CSRN: Collaborative Sequential Recommendation Networks for News Retrieval

Bing Bai, Guanhua Zhang, Ye Lin, Hao Li, Kun Bai, Bo Luo

ABSTRACT

Nowadays, news apps have taken over the popularity of paper-based media, providing a great opportunity for personalization. Recurrent Neural Network (RNN)-based sequential recommendation is a popular approach that utilizes users’ recent browsing history to predict future items. This approach is limited that it does not consider the societal influences of news consumption, i.e., users may follow popular topics that are constantly changing, while certain hot topics might be spreading only among specific groups of people. Such societal impact is difficult to predict given only users’ own reading histories. On the other hand, the traditional User-based Collaborative Filtering (UserCF) makes recommendations based on the interests of the “neighbors”, which provides the possibility to supplement the weaknesses of RNN-based methods. However, conventional UserCF only uses a single similarity metric to model the relationships between users, which is too coarse-grained and thus limits the performance. In this paper, we propose a framework of deep neural networks to integrate the RNN-based sequential recommendations and the key ideas from UserCF, to develop Collaborative Sequential Recommendation Networks (CSRNs). Firstly, we build a directed co-reading network of users, to capture the fine-grained topic-specific similarities between users in a vector space. Then, the CSRN model encodes users with RNNs, and learns to attend to neighbors and summarize what news they are reading at the moment. Finally, news articles are recommended according to both the user’s own state and the summarized state of the neighbors. Experiments on two public datasets show that the proposed model outperforms the state-of-the-art approaches significantly.

CCS CONCEPTS

• Information systems → Recommender systems; Information retrieval; • Computing methodologies → Neural networks.

KEYWORDS

Recommender systems, news recommendation, sequential recommendations, neural networks.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference'17, July 2017, Washington, DC, USA
© 2020 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM ... $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

With the development of the mobile internet, people now tend to read news online rather than reading conventional paper-based media. Such transition provides a great opportunity for personalized news recommendations.

Unlike E-commerce and many other scenes, active users can consume news very fast, for example, browsing over dozens of news articles within ten minutes. As a result, how to capture the user’s rapidly changing interests and react to the user’s latest behavior becomes one of the key challenges of news recommendations [21]. To address these issues, sequential recommendations, which utilize the user behavior sequences and embed previously purchased products for current interest prediction [5, 31], have been applied for news recommendations successfully, especially on some recent models based on Recurrent Neural Networks (RNNs) [15, 21].

Despite the achievements, we argue that such methods still face challenges, mostly caused by the sociality in news reading. We can observe that users may read the news not because she was interested in the topic recently, but because she found it important [23]. People tend to follow these constantly changing “hot” topics, and sometimes the hot topics are only of significance to certain groups of people. With such sociality, predicting the relevance between users and news solely based on the target users’ own browsing activities can be tough. Let’s consider the following example:

Tom is a driver who likes baseball. One day, after he read a dozen baseball-related news articles, a piece of news entitled “Ice is causing trouble in the traffic” emerged and spread among people who were sensitive to the condition of traffic.1

In this example, RNN-based sequential recommendations can hardly succeed in recommending the merging news. Some works try to alleviate this problem by considering the users’ general preference [5, 31] or social relationships [28], however, we argue that using general preferences is too static, while the social relationships among users are hard to obtain. On the other side, traditional User-based Collaborative Filtering (UserCF) approaches recommend news articles based on what other similar people (i.e., neighbors) are reading, which provide a possibility to jump out of the user’s own browsing context and recommend the news that neighbors are interested in. However, the classic UserCF has some

1This example is inspired by the case studies in Section 6.6
limitations, including: (1) It’s difficult for UserCF to react immediately to the target user’s most recent behavior [17]; (2) UserCF only use one single scalar of similarity metric to represent the relationship between two users, which is too coarse-grained. As a result, some researchers suggest that UserCF is not suitable for news recommendations [21, 34], although the idea of using similar people’s behaviors is valuable.

To tackle this challenge, in this paper, we integrate the RNN-based sequential recommendation algorithms and the key idea of User-based collaborative filtering into a framework of deep neural networks, and propose a new model called Collaborative Sequential Recommendation Networks (CSRNs). The proposed model works in the following way.

Firstly, to better model the relationships between users, we design a directed news co-reading network, which is built with early browsing history. With Singular Value Decomposition (SVD), users can be assigned with vectors indicating their general preferences, and similar users can be detected and modeled in a co-reading network. Furthermore, the edges between users can be described with a function of the vectors of connected users. Therefore, compared with UserCF, the relationships between users can be described in a vector space, which is a more fine-grained way.

Secondly, recommendations are made according to the recent browsing history of both the target user and her neighbors. Similar to other sequential recommendation models, we use RNNs to encode users’ recent browsing history, thus the encoded features can represent what kind of news the users are interested in recently. Afterward, based on the current states of the target user and the neighbors, as well as their relationships, the model will pay different attention dynamically to different neighbors and make personalized summarization of what the neighbors are reading. Finally, recommendations are made according to the information from both the target user and the neighbors aspects.

In this way, the model can successfully handle the situation in the example by finding

Jerry and Tom share similar interests in traffic-related news. One day Jerry reads a piece of news in that category, and even if Tom didn’t show any evidence of relevance in his recent behavior, the news could still be recommended to Tom.

To the best of our knowledge, this is the first attempt to integrate the idea of UserCF into a neural network model for sequential recommendations. The key contributions of this paper are summarized as follows:

- We propose an approach to build news co-reading networks, which can describe the relationships meticulously. The co-reading network will be employed for better news recommendation.
- Based on the news co-reading network, we propose a novel neural network model named as Collaborative Sequential Recommendation Networks (CSRNs), which combines the advantages of RNN-based sequential recommendations and UserCF.
- We evaluate the proposed model comprehensively on two public datasets. Extensive experimental results show that the proposed model outperforms the state-of-the-art baselines significantly.

The rest of this paper is organized as follows. We briefly summarize related works in Section 2. Section 3 introduces the design of the news co-reading network. The details of the proposed CSRNs are then presented in Section 4. Section 5 conducts empirical studies on the co-reading network built with a real-world dataset. Section 6 reports the experimental results, and Section 7 concludes the paper.

## 2 RELATED WORK

In this section, we review the literature on two topics that are most relevant to our research, i.e., news recommendations and sequential recommendations. We also explain how our approach differs from the literature.

### 2.1 News Recommendations

News recommendations have been widely studied in the research community. Early work used memory-based and model-based collaborative filtering algorithms [4]. However, since news expires fast, CF-based methods often suffer from the cold-start problem. Therefore, many algorithms that utilized the content of news were proposed [1, 19]. For example, Lu et al. [19] proposed a content-based collaborative filtering approach to bring both content-based collaborative filtering approaches together. Recently, neural network-based algorithms have been widely studied for news recommendations [15, 21, 24]. Okura et al. [21] used Recurrent Neural Networks (RNNs) with users’ recent browsing history to model user preference and make news recommendations. Kumar et al. [15] proposed to use Bi-directional LSTM and self-attention to improve the accuracy of recommendations. Park et al. [24] used a Convolutional Neural Network model to capture user preferences and to personalize recommendation results. There are also many works that combine other features into news recommendations, including knowledge graph [33], location [27] and so on. Karimi et al. reviewed the state-of-the-art of designing and evaluating news recommender systems over the recent years in [13].

The major difference between prior works and ours is that we highlight the sociality in news reading by integrating the idea of UserCF with conventional RNN-based sequential recommendations. By building news co-reading network and utilizing the information from neighbors, better recommendations can be achieved.

### 2.2 Sequential Recommendations

Traditional collaborative filtering models often ignore the temporal information [30]. However, in the real world, the order of historical behaviors matters a lot. To model this phenomenon, many sequential recommendation algorithms have been proposed. Rendle et al. [26] applied Markov chains to model user behavior sequences, and later neural network-based methods have been proposed and significantly improved the performance [2, 5, 31]. For example, Donkers et al. [5] proposed a new type of Gated Recurrent Units incorporating the explicit notion of the user for whom recommendations are specifically generated. Tang et al. [31] proposed a convolutional sequence embedding recommendation model to address the union-level sequential patterns and skip behaviors. Although focused on different aspects, many RNN-based session-based recommendation algorithms actually share related model structures with sequential recommendation models [9, 10]. The difference between
session-based recommendation and sequential recommendation is that, users are often assumed to be anonymous under session-based recommendations, while for sequential recommendations, user IDs can be obtained to link the different sessions of a user together.

While existing sequential recommendation algorithms are trying to utilize the behaviors of target users better, to the best of our knowledge, the proposed CSRN is the first attempt to bring the information from the neighbors into sequential recommendations in a Neural Network model. Jannach et al. found that combining the kNN approach with GRU4Rec can give better results for session-based recommendations in [12], however, they only tried weighted averaging the results given by kNN and GRU4Rec, which is completely different from our work.

3 CO-READING NETWORK CONSTRUCTION

To identify users with similar interest and describe such relationships, we introduce the news co-reading network, which is built with users’ early browsing history. Let

\[ r_{ij} = \begin{cases} 1, & \text{user } i \text{ has interactions with news } j \smallskip \\ 0, & \text{otherwise} \end{cases} \]

indicate whether user \( i \) has read the news \( j \), and

\[ R = [r_{ij}]_{i=1}^{I} \times [j=1]^{J} \]

is the \( I \times J \) binary rating matrix of the early browsing history, \( I \) is the number of distinct users and \( J \) is the number of distinct news. We show how the news co-reading network is built with \( R \).

Firstly, we apply Singular Value Decomposition (SVD) upon the TF-IDF transformed matrix \( R \), and only keep the \( T \) largest singular values and the corresponding vectors, i.e.,

\[ \hat{R} \approx U_{I \times T} \Sigma_{T \times T} V_{T \times J}. \]

where \( \hat{R} = \text{TF-IDF(R)} \), and the TF-IDF transformation here is applied to down-weight the most popular news items.

\( U \) is an \( I \times T \) matrix, and the \( i \)th row of \( U \), i.e., \( u_i \), can be a representation of user \( i \). We consider user \( i \) and user \( k \) are neighbors if the similarity of embeddings satisfies some certain conditions. For example, we can keep the top \( N \) most similar users as neighbors just like UserCF. The similarity score is defined as:

\[ \text{similarity}_{ik} = u_i \Sigma u_k^T \]

Secondly, to represent the relationship between user \( i \) and one of her neighbors, i.e., user \( k \), inspired by [7], we use a directed edge \( e_{ik} \) with features

\[ e_{ik} = [u_i \quad u_k \quad u_i \odot u_k], \]

where \( \odot \) is the Hadamard product operator which computes element-wise multiplication between vectors. The three parts of \( e_{ik} \) describe the target user, the source user, and their undirected relationship respectively. Compared with traditional user-based collaborative filtering, \( e_{ik} \) describes the relationship in a more meticulous way.

Finally, we can define the news co-reading network with a set of triplets

\[ G = \{(i, k, e_{ik})\}. \]

With the network, we can make better recommendations by CSRN.

Note that the network of users can also be built with information other than early browsing histories. For example, we can build a co-purchasing network using APP purchasing records. The intuition is that, users have similar APPs installed might be interested in similar kind of news [18]. By building a network with information other than the news reading domain, we can easily transfer knowledge and potentially handle the user-side cold-start problem.

4 COLLABORATIVE SEQUENTIAL RECOMMENDATION NETWORKS

In this section, we explain the details of the proposed CSRN, which utilize the co-reading network of users to make news recommendations. The framework is illustrated in Figure 1. The model consists of mainly three parts, i.e., user encoding, attending, and recommending.

4.1 User Encoding

We use RNNs to encode users’ recent behaviors similar to [10, 21]. RNNs are networks that deal with sequential data, and they compute the user’s current hidden state based on the hidden state at last step and the input at the current step, i.e.,

\[ h^t = f(\varphi(h^{t-1}), w^t), \]

where \( h^t \) is the hidden state at step \( t \), and \( w^t \) is the input vector at step \( t \). There are many formulation of \( f \), including Long Short-Term Memory (LSTM) [11], Gated Recurrent Unit (GRU) [3], vanilla RNN, and many other variants.

For news recommendations, the inputs are the sequences of news embeddings that the users read recently, and the outputs are the hidden states of users. Details can be found in [15, 21]. Note that a variety of user encoding models are compatible with the framework of CSRNs, as long as that the model can map a user to a vector.

4.2 Attending

Based on the co-reading network \( G \), CSRN can learn to personally summarize what one’s neighbors are interested in at the moment, and utilize the information to make better recommendations. As shown in Figure 1, assuming that user \( i \) is the target user to recommend for, and user \( k \) is one of user \( i \)’s neighbors in the co-reading network \( G \), \( h_i \) and \( h_k \) are users’ hidden states encoded according to what news has been read recently, and \( e_{ik} \) is the feature of the directed edge connecting user \( i \) and user \( k \). The attending procedure works in the following way.

The first step is to decide what information can go through the edge. We use a single layer network defined by the following equation:

\[ p_{ik} = \phi(W_{ph} h_k + W_{pe} e_{ik} + b_p), \]

where \( \phi(\cdot) \) is the \( \text{Tanh} \) function, and \( p_{ik} \) contains the information that can pass through the edge from user \( k \) to user \( i \).

The next step is to decide what information user \( i \) would like to extract from the attended user \( k \) based on the current state. This is achieved by the gates [3] defined with the following equation:

\[ g_{ik} = \sigma(W_{gh} h_i + W_{ge} e_{ik} + b_g), \]

where \( \sigma(\cdot) \) is the \( \text{Sigmoid} \) function, and \( g_{ik} \) is the gate indicating what information user \( i \) would like to extract from user \( k \).
Then, the encoded information from user \( k \) to \( i \) is formulated as:

\[
c_{ik} = g_{ik} \odot p_{ik}. \tag{6}
\]

The final step is to summarize the information from all the neighbors. Inspired by [32], we use (multi-head) attention networks to compute the weights for different neighbors. The attention networks work as follows:

\[
\alpha_{ik} = \text{LeakyReLU}(w_{ah} h_i + w_{ae} e_{ik} + w_{ac} c_{ik} + b_a) \tag{7}
\]

where the negative input slope of LeakyReLU is set to 0.2. Then a Softmax function is used to normalize the weights:

\[
w_{ik} = \frac{\exp(\alpha_{ik})}{\sum_{l \in \mathcal{N}_i} \exp(\alpha_{il})}, \tag{8}
\]

where \( \mathcal{N}_i \) is the set of neighbors of user \( i \).

The final summarization of the information from neighbors are defined as:

\[
n_i = \sum_{k \in \mathcal{N}_i} w_{ik} c_{ik} \tag{9}
\]

To stabilize the learning process of attention, we apply a multi-head attention mechanism. \( H \) independent attention procedures executing the averaging transformation of Equation (9) are conducted, and the outputs are concatenated as the final summarization of the neighbors, i.e.,

\[
n_i = \|_{h=1}^H \left( \sum_{k \in \mathcal{N}_i} w_{ik}^h e_{ik}^h \right), \tag{10}
\]

where \( \| \) represents the concatenation operation.

From the steps above, we can see how the features of edges which represent the relationship between users, and the hidden states of users which indicate their recent interests, are taking effects for news recommendations. The \( n_i \) contains personalized summarization of what the neighbors are reading, and can be valuable for better recommendations.

### 4.3 Recommending

With the hidden states of the target user, i.e., \( h_i \) and the summarization of what the neighbors are reading, i.e., \( n_i \), the model can finally compute recommendation scores for each candidate news. In this paper, we deal with news retrieval tasks, thus the relevance function is restricted to a simple inner-product function [21].

The decoding function for users are formulated as:

\[
q_i = \phi(W_q h_i + W_q n_i + b_q) \tag{11}
\]

and the recommendation score is defined as:

\[
r_{ij} = q_i^T v_j, \tag{12}
\]

where \( v_j \) is the embeddings of news \( j \).

### 4.4 Loss Functions and Regularizations

We experiment three kinds of loss functions, including two pairwise losses TOP1-Max and BPR-Max proposed in [9], as well as the classic Cross-Entropy. The definitions of the loss functions are
CSRN: Collaborative Sequential Recommendation Networks
for News Retrieval

(a) Distribution of the out-degrees (b) Percentage of node pairs that it (c) Visualization of the adjacency (d) Statistics about the proportions of
of nodes and the corresponding user can be reached within certain num-
activities. For demonstration pur-
ers of steps from a start node with
poses, only the nodes with out-degrees greater than 0 are included in this
figure.

Figure 2: Empirical studies on the co-reading network. Best viewed in color.

out-degree is 11,680. We also find that the user with the largest
out-degree clicked 538 news in the training set, which is not a sig-
ificantly larger number compared with other active users. Although
the out-degree is correlated with the user activity, the relationship
is not absolute. Our conclusion is that the TF-IDF transformation
in Equation (1) is taking effect, since if a user shows no signifi-
cant preference on certain kinds of news and only reads the most
popular ones, the value from her browsing history will be little.

Figure 2b focuses on the connectivity of the co-reading network.
Starting from nodes with out-degrees greater than 0, we calculate
the percentage of node pairs that it can be reached within certain
steps out of all possible node pairs. As illustrated, only less than 20%
node pairs can be reached within 10 steps, and the percentage starts
to saturate. This shows that the connectivity of the network is not
strong, indicating that the interests of the users are significantly
different, as the user aggregates into clusters that are separated from
each other in the network. We further visualize the top 1,000 most
active users with t-SNE [20] using the adjacency matrix as inputs in
Figure 2c. The shapes of symbols represent the most clicked news
category by the user. As we can see, there are three major clusters
among them, and although all labeled as “nyheter (news)”, users
can still be divided into two groups. This phenomenon suggests
that there is a large space for personalization, since the interests of
users differ from each other significantly.

In order to explore the potential of utilizing neighbors to make
news recommendations, we calculate the proportions of news click-
ing events in the training set versus how many neighbors had al-
ready read the news before the clicking event took place. Statistics
are shown in Figure 2d. We can find that 78.7% of all the click events
happened after at least one neighbor had read the clicked news
already, and about 31.7% of the click events took place when the
clicked news had been read by equal to or more than 5 neighbors.

As a conclusion, taking the Adressa dataset as an example, we
find that the co-reading network is highly centralized and poorly
connected. This is a preferred property since it indicates that we
only need to focus on a small set of representative users as neighbors
for others. These users are usually very active, and have their own
preferences for the type of news. Their browsing actions can be
valuable for revealing the hot topics at early stages, and help to

5 EMPIRICAL STUDIES ON THE CO-READING NETWORK

To gain a better insight into the co-reading network, in this section,
we conduct empirical studies on the co-reading network built with
the Adressa dataset. We keep the default of top 20 most similar
users as the neighbors for each user, unless other specific configura-
tions are explicitly specified. As a result, every node in the network
has an in-degree of 20, while the out-degrees can vary a lot.

Figure 2a demonstrates the distribution of out-degrees of nodes
(i.e., users). The out-degree shows a very skewed pattern, even more
skewed compared with the distribution of user activities, which is
defined by how many news articles each user read in the training
set. Based on our experiments, only 3,776 out of 50,000 users in
the dataset have out-degrees greater than 0, and the maximum

\[*\] Details about the dataset is introduced in Section 6.1.
We compare our proposed model with the following baselines. We use two publicly available datasets for evaluation, including introduc
ted in Appendix A.1. Statistics about the datasets are listed co-read
networks. Detailed steps of dataset preprocessing are users for both datasets, and split the datasets into history / training
news interaction relationships as well as the content of news. Other
the Adressa dataset [8] and [14]. We use the user-news interaction relationships as well as the content of news. Other attributes of users are beyond the scope of this paper.
We extract the sequences in which the news articles were read by users for both datasets, and split the datasets into history / training / validation / testing sets. The history sets are used for constructing co-reading networks. Detailed steps of dataset preprocessing are introduced in Appendix A.1. Statistics about the datasets are listed in Table 1.

### 6 EXPERIMENTAL RESULTS

In this section, we discuss the experimental results. The datasets used for evaluation and the baselines are introduced first, followed by the evaluation scheme and how the hyper-parameters are de
cided. Evaluation results and case studies are presented at last.

#### 6.1 Datasets

We use two publicly available datasets for evaluation, including the Adressa dataset [8] and the plista dataset [14]. We use the user-news interaction relationships as well as the content of news. Other attributes of users are beyond the scope of this paper.

We extract the sequences in which the news articles were read by users for both datasets, and split the datasets into history / training / validation / testing sets. The history sets are used for constructing co-reading networks. Detailed steps of dataset preprocessing are introduced in Appendix A.1. Statistics about the datasets are listed in Table 1.

#### 6.2 Baselines

We compare our proposed model with the following baselines.

- **POP** [10]: News articles are ranked by their popularities, i.e., how many times the news has been clicked by others.
- **ItemCF** [30]: This is one of the classical neighborhood-based collaborative filtering methods. News articles similar to what the user has read will be recommended to the target user.
- **UserCF** [30]: This is another form of the classical neighborhood-based collaborative filtering. News consumed by similar users will be recommended.
- **BPR** [25]: This is a commonly used matrix factorization method that optimizes a pair-wise ranking loss.
- **GRU** [21]: GRU are used for news sequential recommendation in [21]. To overcome the unavoidable cold-start problem of news, embeddings are first learnt from the content of news articles. We follow the improvement in the loss proposed in [9] in order to get a stronger baseline.
- **Caser** [31]: Caser considers union-level sequential patterns and skip behaviors by modeling user past historical interactions with both hierarchical and vertical convolutional neural networks. It also considers the users’ general preferences.
- **AUGRU** [5]: Attentional User-based GRU (AUGRU) considers individual users’ general preference in addition to sequences of consumed items. Attention mechanism is applied to adaptively shift focus between user and item aspects.

Compared with the proposed CSRN, UserCF utilizes a related basic idea while fails to describe the relationship between users in a fine-grained way. GRU only consider the target user’s own browsing history, thus it can serve as an ablation experiment. Caser and AUGRU are state-of-the-art methods for sequential recommendations that consider both user activities and general preferences.

#### 6.3 Evaluation Scheme and Hyper-parameter Settings

We adopt the leave-one-out evaluation method for news retrieval tasks similar to [15, 16]. The algorithms are requested to rank the ground truth news with 99 negative samples, and the negative samples are fixed and shared by all methods for fairness. The performance is judged by Hit Rate (HR) and Mean Reciprocal Rank (MRR). More detailed evaluation scheme is introduced in Appendix A.2.

The news embeddings are learnt with CDAE proposed in [22] and shared by all the methods that require representations for the content of news. The embedding size is set to 256.

We use grid searching to find the best hyper-parameters for the baselines and the proposed CSRN according to MRR on the validation set, and report the corresponding results on the testing set. For ItemCF, best performances are achieved when the number of neighbors is set to 350 on the Adressa dataset, and 200 on the plista dataset. For UserCF, the number of neighbors is set to 150 on the Adressa dataset, and 250 on the plista dataset. All the hidden sizes for BPR, GRU, Caser, AUGRU and the proposed CSRN are set to 128, and the RNN parts of GRU, AUGRU and CSRN are all single layer GRU units. We use 4 attention heads, and the dimension of $c_{ik}$ for each attention head is set to 32 to keep the dimensions of $h_i$ and $n_i$ equal.

The detailed hyper-parameter settings and the training scheme for NN-based methods are reported in Appendix A.3.

#### 6.4 Evaluation Results

The evaluation results are reported in Table 2. We report HR@1, HR@10, HR@20, and MRR results of the algorithms on both datasets.

As we can see, methods that tend to bias towards popular items significantly, i.e., POP and UserCF, don’t perform well on both datasets, partially because of that the negative sampling strategy is related to the popularities of items. The models need to distinguish between general popularity and personal relevance, thus this is

| Item | Adressa dataset | plista dataset |
|------|----------------|---------------|
| # users | 50,000 | 30,000 |
| # news | 17,453 | 2,146 |
| # clicks in training set | 2,285,316 | 2,726,891 |
| Duration of training set | 31 days | 12 days |
| # clicks in validation set | 344,407 | 368,214 |
| # clicks in testing set | 344,408 | 368,215 |
| Duration of testing set | 9 days | 6 days |
| Avg. # words per news | 122.80 | 28.18 |
| Density | 0.34% | 5.38% |

Table 1: Information about the datasets
a relatively hard setting especially for these methods. In our experiments, ItemCF outperforms UserCF, mainly because we use the cosine similarities of news embeddings in ItemCF, so it can recommend cold-start news. Matrix factorization-based methods, i.e., BPR, get the best results among traditional recommendation algorithms.

On both datasets, GRU, Caser and AUGRU outperform traditional baselines significantly. Comparing the results of these algorithms, we find that Caser and AUGRU enjoy larger promotions on the Adressa dataset than on the plista dataset, our assumption is that it is related to the density of the datasets. The user behavior on the plista dataset is far more dense than it on the Adressa dataset, the recent browsing histories are less likely to get outdated, thus introducing the users’ general preferences cannot help very much.

Among all the tested methods, CSRN gets the best results under most of the metrics. Interestingly, CSRN enjoys a larger improvement under HR@1 and MRR, which are both sensitive to the accuracy of the very top part, on the Adressa dataset than on the plista dataset. Our conclusion is that this phenomenon is related to the density of user behaviors too. The user’s recent behavior on the Adressa dataset is more likely to expire, thus the information from the neighbors can help a lot for identifying the right news and put it to the very top.

Another interesting finding is that the TOP1-Max and BPR-Max loss functions show significantly superior performance than CrossEntropy loss on the plista dataset. We find that it is related to the score regularization. If the score regularization is disabled, we can find a phenomenon similar to the CrossEntropy loss, i.e., serious overfitting occurs and it cannot be settled by dropout and weight decay. This finding shows that the score regularization is beneficial to the learning process, as it can prevent the model from simply memorizing the negative samples, especially on the plista dataset which involves a limited number of distinct news and a relatively short duration of time.

### 6.5 Impacts of Key Hyper-parameters and Ablation Experiments

We study the impacts of some key hyper-parameters and some key components on the Adressa dataset and report the results in Table 3. We mainly focus on HR@10 which represents the performance of the recall ability, and MRR which focuses more on the order of the very top. We use the BPR-Max loss which performs well on both datasets.

Group A studies the impact of RNN cells. We find that while vanilla RNN can give comparable performance, LSTM tends to get overfitted a little. We briefly experimented LSTM with stronger weight decay. This finding shows that the score regularization is beneficial to the learning process, as it can prevent the model from simply memorizing the negative samples, especially on the plista dataset which involves a limited number of distinct news and a relatively short duration of time.

### Table 3: Impact of key hyper-parameters and ablation experiments on the Adressa dataset

| Hyper-paramter | Value | HR@10 | MRR  |
|----------------|-------|-------|------|
| (A) RNN Cell   | LSTM  | 38.12% (+) | 0.1591 (+) |
|               | Vanilla RNN | 38.36% (+) | 0.1602 (+) |
| (B) Hidden Size | 512    | 38.06% (+) | 0.1620 (+) |
| Hidden Layers  | 2      | 38.17% (+) | 0.1582 (+) |
| (C) # Attention Heads | 1 | 38.10% (+) | 0.1600 (+) |
|               | 2      | 38.23% (+) | 0.1600 (+) |
|               | 8      | 38.46% (+) | 0.1602 (+) |
| (D) # Neighbors | 10     | 38.25% (+) | 0.1588 (+) |
|               | 30     | 38.39% (+) | 0.1624 (+) |
| (E) Edge Features | disabled | 37.48% (+) | 0.1558 (+) |
| Neighbor Selection | without TF-IDF | 38.03% (+) | 0.1586 (+) |
| Neighbor Selection | by random | 36.71% (+) | 0.1477 (+) |
| Neighbor Information | disabled | 33.20% (+) | 0.1385 (+) |
Table 4: Example cases and the ranking positions given by different algorithms

| Case No. | Title of the Ground Truth News                                           | GRU | Caser | AUGRU | CSRN | # Viewed |
|---------|------------------------------------------------------------------------|-----|-------|-------|------|----------|
| (1)     | Isen skaper trebbel i trafikken (Ice creates trouble in traffic)         | 57  | 41    | 34    | 3    | 13       |
| (2)     | Ingen holdepunkt for å si at det er skutt mot bussen (No clue to say it’s a shot on the bus) | 33  | 40    | 28    | 4    | 5        |
| (3)     | Norges tennisnasjon invitert til storturnering med stjernene (Norway’s tennis competition invited the stars to the tournament) | 21  | 9     | 5     | 3    | 13       |
| (4)     | Demidov til MLS (Vadim Demidov transferred to MLS club)                 | 37  | 35    | 44    | 4    | 5        |
| (5)     | Offentlig rangering forsterker ulikhetene mellom skolene (Public ranking reinforces inequalities between schools) | 28  | 50    | 36    | 2    | 1        |
| (6)     | Økte bomsatser er den mest effektive måten å redusere trafikken på      | 18  | 10    | 19    | 5    | 0        |

 regularization and find that using stronger dropout and weight decay cannot help significantly.

Group B studies the impact of model capacity. We find that simply adding the number of hidden units can lead to significantly better MRR results, while HR@10 decreases a bit. Adding more layers to the user encoding part cannot give better results under both metrics, confirming the conclusions from [10].

Group C studies the impact of multi-head attention. To prevent from adding too many parameters, we fix the size of $n_i$, i.e., the representations after concatenation. Experiments show that multi-head attention can bring some improvements, especially for HB@10. However, too many heads may bring an adverse impact under MRR.

Group D studies the impact of the number of neighbors. Based on the results, we can see that adding more neighbors can significantly improve both metrics. However, it imposes more computation overhead. It’s a trade-off between the computational cost and the performance.

Group E studies the impact of key components of CSRN. Firstly, we disable the edge features $e_{ik}$ in Equation (4) and (5). Then, we try selecting neighbors without the TF-IDF transformation in Equation (1), and completely by random. Finally, we disable the information from the neighbors, i.e., $n_i$ in Equation (11), which gets the baseline GRU. Experiments show that all the components play a critical role in achieving the best performance.

6.6 Case Studies

To gain a better insight into the proposed CSRN, we take several cases from the Adressa dataset and study their effectiveness. Titles of the ground truth news, the ranks given by GRU, Caser, AUGRU as well as CSRN, and how many neighbors had already viewed the news are reported in Table 4. We use BPR-Max loss for case studies.

The news in Case 1 is about the impact of the bad weather. Intuitively, people might read dozens of news articles about sports, but they don’t read that much news about the weather in a short term, so it’s harder to infer the relevance of weather-related news from the user’s browsing history, which is illustrated by the ranks given by other methods. However, this news is highly relevant to some specific groups of people. CSRN can find the relevance from the neighbors who share similar interest with the target user, thus put the news to the top successfully. Case 2 is about an attack at Saupstadringen, Norway, and we can find results similar to Case 1. By digging into what news the neighbors are reading, emerging news can be recommended accurately even if no evidence according to the user’s own browsing history shows the intrinsic relevance.

Case 3 and Case 4 are two examples of sport-related news, and the news of Case 3 is about a tennis competition, while the news of Case 4 is about a transfer event. Both news could be of great interest to certain groups of readers. CSRN can successfully put the right news on the top of the recommendation list, thanks to the information from neighbors. These cases show that the proposed CSRN models can find the pattern that, if two users both like a certain category of news, and a news article of that category is viewed by one of them, then the news should be recommended to the other. In both cases, this strategy is more effective than simply using the users’ static general preference.

Case 5 and Case 6 illustrate how the models would perform if only a few or none of the neighbors have read the ground truth article. As we can see from the table, even if no neighbors have viewed the news in Case 6, CSRN can still make better recommendations than the baselines sometimes. We dig into the data and find that in Case 6, neighbors were reading news articles entitled with “48 000 flere trailer på veien hver dag (48,000 more trailers are on the road every day)”, “Det er lov å sykle to i bredden (It is allowed to ride two in the width for cycling)”, and “Mange busskur har blitt knust i Trondheim (Many buscars have been broken in Trondheim)”. CSRN can find that the neighbors were interested in traffic-related news, this information could spread along the co-reading network, and finally contributed to a better recommendation.

As illustrated by the cases above, what other similar users are reading can help a lot for news recommendations, and the proposed CSRN can find the pattern from data and give better results.

7 CONCLUSION

In this paper, we present the Collaborative Sequential Recommendation Networks, which integrate the RNN-based sequential recommendations with the idea of User-based collaborative filtering into the framework of deep neural networks. Firstly, we propose the methodology of building co-reading networks with users’ early browsing history, then we propose the CSRN model which can learn attending functions of neighbors and make personal summaries of what other users are reading. Using both the target user’s recent browsing history and the summarization of what the
neighbors are reading, better recommendations can be achieved. Comprehensive experiments on two publicly available datasets show that the proposed CSRN outperforms baselines significantly.

There are several potential extensions to CSRN that could be addressed in our future work. Firstly, explicit temporal information could help the model to decide when to trust the user’s own browsing history more and when to trust information from the neighbors more. Secondly, we’d like to explore the performance of CSRN on other domains, like movie or music recommendations. Finally, extending the model for cold-start users could be another interesting research direction.

REFERENCES

[1] Trapit Bansal, Mrinal Das, and Chiranjib Bhattacharyya. 2015. Content driven user profiling for comment-worthy recommendations of news and blog articles. In Proceedings of the 9th ACM Conference on Recommender Systems. ACM, 195–202.

[2] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Xinyu Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. ACM, 108–116.

[3] Konna Yet Vedi, Bani Van Merrienboer, Damryt Bahadan, and Youshua Bengio. 2014. On the Properties of Neural Machine Translation. Encoder-Decoder Approaches. Computer Science (2014).

[4] Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. 2007. Google news personalization: scalable online collaborative filtering. In Proceedings of the 16th international conference on World Wide Web. ACM, 271–280.

[5] Tim Donkers, Benedikt Loepf, and Jürgen Ziegler. 2017. Sequential user-based recurrent neural network recommendations. In Proceedings of the Eleventh ACM Conference on Recommender Systems. ACM, 152–160.

[6] Hui Fang, Guibing Guo, Danning Zhang, and Yiheng Shu. 2019. Deep Learning-Based Sequential Recommender Systems: Concepts, Algorithms, and Evaluations. In International Conference on Web Engineering. Springer, 574–577.

[7] Liyu Gong and Qiang Cheng. 2018. Adaptive Edge Features Guided Graph Attention Networks. arXiv preprint arXiv:1809.02709 (2018).

[8] Jon Atle Gulla, Lemei Zhang, Peng Liu, Özlem Özgöbek, and Xiaomeng Su. 2017. The Adressa dataset for news recommendation. In Proceedings of the International Conference on Web Intelligence. ACM, 2717–2726.

[9] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks for session-based recommendations. In Proceedings of the ACM Conference on Recommendation Systems. ACM, 293–300.

[10] Mozhgan Karimi, Dietmar Jannach, and Michael Jugovac. 2018. News recommendations on the CSRN: Collaborative Sequential Recommendation Networks

A EXPERIMENTAL SETUP

A.1 Dataset Preprocessing

For the Adressa dataset, we extract the sequences in which the news articles were read by users, and keep the data of the top 50,000 most active registered users. If a user read news for less than 5 seconds, the click is considered as a mis-action and discarded. Records before 2017-02-19 are treated as the history to construct the news co-reading network, records between 2017-02-20 and 2017-03-22 are divided into a validation set and a testing set. For the plista dataset, we also extract the sequences in which the news articles were read by users, and keep the data of the top 30,000 most active users. Clicks on news without content are discarded since we cannot learn the embeddings for them. Records before 2016-02-20 are treated as the training set, and records after 2016-02-22 are used as the testing set, and records after 2017-03-23 are then divided into a validation set and a testing set. Since the news provided by the plista dataset is very limited, we crawl about 140k more news articles from www.tagesspiegel.de for news embedding learning. The categories of the crawled news are inferred from the URL.
A.2 Detailed Evaluation Scheme
We adopt the leave-one-out evaluation method similar to [15, 16]. Given a user's most recent browsing history, only the next clicked news serves as the positive sample. Negative samples for training are dynamically drawn from a larger pool, and negative samples for validation and testing are drawn once and shared by all models to give a fair comparison.

The negative sampling pool is based on whether the news articles were clicked within a time interval, and only news that wasn’t interacted by the users could be drawn as a negative sample. We find that the frequency of a news article being sampled as negative a sample is highly correlated to the popularity of that article. Since it’s too time-consuming to rank all items, similar to [16], for validation and testing we draw 99 negative samples, which means the models need to rank among 100 news and find which one might be clicked by users. The performance is judged by Hit Rate (HR) and Mean Reciprocal Rank (MRR) defined as follows,

\[
\text{HR@K} = \frac{1}{|C|} \sum_{c \in C} \mathbb{1}(R_c \leq K)
\]

\[
\text{MRR} = \frac{1}{|C|} \sum_{c \in C} \frac{1}{R_c}
\]

where \(C\) is the set of clicks, \(R_c\) is the ranking position of ground truth article for click event \(c \in C\), and \(\mathbb{1}(\cdot)\) is the indicative function.

While MRR is more sensitive to the accuracy of the very top part of the ranking list, HR@K treats the top K positions equally.

A.3 Detailed Settings for Hyper-parameters
For co-reading news construction, we set \(T\) to 32, i.e., we keep the 32 largest singular values and the corresponding vectors. We keep the default of top 20 most similar users as the neighbors for each user, unless other specific configurations are explicitly specified.

The news embeddings are learnt with CDAE and shared by all the baselines and CSRN which require representations of news articles. The embedding size is set to 256. The best hyper-parameters for CDAE is decided based on the MRR results given by GRU. For the Adressa dataset, we keep the 10,000 most frequent word tokens which appeared in less than 25% articles as inputs, masking noise level is set to 0.3, and weight decay is set to 8e-5. Keywords and name entities provided by the dataset are concatenated with the documents. For the plista dataset, we keep the 25,000 most frequent word tokens which appeared in less than 20% articles as inputs, masking noise level is set to 0.25, and weight decay is set to 1e-4. We find that the performance of algorithms is more sensitive to the hyper-parameters of CDAE on the plista dataset than it on the Adressa dataset, and the keywords and name entities in the Adressa dataset can help a lot for better embeddings.

Best hyper-parameters for baselines and CSRN are decided according to MRR on the validation set, and we report the corresponding results on the testing set.

For neighborhood-based CF methods, the number of neighbors starts from 50 and increase 50 each time until the performance starts to decrease. For ItemCF, best performances are achieved when the number of neighbors is set to 350 on the Adressa dataset, and 300 on the plista dataset. For UserCF, the number of neighbors is set to 150 on the Adressa dataset, and 250 on the plista dataset. For neighborhood-based CF methods, the number of neighbors starts from 50 and increase 50 each time until the performance starts to decrease. For ItemCF, best performances are achieved when the number of neighbors is set to 350 on the Adressa dataset, and 300 on the plista dataset. For UserCF, the number of neighbors is set to 150 on the Adressa dataset, and 250 on the plista dataset.

All the hidden sizes for BPR, GRU, Caser, AUGRU and the proposed CSRN are set to 128. Weight decay is chosen from \([1e-3, 1e-4, 1e-5, 1e-6]\), and we found that best weight decay for CSRN is 1e-5, while for other models it’s 1e-4. Then we use grid searching to find the best hyper-parameters according to MRR on the validation set. The search range of dropout rate is \([0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4]\). For the Adressa dataset, dropout rate for GRU is set to 0.1 for the input embeddings and before the final decoder. For the plista dataset, dropout rate for GRU is set to 0.2 for the input embeddings and before the final decoder. For Caser, the max height \(h\) is chosen from \([1, 2, 4, 8]\), the number of horizontal filters for each height and the number of vertical filters are all chosen from \([4, 8, 16, 32]\), and we found that best performance is achieved when they set to 4, 8, 16 respectively for the Adressa dataset, and 2, 8, 8 respectively for the plista dataset. For AUGRU, we find that the best dropout rates are 0.15 for input embeddings and 0.1 before the final decoder function on the Adressa dataset, and 0.25 for both on the plista dataset. For CSRN, we find that the best performance is achieved when the dropout rate is set to 0.15 for the input embeddings and 0.2 for the decoder with the Adressa dataset, and 0.2 for the input embeddings and before the decoder with plista dataset respectively. We use 4 attention heads, and the dimension of \(c_{ik}\) for each attention head is set to 32 to keep the dimensions of \(h_i\) and \(n_i\) equal.

For all the NN-based methods, we use the PyTorch\(^3\) for implementation with 2 NVIDIA Tesla M40 GPUs. RMSprop is used as the optimizer, batch size is set to 256. Learning rates start from 0.0001, and then decay at a fixed rate of every 1000 steps. Gradient clipping is used to avoid gradient explosion and set to 5. We use single layer GRUs for RNN-based methods if no specific configuration is mentioned.

\(^3\)https://pytorch.org