-user, news-news, and user-news relatedness in the user-news bipartite graphs with a graph attention network.

Some research focuses on how to model user interests better. LSTUR [7] introduces the GRU network to generate short-term representations of users from their recent browsing history. GnewsRec [8] constructs a heterogeneous graph and applies graph networks to learn user and news representations to encode high-order structure information while using an attention-based LSTM model to capture users' short-term interests. GNUD [9] designs a preference regularizer to enforce each disentangled embedding space to independently reflect an isolated preference, which could improve the quality of disentangled representations for users and news. With the user-news bipartite graph, GERL [10] learns better representations of users and news through a two-hop learning method. FIM [11] uses stacked dilated convolution to construct representations of the browsing news and interacts with candidate news instead of fusing click history into a single user vector. In addition, graph neural network has been widely used due to the characteristics of news recommendation

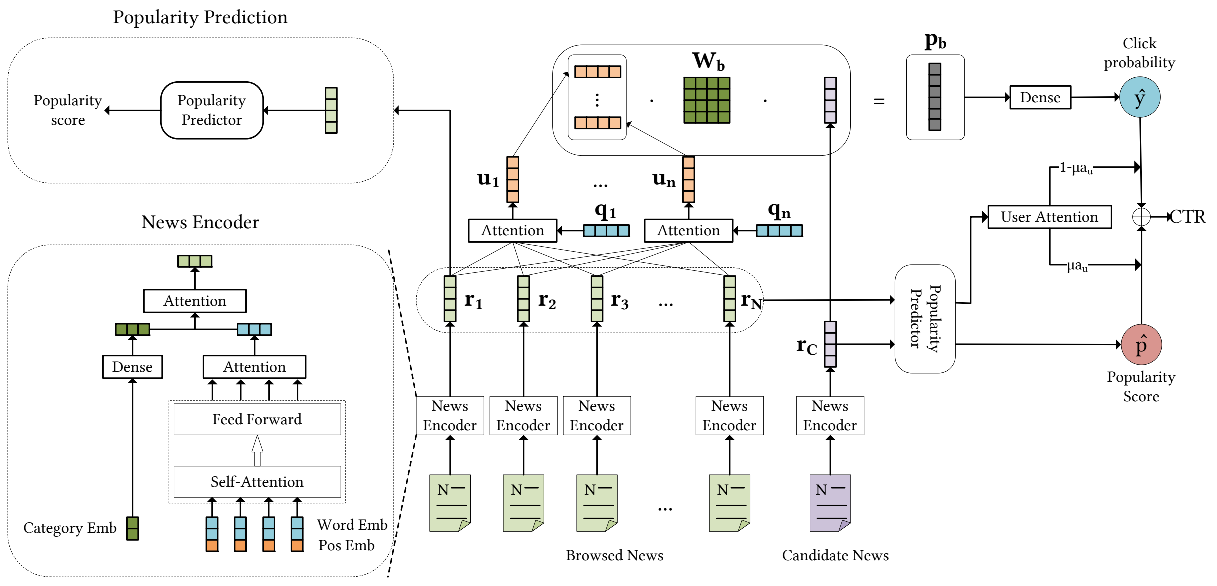


Figure 2: Overall Structure of PENR Model.

[26, 27].

3 PROPOSED APPROACH

The overall framework of our proposed method is illustrated in Figure 2. In order to make better use of the textual information in news articles, we add position encoding and feedforward network to the original multi-head self-attention layer.

3.1 News Encoder

Since news article contains rich textual information, it is essential to obtain high-quality text representation via the news encoder. In recent Natural Language Processing research, it has been proved that multi-head self-attention framework [12] captures more textual information than CNN. Thus in this paper, we use multi-head self-attention layer to encode news titles and abstracts. Given a news title T which consists of a word sequence $\{w_1, w_2, \ldots, w_n\}$ where n is the length of the title, we first convert each word into its low-dimensional vector using pre-trained word embeddings and get a vector sequence $\{e_1, e_2, \ldots, e_n\}$. Then we feed it to the multi-head self-attention layer, which is a particular case of attention mechanism. The multi-head self-attention layer is designed to capture different information from multiple perspectives. Explicitly speaking, we feed the vector sequence into the self-attention layer consisting of h attention heads and calculate the representation of the i-th word through the k-th attention head as follows:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}(\frac{\mathbf{QK}^T}{\sqrt{d_k}})\mathbf{V} \tag{1}$$

$$\text{head}_{i,k} = \text{Attention}(\mathbf{e}_i\mathbf{W}_k^Q, \mathbf{e}_i\mathbf{W}_k^K, \mathbf{e}_i\mathbf{W}_k^V) \tag{2}$$

Where $\mathbf{W}_k^Q, \mathbf{W}_k^K, \mathbf{W}_k^V$ are the projection parameters of the k-th attention head for query, key, and value respectively. Then we concatenate all the vectors generated by these attention heads and get a single vector using a linear projection matrix:

$$\mathbf{h}_i = \text{Concat}(\text{head}_{i,1}, \ldots, \text{head}_{i,h}) \tag{3}$$

$$\mathbf{H}_i = \mathbf{W}_p\mathbf{h}_i \tag{4}$$

The state-of-the-art framework NRMS only uses multi-head self-attention layer and an additive word attention network to form the news encoder. However, in the news recommendation task, the position of words in news text is crucial. Intuitively, keywords often appear in specific positions in news titles and abstracts. For instance, names of people and events often appear at the beginning of news titles. Therefore, we introduce position information to enhance the expressive ability of the news encoder, follow the Transformer framework:

$$\text{timing}(t, 2i) = \sin(t / 10000^{2i/d}) \tag{5}$$

$$\text{timing}(t, 2i+1) = \cos(t / 10000^{2i/d}) \tag{6}$$

In this way, we add the position information to the word embeddings in the news text through the sine and cosine function and does not introduce additional parameters.

Furthermore, after the multi-head self-attention layer, we feed the output vector $\mathbf{H}_i$ to a feed forward network:

$$\mathbf{m}_i = \text{ReLU}(\mathbf{H}_i\mathbf{W}_1)\mathbf{W}_2 \tag{7}$$

Considering that in news recommendation scenario, people are always attracted by a few specific keywords, so the importance of each word in the news text is different. It is evident that a few specific keywords can cause users to click on the news. Thus, we use an additive attention network to select keywords in news text. The attention weight of the *i*-th word in a news title is calculated as:

$$\alpha_i^t = \mathbf{q}_t^T \tanh(\mathbf{V}_t \times \mathbf{m}_i^t + \mathbf{v}_t) \tag{8}$$

where $\mathbf{V}_t$ is the projection matrix and $\mathbf{v}_t$ is the bias parameters, $\mathbf{q}_t$ is a random initialized attention query vector that is updated during training. Then we calculate the title representation by the weighted sum operation:

$$\mathbf{r}^t = \sum_{i=1}^{n} \alpha_i^t \mathbf{m}_i \tag{9}$$

In news recommendation, each news item has a category tag, so we concatenate the category embedding with the news title representation to form the final output of news encoder, where $\mathbf{c}$ denotes the category embedding.

$$\mathbf{r} = \mathrm{Concat}(\mathbf{r}^t, \mathbf{c}) \tag{10}$$

3.2 User Encoder

After getting the representation of the news, we need to model the user's representation based on his/her browsing history. In practical applications, the numbers of users and news are huge. Therefore, we are supposed to generate a universal user vector for a single user to improve computational efficiency rather than generate a specific user vector for each candidate news to improve accuracy. In other words, the calculation of the user vector is independent and should not be affected by candidate news. In this way, we propose our Multi-View Interest representation method to obtain user vectors from multiple perspectives.

In real-world datasets, a user's browsing history may consist of dozens or even hundreds of news, so it is difficult to use a single vector to express user interests completely and precisely. Supposed that a user clicks on a series of sports news and then clicks on several political news, a single vector cannot subtly express the user's interest in sports and politics separately using the existing methods, which simply mix the news representations in the user's click history. Thus, we obtain the Multi-View Interest representation by adding several independent attention networks, the i-th attention network calculates attention weights of the k-th browsing news as follows:

$$\alpha_{i,k} = \mathbf{q}_i^T \tanh(\mathbf{V}_i \times \mathbf{r}_k + \mathbf{v}_i) \tag{11}$$

Consequently, we obtain the final user representation of the *i*-th aspect by calculating the weighted sum of the browsing history. Then we concatenate the outputs of all the attention networks to form the final user representation $\mathbf{u}$:

$$\mathbf{u}_i = \sum_{k=1}^{N} \alpha_{i,k} \mathbf{r}_k \tag{12}$$

$$\mathbf{u} = \mathrm{Stack}(\mathbf{u}_1, \ldots, \mathbf{u}_{N_a}) \tag{13}$$

Where N_h is the number of news in the browsing history, N_a is the number of attention networks.

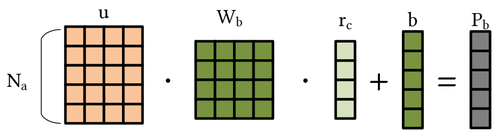


Figure 3: The Structure of bilinear click predictor.

3.3 Click Predictor

After getting user representation $\mathbf{u}$ and candidate news representation $\mathbf{r}_c \in \mathbb{R}^d$, the click predictor is used to calculate the probability of the user $\mathbf{u}$ clicking the candidate news $\mathbf{r}_c$. This step is essential to predict the interaction between two vectors. Inspired by biaffine attention framework, which is widely used in NLP tasks to acquire the interaction between two vectors, we obtain click probability $\mathbf{P}_b$ of candidate news by a bilinear function:

$$\mathbf{p}_b = \mathbf{u}\mathbf{W}_b\mathbf{r}_c + \mathbf{b} \tag{14}$$

where $\mathbf{u} \in \mathbb{R}^{N_a \times d}$ is the user vector generated by user encoder, $\mathbf{W}_b \in \mathbb{R}^{d \times d}$ is the biaffine matrix, $\mathbf{b}$ is the bias term. Thus, $\mathbf{p}_b \in \mathbb{R}^{d \times 1}$ indicates the click probability of candidate news in each interest subspace. We connect a feed forward network behind the bilinear layer in order to produce the final click probability $\hat{y}$.

$$\hat{y} = \mathrm{FFN}(\mathbf{p}_b) \tag{15}$$

3.4 Popularity Prediction Module

3.4.1 News popularity. The popularity of news is crucial to news recommendation. Users' click behaviors are caused by their interests and are affected by the popularity of news. The example in Figure 1 illustrates this phenomenon. Sometimes, when a significant event suddenly happens, we will see it everywhere on TV and the Internet. Under these circumstances, whether we are interested in the news or not, we will tend to click on it. In addition, we find that a small part of users have no browsing history, so it is hard for the previous methods to generate user vectors from empty click history. Thus, when we use the existing methods to predict CTR of this part of candidate news, they always perform random results. In our framework, we additionally predict the popularity of the candidate news, so that we can get credible results on this small part of data.

Therefore, how to express the degree of news popularity has become a core issue. Unfortunately, there is no explicit feature in the news dataset which indicates the popularity of news. Instead, we count the click behaviors in the dataset to approximate news popularity. Specifically, for each news that appeared in the dataset, we count the times it appears in all the users' click history. The more the news appears, the higher popularity the news has. After getting all the news' clicks, we divide them by the maximum value. In this way, we generate a column of new feature to the original dataset. Then we add the popularity prediction task as an auxiliary task to the base model. We use multi-task

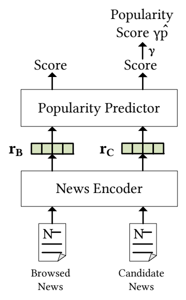


Figure 4: The structure of popularity predictor.

learning framework to train these two tasks. The popularity prediction module is shown in Figure 4.

In the training process, we feed the browsing history and candidate news into the news encoder to obtain their representation and then input these vectors into the popularity predictor. The popularity predictor predicts the popularity score of the news as follows:

$$\hat{p} = \sigma(\mathbf{W}_p \times \mathbf{r} + \mathbf{b}_p) \tag{16}$$

where $\mathbf{W}_p$ and $\mathbf{b}_p$ are the parameters of the feed forward network, while σ is the nonlinear activate function. Here we use sigmoid as the activate function. While in the forecasting process, we predict popularity a score $\hat{p}_c$ for candidate news, and then multiply it by a trainable parameter γ to generate a popularity value more suitable for this task. Finally, we add the popularity score $\gamma\hat{p}_c$ to the original click probability $\hat{y}$ produced by the click predictor to obtain CTR with news popularity.

$$\text{CTR} = \hat{y} + \gamma\hat{p}_c \tag{17}$$

3.4.2 Users' attention to hot news. Different users pay different attention to hot news. Users who only focus on their areas of interest may not click on hot news, while others may tend to click on hot news whether they are interested in the news or not. Therefore, simply adding a popularity score to CTR seems not reasonable enough. We attempt to calculate the attention a_u of user u to popular news as follows:

$$a_u = \frac{1}{n}\sum_{h=1}^{n} p_h \tag{18}$$

where p_h is the predicted popularity of news in the click history, while n is the number of clicked news. In this way, we obtain users' level of attention to hot news. Intuitively, users' high attention to popular news will lead to a high proportion of popularity score in CTR result, and vice versa. In the last step, we take users' attention as the weight of click probability and popularity score to acquire the final CTR result:

$$\text{CTR} = (1 - \mu a_u)\hat{y} + \mu a_u \gamma \hat{p}_c \tag{19}$$

where μ is a trainable parameter.

3.5 Model Training

Following the previous work, we use negative sampling techniques for model training. For each user, given his/her browsing history and impression log, we randomly sample K negative samples from the news that are not clicked for each positive sample. During training, we jointly predict the click probability score for both positive samples and negative samples. Given a user u and candidate news c, we compute the click probability score denoted as $\hat{s}_{u,c}$.

Then we take cross entropy as the base loss function for the training sample (u,c), $s_{u,c}$ denotes the ground truth label:

$$\ell_{rec} = -\frac{1}{N}\sum_{(u,c)\in S}(s_{u,c}\log(\hat{s}_{u,c}) + (1 - s_{u,c})\log(1 - \hat{s}_{u,c})) \tag{20}$$

where N is the total number of training samples, S is the training dataset.

Considering that the value of popularity is continuous, we use mean squared error(MSE) to formulate the loss function:

$$\ell_{pop} = \frac{1}{M}\sum_{i=1}^{M}(p_i - \hat{p}_i)^2 \tag{21}$$

where M is the number of news in the training dataset.

In the user encoder, we use several independent attention networks to generate multiple interests of users. Thus we hope each attention network learns a specific interest aspect, rather than all the attention networks get similar user vectors. In order to force different attention networks aggregating different browsing news and obtain user vectors in different subspaces of interest, we add a fully-connected layer to distinguish which subspace the user vector in $\mathbf{u} = \text{Stack}(\mathbf{u}_1, ..., \mathbf{u}_{N_a})$ belongs to:

$$\hat{d}_i = \text{softmax}(\mathbf{W}_{sub}\mathbf{u}_i + \mathbf{b}_{sub}) \tag{22}$$

where $\hat{d}_i \in \mathbb{R}^{N_a \times 1}$ indicates the probability that vector u_i belongs to each subspace. In this way, the discriminator proves that each vector has its own characteristics. Meanwhile, an auxiliary loss is formulated as follows:

$$\ell_{aux} = -\frac{1}{NN_a}\sum_{i=1}^{N}\sum_{j=1}^{N_a} d_{i,j}\log(\hat{d}_{i,j}) + (1 - d_{i,j})\log(1 - \hat{d}_{i,j}) \tag{23}$$

The overall loss function is the weighted sum of the above three loss functions:

$$\ell = \ell_{rec} + \lambda\ell_{pop} + \beta\ell_{aux} \tag{24}$$

where λ, β are both pre-defined coefficients.

4 EXPERIMENTS

4.1 Dataset and Settings

4.1.1 Dataset Introduction. Our experiments are conducted on a real-world dataset, which is constructed from the user click logs of Microsoft News, MIND[2] [14] contains 1 million users and more than 160k English news articles, each of which has rich textual content such as news title, abstract and body. In order to facilitate research in news recommendation, the MIcrosoft News Dataset (MIND) was published. It was collected from the user

[2] https://msnews.github.io

Table 1: Statistics of the dataset.

Dataset	# Users	# News	# Clicks
Train	711,222	101,527	3,383,656
Validation	255,990	72,023	574,845
Test	702,005	120,961	-
Total	876,956	130,379	-

behavior logs of Microsoft News, and the publisher randomly sampled 1 million users who had at least five news click records during six weeks from October 12 to November 22, 2019. The statistics of the dataset are shown in Table 1.

4.1.2 Evaluation metrics. In all the following experiments, we use AUC (Area Under the ROC Curve) as the main metric. Additionally, we use another two metrics which are commonly used in recommender system: MRR (Mean Reciprocal Rank) and NDCG (Normalized Discounted Cumulative Gain).

4.1.3 Experimental settings. We implement all the experiments based on Pytorch [24]. In our experiments, we use 300 dimensional pre-trained Glove embedding [15]; The category embedding and position embedding are both 300-dimensional; The number of attention heads in multi-head self-attention layer is 6; The attention query vector is 200-dimensional; The dropout rate is 0.2; The number of attention networks in user encoder is set to 5; In multi-task learning, the coefficient λ is set to 0.01, and the coefficient β is set to 0.001. The maximum lengths of news title and abstract are 20 and 50, respectively, and the length of browsing history is 50. The negative sampling ratio K is set to 4. Adam [16] is used for model optimization, and the learning rate is set to 0.0001. The batch size is set to 64. For a fair comparison, we only used news title, news abstract, and category to conduct the experiments. Although there are lot of extra information in the dataset, such as user/news id, news body, entities, etc., we do not incorporate this information as the previous works did.

4.2 Performance Evaluation

We compare PENR with the following methods, the results on the test set are summarized in Table 1, our model PENR consistently outperforms other baselines in terms of all metrics, including:

- DFM [21]: a deep fusion model, which utilizes an inception network to integrate models with different depths to capture the high order interactions between features.
- GRU [22]: a news recommendation model using an auto-encoder to learn news representations and a GRU network to learn user representations from the click history.
- DKN [3]: a model using Kim CNN to incorporate information in knowledge graph and enhance news representation.
- NPA [23]: a model which is based on personalized attention mechanism to learn user and news representations in different from different perspectives.
- NAML [1]: a news recommendation method with attentive multi-view to integrate various kinds of news information.
- FIM [11]: a model which uses Hierarchical Dilated Convolution to model fine-grained interests of users and calculates CTR via a cross matching and aggregation module.
- NRMS [2]: a model which uses multi-head self-attention to learn news representations from news text and learn user representations from all the news in the browsing history.
- KRED [6]: a model which integrates entity information and introduces multiple auxiliary tasks

Table 2: Performance comparison of different methods on the test dataset.

	AUC	MRR	nDCG@5	nDCG@10
Model without popularity				
DFM	0.6228	0.2942	0.3152	0.3722
GRU	0.6570	0.3150	0.3413	0.3984
DKN	0.6460	0.3132	0.3384	0.3948
NPA	0.6626	0.3215	0.3482	0.4051
NAML	0.6774	0.3290	0.3581	0.4149
NRMS	0.6788	0.3315	0.3602	0.4171
FIM	0.6834	0.3329	0.3629	0.4204
Model with popularity				
KRED	0.6852	0.3378	0.3676	0.4245
PENR	**0.6925**	**0.3416**	**0.3731**	**0.4304**
w/o *UA*	0.6899	0.3381	0.3690	0.4263
w/o *sum_pop*	0.6877	0.3362	0.3671	0.4245

For our PENR model, we conduct two extra experiments on which we remove User Attention (*UA*) module and popularity sum (*sum_pop*) module, respectively. In the first experiment, we directly add the predicted popularity of candidate news to the click probability without calculating users’ attention to popular news. In the second experiment, we only treat popularity prediction as an auxiliary task to enhance the ability of the news encoder and do not add predicted popularity to the final CTR, because most of the previous work (e.g., KRED) use news popularity in this way.

The results of different methods are shown in Table 2. It is evident that models with self-attention based text encoders significantly outperform models without self-attention. With the advanced techniques widely used in NLP tasks to generate news representation, we can vastly improve the model’s capability. Surprisingly, NAML achieves similar results to NRMS, which are better than all other baseline models except FIM and KRED, even though it only takes CNN as its news encoder. Our PENR model proposed in this paper improves the previous best model KRED

Table 3: Ablation studies on validation dataset. Performance declines when we remove the proposed modules.

	AUC	MRR	nDCG@5	nDCG@10
Base model	0.6803	0.3268	0.3651	0.4271
+ FFN&POS	0.6811	0.3325	0.3671	0.4309
+ Multi-view	0.6847	0.3360	0.3680	0.4293
+ Popularity	0.6940	0.3402	0.3762	0.4409
+ UA	0.6971	0.3410	0.3789	0.4421

by 1.1% in AUC score, and consistently outperforms in all the four metrics listed in Table 2. It is mainly because our model encodes textual information and browsing history of users better. Meanwhile, we introduce news popularity more appropriately. The results show that popularity prediction task provides the greatest benefit, while the other auxiliary tasks in KRED do not provide significant improvement.

The last three rows illustrate that removing *sum_pop* operation in our PENR model results in a significant decrease in AUC. It shows that incorporating popularity without adding it to the final CTR as previous work cannot take full advantage of the information contained in the popularity. In addition, utilizing User Attention (*UA*) to popular news to control the proportion of click popularity and news popularity has proved to be effective. It leads to a 0.38% improvement in AUC. These experimental results indicate that the way we extract and incorporate news popularity is reasonable. It is worth mentioning that the experiments in Table 2 are performed on the test set. However, due to the limitation of the test result submission rules, we cannot get the ground truth labels of the test dataset. Thus, for the convenience of experimental analysis, we conduct our experiments in the following sections on the validation dataset.

4.3 Ablation Study

Next, we evaluate how the proposed modules affect the performance of PENR model. The base model in our experiment is the same as NRMS, in which the news encoder only consists of a multi-head self-attention layer. First, we add position embedding and feed forward network with residual connection and layer normalization, denoting as FFN&POS. Second, we add multi-view interest representation to the model, then we obtain a user vector from different interest subspaces. Third, we add popularity prediction task to the model. Finally, we calculate users' level of attention to popular news and form the complete PENR model. The results are shown in Table 3. With each component adding to the original model, the model performance gets better and better. According to the result denoted as multi-view, multi-view interest representation allows the model to represent user interests in different aspects so that we can match user with candidate news from multiple perspectives. The fourth line represents the result of incorporating popularity, and it indicates that news popularity is highly related to news recommendation. Using news popularity correctly can significantly improve the accuracy of the model, the AUC is increased by 1.4%. The last line proves that calculating users' attention to popular news helps improve model accuracy.

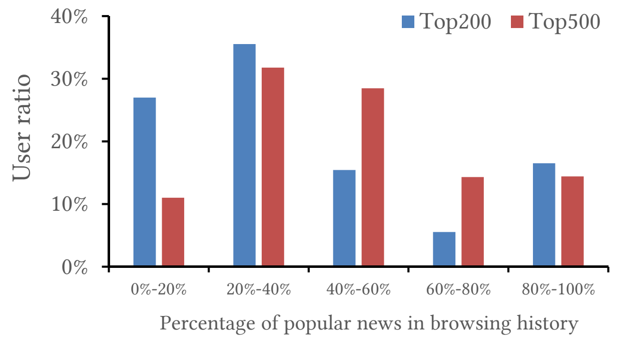
Table 4: Different ways to incorporate popularity prediction task.

	AUC	MRR	nDCG@5	nDCG@10
CosSim	0.6830	0.3311	0.3662	0.4301
PENR+user γ	0.6794	0.3287	0.3628	0.4266
PENR+trunc	0.6801	0.3297	0.3636	0.4272
PENR w/o pop	0.6893	0.3390	0.3734	0.4385
PENR	0.6971	0.3410	0.3789	0.4421

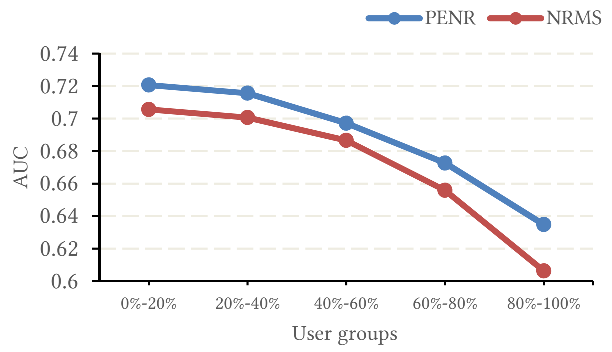
4.4 Popularity Prediction Method

To explore the best way to generate and incorporate popularity to CTR, we conduct the following experiments. The results on the validation set are summarized in Table 4. In CosSim, we use the method mentioned in [31], to get news popularity. The news popularity is calculated as the sum of the cosine similarity of candidate news and other news. In the second line PENR+user γ, we consider the user representation may affect the proportion of popularity in the final click-through rate, so we use the user vector as input, stacking a fully connected layer with a sigmoid function, and eventually predict the popularity γ value. Some users are very susceptible to hot news, while other users are not affected by hot news and only focus on their favorite areas. In contrast to PENR, we model the users' attention to hot news in another way. Another finding is that in the training dataset, the click volume of different news varies greatly. Some news gets only single-digit hits, while a small part of news is clicked tens of thousands of times. So in the third method PENR+trunc, we process the data by truncating the click volume which is larger than a pre-set threshold. Here we set the threshold to 20000. In addition, to verify the effectiveness of adding popularity to CTR result, we compare a method denoted as PENR w/o popularity prediction in the third line, which only treats popularity prediction task as a subtask of multi-task learning framework. At the time of prediction, the final CTR only consists of the output of bilinear click predictor, without the popularity score produced by the popularity prediction module. The last line PENR is the method mainly introduced in this paper. It divides all the visits count data by the largest value in data as normalization, multiplies popularity score by a learnable parameter γ, and adds the weighted popularity score to the final CTR. Results show that the last approach outperforms the methods mentioned above.

Unexpectedly, when we use the previous three methods, the AUC scores do not show substantial improvement. The second and the third methods even achieve worse performance. This phenomenon may occur because when we use user representation to predict popularity weight in the prediction process, the prediction result is of low quality. Thus, it incorrectly magnifies or minimizes the impact of popularity on click-through rate predictions. Besides, cosine similarity cannot effectively represent



(a) statistics



(b) AUC of different user groups

Figure 5: (a) Group users according to the percent of popular news in their browsing history. We count the percentage of top200 news and top500 news, respectively. (b) Calculate AUC for different user groups on validation dataset with baseline model NRMS and our PENR model.

news popularity. Only with the way we proposed in this paper can we get a significantly improved result, which proves the validity of the proposed popularity prediction module.

To verify the effect of our model on different users, we divide all users into five groups according to the proportion of popular news in click history. We take the top 200 clicked news and top 500 clicked news as popular news, respectively. The statistical results are shown in Figure 5(a), more than 60% of users read popular news at a rate between 20% and 60%. Then we consider the top 500 news as popular news and calculate AUC of the four user groups. Figure 5(b) reports the results. The performance of both models declines as the percentage of clicked popular news increases. This is not difficult to understand since when a user clicks more popular news, his/her click behaviors usually contain more randomness. In other words, he/she may click on news that could not be inferred from the click history. Our PENR model gains better results on all user groups, especially on users with more than 60% popular news in their click history. It indicates that popularity in our model helps to better deal with users who are susceptible to popular news, and reduces the impact of randomness on recommendation results.

4.5 Multi-view Interest Representation

To further investigate how the number of attention networks in the user encoder influences the model performance, we conduct experiments that integrate 1 to 10 attention networks to the original model, respectively. The results are shown in Figure 6. With the number of attention networks N_a increase, the AUC score improves at first. When N_a is set to 5, it achieves the best performance. While we continue to join more attention networks, the performance shows a significant drop. At first, the user interests are separated into different interest subspace, so that the model can match candidate news from multiple perspectives. However, when there are too many subspaces, much noise will be introduced to worsen the results. Consequently, considering the accuracy and the computational efficiency, we set N_a to 5.

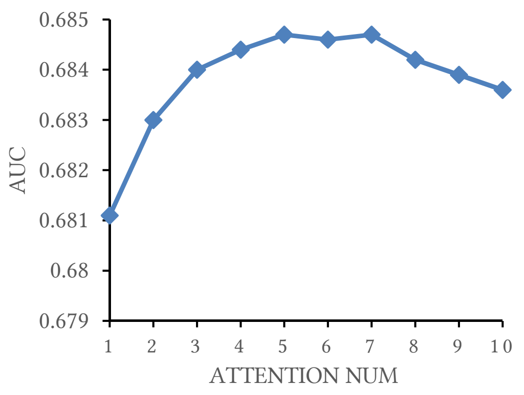


Figure 6: Performance with different number of attention networks in user encoder.

Table 5: Performance comparison of different user groups. According to the number of news in click history denoted as n, we divide users into four groups: n=0; n<5; 0<n<50; overall.

	AUC			
	n=0	n<5	0<n<50	overall
NPA	0.5824	0.6129	0.6672	0.6647
NRMS	0.5751	0.6105	0.6835	0.6803
FIM	0.5806	0.6153	0.6865	0.6833
PENR	0.5975	0.6304	0.6991	0.6971

4.6 Cold Start Problem

We find that a small part of users in the dataset have no click history or only click on a few news. It is hard for previous methods to generate appropriate user vectors, which lead to almost random prediction results. Cold start is a crucial issue in news recommendation. This can be improved by adding popularity to CTR according to our method. We select four groups of users based on the number of news in click history and calculate AUC of each group, as shown in Table 5. Experiments show that our model achieves promising results on users with few click history, especially on users who have only clicked less than five news. Therefore, our model solves the cold start problem to a certain extent.

Table 6: A case study on user with browsing history, the original score shows that the user is most likely to click N21018 which is not clicked by the user. When we add popularity to the original score, it finally gives the correct recommendation. The bold line represents the positive sample, others represent negative samples.

	NewsID	News Title	Original	Popularity	Final
History	N122047	Something fishy happened in that Texans-Chiefs call	-	-	-
	N26193	World Series storylines to keep an eye on	-	-	-
Candidate	**N46641-1**	**Former North Carolina State, NBA player Anthony Grundy dies in stabbing, police say**	**1.765**	**0.393**	**2.158**
	N2110-0	66 Cool Tech Gifts Anyone Would Be Thrilled to Receive	0.784	0.011	0.795
	N21018-0	3 Indiana judges suspended after a night of drinking turned into a White Castle brawl	1.814	0.049	1.863
	N46555-0	Federal Prosecutors Probe Giuliani's Links to Ukrainian Energy Projects	0.237	0.025	0.262
	N129416-0	Taylor Swift Rep Hits Back at Big Machine, Claims She's Actually Owed $7.9 Million in Unpaid Royalties	1.587	0.049	1.636

Table 7: Training time per step with batch size 128 and average inference time for each user.

	Training Time (s)	Inference Time (ms)
NPA	0.071 (1×)	8.807 (1×)
NRMS	0.106 (1.49×)	8.959 (1.02×)
FIM	0.372 (5.25×)	14.013 (1.59×)
PENR	0.120 (1.69×)	9.312 (1.06×)

4.7 Computational Efficiency

To evaluate the computational efficiency of our proposed PENR model, we compare three other baseline methods. We list the training and inference time for the four models in Table 7. All the experiments are conducted on a single NVIDIA RTX A6000 GPU. We record the average time spent training a step with batch size 128 and the average inference time for a single user. NPA is the fastest in training and testing of the four models, while NRMS gets similar results. Although FIM achieves higher AUC than NRMS and NPA in previous experiments, it takes about five times longer to train and 1.6 times longer to infer. It might be because each candidate news needs to interact with all the news in click history in FIM, while in other models, each candidate news only interacts with a single user vector generated from click history. The results indicate that our PENR model improves prediction accuracy but does not significantly reduce computational efficiency.

4.8 Case Study

We provide a case in Table 6, in which the user has two news in his/her browsing history. The original score is the output of the bilinear click predictor without news popularity. Without the news popularity score predicted by our model, it recommends news N21018 that will not be clicked in the end. However, after adding the news popularity score, the news most likely to be clicked on becomes N46641, which the user finally clicks. Therefore, we prove the effectiveness and importance of popularity for news recommendation.

5 CONCLUSION

News recommendation technology dramatically improves the efficiency of people to obtain external information. In this paper, we propose the PENR model, which introduces news popularity to news recommendation architecture, and enhances user interest representation in multiple perspectives. Our model generates different user interest vectors in several interest subspaces and match candidate news separately. In addition, we integrate the popularity prediction task in a novel way, in which the popularity score directly affects the final CTR. We conduct various experiments on a real-world dataset. The experimental results demonstrate that our model significantly outperforms the baseline models. Further analysis shows the effectiveness of the proposed component in our PENR model.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Program of China (No. 2020YFC0833301).

REFERENCES

[1] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with attentive multi-view learning. *arXiv preprint arXiv:1907.05576* (2019).

[2] Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Multi-Head Self-Attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 6390–6395.

[3] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1835–1844. https://doi.org/10.1145/3178876.3186175

[4] Qiannan Zhu, Xiaofei Zhou, Zeliang Song, Jianlong Tan, and Li Guo. 2019. Dan: Deep attention neural network for news recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 5973–5980.

[5] T.Y.S.S Santosh, Avirup Saha, and Niloy Ganguly. 2020. MVL: Multi-View Learning for News Recommendation. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Association for Computing Machinery, New York, NY, USA, 1873–1876. https://doi.org/10.1145/3397271.3401294

[6] Danyang Liu, Jianxun Lian, Shiyin Wang, Ying Qiao, Jiun-Hung Chen, Guangzhong Sun, and Xing Xie. 2020. KRED: Knowledge-Aware Document Representation for News Recommendations. In *Fourteenth ACM Conference on Recommender Systems (RecSys '20)*. Association for Computing Machinery, New York, NY, USA, 200–209. https://doi.org/10.1145/ 3383313.3412237

[7] Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long-and short-term user representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 336–345.

[8] Linmei Hu, Chen Li, Chuan Shi, Cheng Yang, and Chao Shao. 2020. Graph neural news recommendation with long-term and short-term interest modeling. *Information Processing Management*, 57(2):102142.

[9] Linmei Hu, Siyong Xu, Chen Li, Cheng Yang, Chuan Shi, Nan Duan, Xing Xie, Ming Zhou. 2020. Graph Neural News Recommendation with Unsupervised Preference Disentanglement. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 4255-4264.

[10] Suyu Ge, Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2020. Graph Enhanced Representation Learning for News Recommendation. *Proceedings of The Web Conference 2020*. Association for Computing Machinery, New York, NY, USA, 2863–2869. https://doi.org/10.1145/ 3366423.3380050

[11] Heyuan Wang, Fangzhao Wu, Zheng Liu, Xing Xie. 2020. Fine-grained Interest Matching for Neural News Recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 836-845.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin. 2017. Attention is all you need. s. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*. 5998-6008.

[13] Timothy Dozat, Christopher D. Manning. 2017. Deep Biaffine Attention for Neural Dependency Parsing. In *5th International Conference on Learning Representations*. https://arxiv.org/abs/1611.01734

[14] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, Ming Zhou. 2020. MIND: A Large-scale Dataset for News Recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 3597–3606.

[15] Jeffrey Pennington, Richard Socher, Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[16] Diederik P. Kingma, Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv*:1412.6980.

[17] Steffen Rendle. 2012. Factorization Machines with libFM. *ACM Transactions Intelligent Systems and Technology*. 3(3): 57:1-57:22. https://doi.org/10.1145/ 2168752.2168771

[18] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management (CIKM '13)*. Association for Computing Machinery, New York, NY, USA, 2333–2338. https: //doi.org/10.1145/ 2505515.2505665

[19] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems (DLRS 2016)*. Association for Computing Machinery, New York, NY, USA, 7–10. https://doi.org/10.1145/2988450.2988454

[20] Huifeng Guo, Ruiming TANG, Yunming Ye, Zhenguo Li, Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence Main track*. 1725-1731. https://doi.org/10.24963/ijcai.2017/239

[21] Jianxun Lian, Fuzheng Zhang, Xing Xie, Guangzhong Sun. 2018. Towards Better Representation Learning for Personalized News Recommendation: a Multi-Channel Deep Fusion Approach. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence Main track* 3805-3811. https://doi.org/10.24963/ijcai.2018/529

[22] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based News Recommendation for Millions of Users. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*. Association for Computing Machinery, New York, NY, USA, 1933–1942. https://doi.org/10.1145/ 3097983.3098108

[23] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. NPA: Neural News Recommendation with Personalized Attention. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Association for Computing Machinery, New York, NY, USA, 2576–2584. https://doi.org/10.1145/3292500.3330665

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. 8026-8037.

[25] Chuhan Wu, Fangzhao Wu, Mingxiao An, Yongfeng Huang, Xing Xie. 2019. Neural News Recommendation with Topic-Aware News Representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 1154-1159.

[26] ShuWu, Yuyuan Tang, Yanqiao Zhu, LiangWang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 346–353.

[27] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *KDD*. Association for Computing Machinery, 974–983.

[28] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. 1997. GroupLens: applying collaborative filtering to Usenet news. Commun. ACM 40, 3 (March 1997), 77–87. https://doi.org/10.1145/245108.245126

[29] Guang Ling, Michael R. Lyu, and Irwin King. 2014. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the 8th ACM Conference on Recommender systems (RecSys '14)*. Association for Computing Machinery, New York, NY, USA, 105–112. https://doi.org/10.1145/2645710.2645728

[30] Zhaochun Ren, Shangsong Liang, Piji Li, Shuaiqiang Wang, and Maarten de Rijke. 2017. Social Collaborative Viewpoint Regression with Explainable Recommendations. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 485–494. https://doi.org/10.1145/ 3018661.3018686

[31] Jonnalagedda N, Gauch S, Labille K, Alfarhood S. 2016. Incorporating popularity in a personalized news recommender system. *PeerJ Computer Science* 2:e63 https://doi.org/10.7717/peerj-cs.63