2022 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

Session-based News Recommendation from Temporal User Commenting Dynamics

Chen Shen[1] Chao Han[2] Lihong He[3] Arjun Mukherjee[4] Zoran Obradovic[2] Eduard Dragut[2]
[1]*Megagon Labs, USA*
[2]*Computer and Information Sciences, Temple University, USA*
[3]*IBM Research, Almaden Lab, USA*
[4]*Computer Science Department, University of Houston, USA*
chen_s@megagon.ai, chao.han@temple.edu, lihong.he@ibm.com, arjun@cs.uh.edu, {zoran.obradovic, edragut}@temple.edu

***Abstract*—With the increase in volume of daily online news items, it is more and more difficult for readers to identify news articles relevant to their interests. Thus, effective recommendation systems are critical for an effective user news consumption experience. Existing news recommendation methods usually rely on the news click history to model user interest. However, there are other signals about user behaviors, such as user commenting activity, which have not been used before. We propose a recommendation algorithm that predicts articles a user may be interested in, given her historical sequential commenting behavior on news articles. We show that following this sequential user behavior the news recommendation problem falls into in the class of session-based recommendation. The techniques in this class seek to model users' sequential and temporal behaviors. While we seek to follow the general directions in this space, we face unique challenges specific to news in modeling temporal dynamics, e.g., users' interests shift over time, users comment irregularly on articles, and articles are perishable items with limited lifespans. We propose a recency-regularized neural attentive framework for session-based news recommendation. The proposed method is able to capture the temporal dynamics of both users and news articles, while maintaining interpretability. We design a lag-aware attention and a recency regularization to model the time effect of news articles and comments. We conduct extensive empirical studies on 3 real-world news datasets to demonstrate the effectiveness of our method.**

***Index Terms*—Recommender systems, Neural networks, Session based recommendation**

I. Introduction

Many news media outlets allow users to comment on the news stories published on their websites. User comments have evolved into a standard feature of online news and are considered one of the popular form of public online participation [1]. In this work, we aim to predict the article of interest for users. Comments are used as the proxy for an article of interest. We tackle the problem from a session-based news recommendation perspective.

Session-based recommendation is a scenario where implicit feedback (e.g., browsing, comments) is collected within a session from anonymous users [2]. The sequence of their implicit feedback determines the recommendation actions. Unlike session-based recommendation in e-commerce or movie settings, news articles and their readers exhibit unique temporal dynamics: the expected lifetime of news articles is short and their impact is typically bounded by their *immediacy*– their closeness to an emerging event– and readers' interest changes over time. Hence, news recommendation faces challenges specific to both session-based recommendation and temporal dynamics modeling.

The classic way to predict user interest in news recommendation is based on clicks, which may inaccurately reflect a user's interest. Many clicks are accidental or for a quick glance. A user may be attracted by the title and lose interest in the news soon after the click. Commenting, on the other hand, are of good quality. Many news websites have semi-automatic moderation and filtering systems in place. In addition, the text of comments provide richer information for filtering and preprocessing than clicks for recommendation approaches. Thus, *we hypothesize that a user's commenting action is a strong signal about news consumption interest.*

We also need to consider that a user's interest changes over time, commenting activity is unevenly distributed over time, and the user may not comment on all articles; and that the news value of an article to a user decays over time. Thus, we have to abstract out user-news article-comment temporal dynamics into new temporal variables, such as the time interval from a historical commenting action until the time when the recommendation is provided, and the age of published article at the time we compute a recommendation. We call the former *lag*, and the latter *recency*.

In this work, we propose a method to capture the unique temporal user commenting dynamics with the following capabilities: (1) it uses RNN-based models to capture a user's sequential behavior; (2) it includes an attention mechanism to model a user's uneven commenting actions and another to model the importance of a reader's historical actions; and (3) it includes a recency-based regularizor to penalize the prediction scores of fresh articles (small recency) less, and older articles more. We make the following contributions in this work:

- We propose user commenting activity and its associated temporal dimensions as a new basis for news recommendation.
- We propose an interpretable attentive neural network framework that captures temporal dynamics observed in

IEEE/ACM ASONAM 2022, November 10-13, 2022
978-1-6654-5661-6/22/$31.00 © 2022 IEEE

news recommendation.
- We perform an extensive empirical study with a large dataset from three news outlets. The performance gain of our model over the baselines is at least 40%.

II. Related Work

In this section, we present related studies in session-based recommendation methods and news recommendation methods.

A. Sequential Recommendation Methods

Non Deep Learning Approaches. Methods based on Markov Chains are typical for sequential recommendation [3]–[7]. Markov Chain based methods however are limited by the Markov assumption, so it is difficult to capture a user's evolving interest. One other family of methods proposes to model temporal related factors in recommendation [8], [9]. Session-based KNN approaches [10]–[12] are proposed as strong baselines compared to deep learning approaches. However, this general method does not focus on sequential information and user's interest, which play essential roles in news recommendation.

Deep Learning Approaches. Deep learning models have been applied successfully in sequential recommendation, a.k.a., *session-based recommendation*, given their advantages in modelling long-term and short-term aspects in user's sequential behavior online [11]. A Gated Recurrent Unit model is the first successful attempt of applying RNN to session-based recommendation [13]. It was latter improved by adding data augmentation and dropout on the input [14]. Li et al. [15] propose a hybrid attentive RNN model with a global recommender to capture user's general interest, and a local recommender to model user's current interest. Similarly, STAMP [16] is an attentive multilayer perceptron neural model to capture the long-term memory from the aggregation of user's sequential behaviors. Recurrent approaches have also been applied in combination with collaborative filtering approaches by modeling the evolution of user rating and item rating via RNNs [17], [18]. Wang et al. [19] applies collaborative neighborhood information to session-based recommendations with hybrid inner and outer memory encoder modules. Although those methods can model user's sequential behavior, they do not exploit any temporal variables, like the time interval between user's actions. Zhu et al. [20] propose a variant of LSTM by considering the effect of time intervals. Beutel et al. [21] propose a method to utilize contextual data like time intervals, page, and software client for video recommendation. Several other RNN based methods are developed to model sequential data by utilizing contextual data in similar applications, for example, multivariate time series forecasting [22], and predicting user's retention time [23]. Since their applications are different, these works are out of the scope of the study in this paper. Nowadays, graph neural network has attracted a lot of attention. Wu et al. [24] models session sequences as graph structured data and captures complex item transitions. Xu et al. [25] enhances self-attention network of session-based recommendation by combining graph neural network and self-attention model. Song et al. [26] extends graph-attention network by modeling dynamic interests and user influences. Repeat recommendation in session based recommendation is explored in [27] by proposing a repeat-explore mechanism with encoder-decoder structure.

These methods do not cope well with the dynamic changes (fast outdated and evolving user interest) in the news environment [28]. The general recommendation systems do not handle well item value decay (e.g., the sharp relevance decay of a news article to the daily news) and shifts in a user's preferences [29], specific to news.

B. News Recommendation Methods

Deep learning has been successfully applied to news recommendation. For instance, one effort [30] uses an RNN method with a global encoder very similar to that proposed in [15] (described above) to recommend news articles. Its input consists of the traditional elements: the content of articles and reader browsing history. Moreira et al. [29] propose a two-step algorithm to provide session-based news recommendation by first learning the representation of news articles using their categories, and then applying a regular RNN model to serve as the predictor. The approach is similar to that of Tan et al. [14], except for the procedure of learning the embeddings of news articles. Zheng et al. [28] present a reinforcement learning approach for personalized news recommendation by utilizing news, user, and context features. We do not compare with them as we aim to provide recommendation in anonymous sessions and user information is hence not available. Wang et al. [31] conduct news recommendation by learning both reader embedding and news article embedding using convolutional neural network and knowledge graph, while Wu et al. [32] exploit heterogeneous user behavior via CNN networks and attentive learning framework. These two works solve the problem for news aggregators (e.g., Bing News) and use external knowledge, such as search query logs, none of which are available to news outlets. News outlets however have the user commenting activity, which the aggregators do not have.

In our work, we conduct session-based news recommendation based on users' historical commenting activities at news outlets. We utilize temporal variables like lag of historical events and recency, and incorporate them into an interpretable recency regularized attentive neural method to capture users' evolving interest and uneven commenting activity over time as well as the short lifespan of news article. To our knowledge, none of the above works have these characteristics.

III. Problem Definition

Notations: We introduce the key concepts and define the session-based news recommendation problem in this section. We will use bold lowercase letters for vectors (e.g., $\boldsymbol{h}_1$), and normal lowercase letters for scalars (e.g., w_1).

An *event* is defined as a reader's action of commenting on a news article, and a *session* is defined as a sequence of events. Let $\mathcal{A} = \{A_j | j = 1, \cdots, n\}$ be a set of n distinct news articles (i.e., $|\mathcal{A}|$ = n). t_j^P denotes the publication time of news

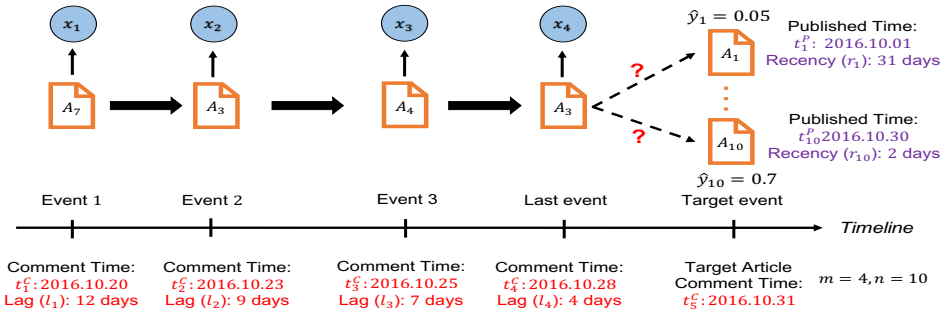

Fig. 1: An example of a session with 4 historical events and 10 news articles in news recommendation ($m = 4, n = 10$).

article A_j. Let $[x_1, \cdots, x_m, x_{m+1}]$ denote a session (S) of historical events ordered by time, where x_i is the index of the news article commented in event i. We use t_i^C to denote the commenting time on the same article in event i. The *last event* and the *target event* correspond to event m and event m+1 in S, respectively. The *lag* l, which is the time interval between the historical event and the target event, is calculated as $l_i = t_{m+1}^C - t_i^C + 1$, for event i. The *recency* r of article A_j, which is the "age" of article A_j at the time when *target event* occurs (t_{m+1}^C), is calculated as $r_j = t_{m+1}^C - t_j^P + 1$.

The problem: We are given the sequence of historical events in an anonymous session, the pre-computed lags for historical events and recency for each article in $\mathcal{A}$. There is no other information for each user besides the comments he left. The recommendation task is to predict which article a user will comment in the target event. $\boldsymbol{y} \in \mathbb{R}^n$ denotes the target label, where $y_j = 1$ if A_j is the target article, and 0 otherwise. The prediction is denoted by $\hat{\boldsymbol{y}}$, where $\hat{y}_j$ is the probability that A_j is the target article. Figure 1 gives an example of a session with 4 historical events and 10 articles. In Figure 1, all articles are marked in orange and their corresponding representations are marked in blue. Lag and commenting time of news articles in historical events are marked in red. Published time and recency for articles are marked in purple. In this example, the session contains a sequence of user's commenting activities on A_7, A_3, A_4 and A_3 again. The prediction score for A_1 and A_{10} are 0.05 and 0.7, respectively. A_{10} is very likely because A_{10} is a fresher article (with smaller recency).

IV. The Proposed Methods

We describe our proposed recency regularized attentive neural model in this section. We describe our methods to capture user-article-comment temporal variables, such as, user's time-varying interest, user's irregular commenting activity, and limited lifespan of an article.

A. Sequential Behavior Modeling

The traditional collaborative filtering methods are not feasible candidates in our temporally dynamic setting. They however work well in stationary settings. In this work, we use RNN to model the user's sequential behavior while consuming news. In particular, the information from previous user actions is propagated through the hidden unit of previous RNN cell to the current RNN cell.

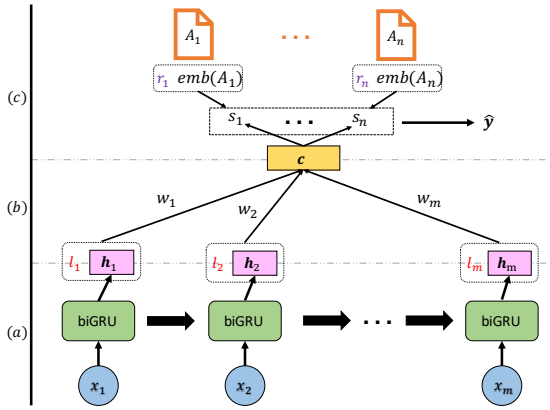

Fig. 2: The architecture of recency regularized attentive neural model has three layers: layer (a) is the sequence modeling, layer (b) is the attention mechanism, and layer (c) is the recency regularized decoding.

We use Gated Recurrent Unit (GRU) [33] in our design, as GRU is able to learn when and by what weight we need to update the hidden unit in the recurrent cell. Bidirectional extensions further enhance recurrent models by considering input sequences from both past and future during training. Given its advantages of sequence modeling, (bidirectional) GRU is the preferred building block for solving general session-based recommendation problems [13], [14]. For ease of representation, the bidirectional GRU (biGRU) model is,

$$[\boldsymbol{h}_1, \cdots, \boldsymbol{h}_m] = biGRU([x_1, \cdots, x_m]) \quad (1)$$

where x_i, the index of the commented article in the i^{th} event, is the input of the biGRU cell i, and $\boldsymbol{h}_i \in \mathbb{R}^{d_h}$ is the corresponding hidden unit of the biGRU cell i, where d_h is the hidden size. In particular, the hidden unit $\boldsymbol{h}_i$ is the sum of the forward hidden unit $\overrightarrow{\boldsymbol{h}_i}$ and the backward hidden unit $\overleftarrow{\boldsymbol{h}_i}$. The chain structure of RNN is effective in capturing the time-evolving interest of a user.

With this RNN architecture, one can make prediction using the last hidden unit, i.e., the output of the last RNN cell $\boldsymbol{h}_m$ in the RNN sequence. The prediction is accomplished with a linear transformation on the hidden output

$$\hat{\boldsymbol{y}} = softmax(\Theta \boldsymbol{h}_m + \boldsymbol{\gamma}) \quad (2)$$

where $\Theta \in \mathbb{R}^{n \times d_h}$ and $\boldsymbol{\gamma} \in \mathbb{R}^n$ are the parameters.

B. Neural Attentive Framework

Although sequence data can be naturally handled by recurrent models, a user's general interest is not adequately captured. Inspired by the successful application of attention mechanisms in other domains, e.g., machine translation [34]–[36], we assume that *all* historical user's actions (behaviors) are important for a comprehensive depiction of a user's general interest over time. Nevertheless, we need to put different emphasis on a user's actions over time as some are less informative than others. We thus propose a general attentive neural model for our problem. Similar to the biGRU model introduced in Section IV-A, our approach here is to firstly model the sequence data using biGRU. This is illustrated in Figure 2(a). However, instead of relying on the hidden unit for the last cell ($\boldsymbol{h}_m$) to make prediction, the attentive model is capable of learning the attention weight of each hidden unit and integrate them. Figure 2 (b) depicts this process. The attention weight for hidden unit $\boldsymbol{h}_i$ is denoted by w_i. And w_i is calculated via a function of hidden units and associated temporal variables, like *lag*. It is formulated as

$$w_i = \frac{\exp\left(attn(\boldsymbol{h}_i, \boldsymbol{h}_m, l_i; \Lambda)\right)}{\sum_{j=1}^{m} \exp\left(attn(\boldsymbol{h}_j, \boldsymbol{h}_m, l_j; \Lambda)\right)} \tag{3}$$

where Λ is a set of parameters (that need to be learned), $attn(\cdot)$ is the function for calculating the attention score, and the attention weight is the normalized attention score such that $\sum_{i=1}^{m} w_i = 1$. We present more details about their realizations in the following sections. Given the attention weights, the hidden units from all time steps can be weight aggregated into a context vector $\boldsymbol{c} \in \mathbb{R}^{d_h}$, where $\boldsymbol{c} = \sum_{i=1}^{m} w_i \boldsymbol{h}_i$.

Then, we can use the resulted context vector for output prediction. Under this framework, we propose two different designs of attention mechanism by exploiting the temporal variables extracted from the temporal characteristics in news recommendation.

1) Lag-Aware Attention: Here, we describe our approach in modeling the temporal dynamics of a user's irregular commenting activity. The influence of user's commenting behavior on the final prediction is made not only by the content and sequences of comments, but also the time of commenting events. The dynamic values of same commenting contents varies according to time. Intuitively, given two events that occur at the same sequential position in two different sessions, the older event (i.e., with larger lag from the target event) of the two should be given lesser importance in the prediction task than the more recent event (i.e., with smaller lag from the target event). Based on this intuition, the relevance of a historical event need not be decided by its position in the sequence of a session, but it needs to be quantified by the lag between its commenting time and the commenting time of the target event. An example of lag is illustrated in Figure 1 and explained in Section III. In particular, we propose to model the lag-aware attention as

$$attn(\boldsymbol{h}_i, \boldsymbol{h}_m, l_i; \Lambda) = a * l_i + b \tag{4}$$

where $\Lambda = \{a, b | a \in \mathbb{R}, b \in \mathbb{R}\}$ is the set of parameters to learn. As we expect that a user's current interest is more affected by her recent actions (small lag) and less affected by her older actions (large lag), the learned a needs to be negative, so that w is monotonically decreasing with l. We do not add any box constraints to a in our optimization algorithm as we want to prove this hypothesis by learning from the data. (We will show empirically that this hypothesis is strongly supported by the data. The empirical evidence is presented in Section V-F.) The design for lag-aware attention is

$$w_i = \frac{\exp\left(a * l_i + b\right)}{\sum_{j=1}^{m} \exp\left(a * l_j + b\right)} \tag{5}$$

We notice from Equation (5) that the lag of the last event l_m does not affect the attention score. The gap between lags, i.e., $l_i - l_m$, however matters. In this way, we can guarantee that the last event is sufficiently relevant to the target event, while the contributions of other historical events are still weighted by their lags to the last event. Although b is omitted in Equation (5), we keep it in Equation (4) as it can increase the numerical stability by avoiding the explosion in the learning process. We also try modeling the attention weight via $w_i = sigmoid(a * l_i + b)$. But this model performed worse in our experiments, as the attention weights are not normalized. We omit it in the rest of the paper.

2) History-Aware Attention: In addition to the lag-aware attention, we propose another realization of the attention mechanism in which the history-aware attention weight for event i is decided by the joint effort of itself and the last event. Similar designs have been successfully applied elsewhere [15], [35]. The attention function is formulated as:

$$attn(\boldsymbol{h}_i, \boldsymbol{h}_m, l_i; \Lambda) = \boldsymbol{u}^T tanh(V_h \boldsymbol{h}_i + V_l \boldsymbol{h}_m) \tag{6}$$

where $\Lambda = \{\boldsymbol{u}, V_h, V_l | \boldsymbol{u} \in \mathbb{R}_a^d, V_h \in \mathbb{R}^{d_a \times d_h}, V_l \in \mathbb{R}^{d_a \times d_h}\}$ is the set of parameters to be learned. We set d_a the same as the hidden size d_h for ease of hyperparameter tuning. The attention weight is calculated according to Equation (3).

C. Recency Regularized Decoder

In this section, we present our method to capture the temporal dynamics related to news articles: users prefer to read and comment on recent articles given the short lifespan of news articles.

1) Efficient Output Decoding: A potential issue in the output layer of recurrent models is the low efficiency of parameter learning in terms of both time complexity and space complexity. As we can see from Equation (2), the number of scalar parameters in Θ is $n \times d_h$, which can easily become unmanageable with a large set of news articles. Similar issues are observed in other applications [14], [37]. To address this problem, we propose a generalized inner product for output prediction. We calculate the prediction score via the generalized inner product of article embedding and the context vector, $s_j = \langle \boldsymbol{c}\Omega, emb(A_j) \rangle$, where $s_j \in \mathbb{R}$ is the prediction score, $\Omega \in \mathbb{R}^{d_h \times d_e}$ is the projection matrix for the context vector, and $emb(A_j) \in \mathbb{R}^{d_e}$ is the embedding vector for A_j, and d_e is embedding size. In this way, the number of scalar parameters is reduced from $n * d_h$ to $d_e * d_h$ as $n \geq d_e$.

2) Recency Regularization: News articles are typically short lived with a rapid decline in audience interest– as the old saying goes, "today's news is wrapping tomorrow's fish

TABLE I: Datasets of news articles with comments from Daily Mail (DM), Fox News (Fox), and Wall Street Journal (WSJ).

	#events	#articles	#distinct users	#aug. training sess.	#training sess.	#validation sess.	#testing sess.	avg. length
DM	198,272	1,577	30,673	122,625	47,212	4,784	4,773	3.49
Fox	338,380	1,040	28,561	213,738	64,164	8,017	8,018	4.22
WSJ	74,389	1,544	6,353	44,448	15,016	1,864	1,866	3.97

and chips". Without considering the recency of news articles, a user is equally likely to comment on multiple articles with similar contents. However, a more recent article is more likely to receive a comment than an older article is. In other words, the recency of an article needs to be taken into consideration for prediction. An example of the recency (r) is illustrated in Figure 1 and explained in Section III. The recency is set to 0 if the article has not been published yet. In particular, the recency regularizor is formulated as

$$reg(r_j) = \begin{cases} sigmoid(\alpha)^{r_j}, \text{ if } r_j \neq 0 \\ 0, \text{ if } r_j = 0 \text{ (not published yet)} \end{cases} \quad (7)$$

where $\alpha \in \mathbb{R}$ is the parameter to learn, and $reg(\cdot)$ is the regularization function. To ensure that the base of this exponential function falls into the interval $(0, 1)$, we model it via $sigmoid(\alpha)$ instead of learning it directly. With this approach, more penalties are placed on older articles (with large recency) – $sigmoid(\alpha)$ is small. The recency regularized output scoring function is given by $s_j = reg(r_j) * \langle \boldsymbol{c}\Omega, emb(A_j)\rangle$.

The prediction is computed by $\hat{\boldsymbol{y}} = softmax(\boldsymbol{s})$, where $\boldsymbol{s} \in \mathbb{R}^n$ is the vector concatenation of $\{s_j | j = 1, \cdots, n\}$. Figure 2 (c) depicts the entire process of output decoding.

D. Model Learning

Up to this point, we described our approach of incorporating the temporal variables into the designs of the attention and decoder. We also described the entire data flow of the proposed method from input to prediction. As the recommendation is essentially a multi-class classification problem, we use the negative log-likelihood loss function as the objective, which is $L(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\boldsymbol{y}^T \log(\hat{\boldsymbol{y}})$. This objective function is optimized using the adaptive moment estimation algorithm (Adam) [38].

Let RHAM denote our proposed recency regularized history-aware attention model and RLAM denote the recency regularized lag-aware attention model. Their non-regularized variants are denoted by HAM - history-aware attention model and LAM - lag-aware attention model, respectively.

V. Experimental Results

In this section, we present the details of our experimental settings and results. Our main goal is to demonstrate the effectiveness of our proposed approaches against a set of representative baselines. We will also give an in-depth analysis of our proposed methods.

A. Datasets

Our data consist of news articles and their user comments from three major news outlets: Daily Mail (DM), Fox News (Fox) and Wall Street Journal (WSJ). We crawled the data over 2-year period. We removed mediator comments, which usually replace spam, hate and meaningless comments detected by filtering systems. We also removed the news articles with fewer than 5 comments. We break reader commenting sequences into sessions such that all neighboring events from the same session take place within 7 days. It is because we observe that 80% of neighboring events happen within a week regardless of news outlet. We exclude sessions of length 1 and remove the tail events from sessions with more than 10 events. We sort the sessions in ascending order of the comment time of their first event. We use the first 80% of the sessions for training, then uniformly sample half from the remaining sessions for validation, and use the final 10% of sessions for testing. To effectively utilize the label information in the training data, we augment it according to the procedure in [14]. Table I provides the statistics of the data used in our studies.

B. Baseline Methods

We compare our method against the following methods:

POP: It always recommends the most popular articles in the training set [11].

Item-KNN: It recommends the most similar article to the one commented in the last event. The similarity is defined as the co-occurrences of two articles in the training set divided by the square root of the product of the number of sessions in which each articles occurred. Regularization is included to avoid high similarities of rarely commented articles [39], [40].

A2V-KNN: It is similar to Item-KNN. The difference is that similarity is defined as the cosine of article embedding vectors (A2V), which is calculated as the aggregation of word embeddings from pre-trained Google news corpus weighted by the normalized frequencies of words in the article.

STAN: STAN is the state-of-the-art neighborhood-based approach for session-based recommendations. It improves session-based KNN approach by introducing 3 types of decay factors according to the timestamp and sequence of sessions and items [12].

GRU4Rec+: GRU4Rec is an GRU model for session-based recommendation [13]. It employs ranking-loss functions in a session-parallel mini-batch training process. GRU4Rec+ improves GRU4Rec model by considering temporal changes and augmenting the training data [14].

NARM: It is a RNN based neural attentive model which integrates a local encoder and a global encoder to jointly model user's behavior and capture the user's main purpose [15].

STAMP: It is a multilayer perceptron based neural attentive model that captures both users' long-term and short-term memory for users' general and current interests [16].

tLSTM: tLSTM is Time-LSTM that models users' sequential actions. It explicitly models the effect of time interval between

user's neighbor actions by time gates [20].

CHAMELEON: It's a news session-based recommendation system with metadata & text based representation learning module and LSTM-based recommendation module [29].

In summary, we select 8 session-based recommendation algorithms that can be used with datasets. Among them, POP, Item-KNN, A2V-KNN and STAN are non-deep learning methods; the rest are deep learning methods. We implement CHAMELEON, a recent news recommendation algorithm. We believe that our set of baselines is representative of the current spectrum of approaches in this field.

C. Experimental Setup

Evaluation Metric. We use the traditional evaluation metrics for recommendation systems:

- Recall@k: It is the proportion of testing sessions in which the desired target article is among the top k predicted articles in all testing sessions.
- MRR@k: It is the average of reciprocal ranks of the desired target articles in predictions of all testing sessions. The reciprocal rank is set to 0 if it is above k.

Hyperparameters. We tune the hyperparameters with the validation set on the best recall@k during model training and use them on the testing set. The tuned hyperparameters include: the input article embedding, which is either onehot embedding or dense embedding learned within the model; d_h - the hidden unit size, $d_h \in \{100, 500, 1000\}$; the learning rate for Adam, which is set to one of $\{5e^{-2}, 1e^{-2}, 5e^{-3}, 1e^{-3}\}$; and the index of epoch with the best performance. When choosing dense embedding for input, we tune embedding size (d_e) from $\{100, 300, 500, 1000\}$. We use the bidirectional model with one hidden layer without dropout regularization.

D. Effectiveness Evaluation

To demonstrate the effectiveness of our proposed recency regularized methods, we compare them against 8 baselines described in Section V-B on data from three news outlets described in Section V-A. We use Recall@k and MRR@k to evaluate performance. $k = 20$ is a common choice in studies about session-based recommendation [13]–[16], [20]. $k = 5$ is a much harder setting than $k = 20$, which can be used to evaluate the lower bound performance. We evaluate all methods using both settings. Table II presents the outcome for $k = 20$ on the left-hand side and the outcome for $k = 5$ on the right-hand side. The results of the best two methods are marked in bold. Using the results in Table II, we make the following observations:

(1) Our recency regularized methods (RLAM and RHAM) consistently outperform all other methods by at least 40% (in some cases by more than 100%) across all experimental settings on all datasets in terms of both Recall and MRR (marked in bold).

(2) Both RLAM and RHAM achieve at least 0.197 in terms of MRR on all datasets with different k. The MRRs from baselines are less than 0.2 in most experiments. This indicates that, on average, the predictions of our methods fall within the top-5 more often than those of the baselines.

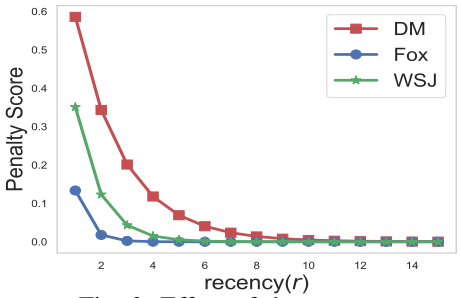

Fig. 3: Effect of the recency.

(3) Among all deep learning baselines, GRURec+ performs the best for $k = 20$, whereas tLSTM performs the best for $k = 5$. $k = 5$ is a more challenging criterion. It is likely that the explicit modeling of time factor provides tLSTM with some advantage over the other algorithms. Overall however, the results of the RNN based methods are close to each other.

(4) STAMP reports better performance than RNN based methods on general session-based recommendation [16]. However, it performs worse in our experiments. It also requires about 100 epoches of training to converge in our experiments, while other RNN based methods need around 30 epoches to converge. Given these observations and the fact that STAMP is not a RNN based approach, the drop in its performance on news recommendation may be explained by its inability to model a user's sequential behavior. This however is where the RNN based methods excel.

(5) Though CHAMELEON is a RNN based model designed for news recommendation, it is not the best performing algorithm among the RNN-based baselines for $k = 5$. It performs worse for $k = 20$. This may be due to the lack of metadata (e.g., categories) as input for article representation learning.

(6) The non deep learning methods achieve inferior results to deep learning approaches. Although these methods are able to model the similarities between articles, they can not leverage sequential signals in the data as effective as deep learning approaches, and they fail to capture user-article-comment connections.

E. Effect of Recency Regularizor

In this section, we study how recency affects the penalty on prediction scores. We plot the penalty score $reg(r)$ against recency r from 1 to 15 using the learned α from the best RLAM model used in our experiments. The calculation of $reg(r)$ is explained in Equation (7). Figure 3 clearly shows that recency is a key factor deciding the value of articles across all outlets. The predictions for fresh articles (with less recency) are less penalized, while penalty is larger for older articles. Moreover, we observe that the effect of recency to the penalty is quite different on different outlets. Specifically, the effect of the recency to the penalty is decided by the base $sigmoid(\alpha)$ in Equation (7). In our results, the bases calculated from the learned α are 0.586, 0.134 and 0.351 for Daily Mail, Fox and WSJ, respectively. It shows that recency is more important to

TABLE II: Effectiveness evaluation using Recall@k and MRR@k, $k \in \{5, 20\}$. The best two results are marked in bold.

Method		$k = 20$			$k = 5$		
		Daily Mail	Fox	WSJ	Daily Mail	Fox	WSJ
		Recall / MRR	Recall / MRR	Recall / MRR	Recall / MRR	Recall / MRR	Recall / MRR
Non DL	POP	0.000 / 0.000	0.019 / 0.002	0.000 / 0.000	0.000 / 0.000	0.014 / 0.002	0.000 / 0.000
	A2V-KNN	0.031 / 0.013	0.090 / 0.034	0.050 / 0.010	0.013 / 0.011	0.085 / 0.033	0.015 / 0.007
	Item-KNN	0.436 / 0.111	0.352 / 0.105	0.206 / 0.062	0.202 / 0.088	0.182 / 0.088	0.108 / 0.052
	STAN	0.471 / 0.119	0.358 / 0.096	0.233 / 0.066	0.203 / 0.093	0.162 / 0.078	0.101 / 0.053
DL	GRU4Rec+	0.571 / 0.227	0.477 / 0.193	0.290 / 0.086	0.351 / 0.203	0.304 / 0.173	0.133 / 0.070
	NARM	0.555 / 0.222	0.450 / 0.193	0.220 / 0.079	0.339 / 0.200	0.296 / 0.177	0.120 / 0.069
	tLSTM	0.533 / 0.246	0.422 / 0.236	0.247 / 0.098	0.365 / 0.228	0.328 / 0.225	0.146 / 0.087
	STAMP	0.508 / 0.106	0.361 / 0.073	0.228 / 0.060	0.215 / 0.100	0.142 / 0.059	0.104 / 0.048
	CHAMELEON	0.570 / 0.230	0.466 / 0.197	0.276 / 0.096	0.344 / 0.207	0.317 / 0.181	0.149 / 0.084
Ours	RLAM	**0.867 /0.426**	**0.824 /0.579**	**0.645 /0.224**	**0.656 /0.402**	**0.753 /0.569**	**0.400 /0.197**
	RHAM	**0.894 /0.410**	**0.857 /0.615**	**0.674 /0.238**	**0.643 /0.383**	**0.757 /0.601**	**0.398 /0.206**

TABLE III: Recall@k and MRR@k on three news outlets.

		Daily Mail		Fox		WSJ	
k	Method	Recall	MRR	Recall	MRR	Recall	MRR
5	LAM	0.352	0.202	0.301	0.166	0.133	0.072
	HAM	0.360	0.209	0.309	0.175	0.143	0.080
	Joint	0.364	0.216	0.310	0.188	0.137	0.071
	RLAM	**0.656**	**0.402**	**0.753**	**0.569**	**0.400**	**0.197**
	RHAM	**0.643**	**0.383**	**0.757**	**0.601**	**0.398**	**0.206**
10	LAM	0.472	0.219	0.427	0.185	0.220	0.085
	HAM	0.480	0.226	0.405	0.188	0.207	0.089
	Joint	0.476	0.231	0.414	0.190	0.205	0.080
	RLAM	**0.788**	**0.420**	**0.818**	**0.579**	**0.549**	**0.217**
	RHAM	**0.801**	**0.404**	**0.849**	**0.614**	**0.563**	**0.230**
20	LAM	0.566	0.225	0.533	0.193	0.287	0.089
	HAM	0.577	0.232	0.468	0.193	0.278	0.094
	Joint	0.571	0.216	0.488	0.207	0.271	0.085
	RLAM	**0.867**	**0.426**	**0.824**	**0.579**	**0.645**	**0.224**
	RHAM	**0.894**	**0.410**	**0.857**	**0.615**	**0.674**	**0.238**

the value of articles in Fox than in others. This appears to be due to increased daily reporting on the U.S. presidential election in 2016 at Fox.

Previously, we presented the effectiveness of our proposed recency regularized neural models, but without knowing the contribution of each module. Here, we conduct an ablation study to explore the contribution of plain attentive neural models. In particular, we repeat the same experimental setting in Section V-D. The results are presented in Table III. From the results, we conclude that: (1) After removing the recency regularizor from RLAM and RHAM, the performance of the non-regularized methods (LAM and HAM) drops across all experimental settings. This observation demonstrates the importance of modeling the dynamics of recency in our problem. (2) Although HAM and LAM are not as good as recency regularized models, they however are still comparable with other deep learning approaches (if we compare the results from Tables III and II). They are also interpretable given their attention mechanism. We specifically discuss the effect of lag-aware attention proposed in LAM in Section V-F. Regarding the attention in HAM, one can also visualize their weights

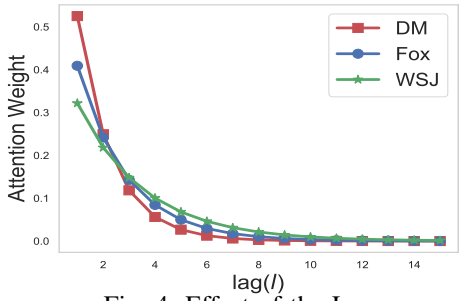

Fig. 4: Effect of the Lag

for further investigation. Exploration on similar attention is discussed in other studies [15], [35]. We do not discuss it further in this paper. (3) In addition, we also investigate the effect of a hybrid attention model which integrates HAM and LAM by concatenating their context vectors. We call this approach Joint and present its performance in Table III. Analyzing the results in the table we note that the joint model is not always better than its two modules due to its heavier modeling. Hence, we do not experiment further with this approach in this work.

F. Interpretable Lag-Aware Attention

In this section, we study the effect of lag l to Lag-Aware Attention weight w. We firstly want to demonstrate the soundness of our hypothesis that *user's recent actions are more relevant to her current interest*. In other words, the parameter a in Equation (5) learned from the data should be negative, such that the attention weight is negatively correlated with the lag. For instance, recent actions with small lags have larger attention weights. The learned a for Daily Mail, Fox and WSJ dataset is -0.745,-0.526 and -0.388 respectively. They are all negative and they are learned without adding any box constraints. We contend that this asserts the soundness of our hypothesis.

To understand the effect of lag-aware attention weight better, we create a synthetic session with event lags ranging from 1 to 15. We plot the attention weight w against lags l with a learned from the best LAM models (Figure 4). We observe

from the figure that more recent events are associated with larger weights in all 3 outlets.

VI. Conclusion

In this paper, we try to predict the news article a user will comment on given her historical sequential behavior from an anonymous session. We treat the problem as an instance of the session-based news recommendation problem. Unlike the general session-based recommendation problem, we observe unique temporal dynamics in reader's commenting activities and news articles in our problem. To capture them, we propose a recency regularized neural attentive framework that seeks to model the temporal variables observed from the time-varying interests of a user. We demonstrate the effectiveness of our approach, in particular that of the recency regularized model, in comparison with state-of-the-art methods on real-world data from three major news outlets. In addition, we show that our approach is interpretable in terms of understanding (i) the effect of temporal variables in the prediction outcome and (ii) their varied behavior across news outlets.

Acknowledgment

This research was supported in part by the following grants: the U.S. NSF BIGDATA 1838145 and 1838147 grants; U.S. ARL subaward 555080-78055 under Prime Contract No. W911NF2220001; Temple University office of the Vice President for Research 2022 Catalytic Collaborative Research Initiative Program; and a gift from NVIDIA Corporation.

References

[1] P. Weber, "Discussions in the comments section: Factors influencing participation and interactivity in online newspapers' reader comments," *New Media & Society*, vol. 16, no. 6, pp. 941–957, 2014.

[2] J. B. Schafer, J. Konstan, and J. Riedl, "Recommender systems in e-commerce," in *EC*. ACM, 1999, pp. 158–166.

[3] A. Zimdars, D. M. Chickering, and C. Meek, "Using temporal data for making recommendations," in *UAI*, 2001, pp. 580–588.

[4] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims, "Playlist prediction via metric embedding," in *KDD*, 2012, pp. 714–722.

[5] G. Shani, D. Heckerman, and R. I. Brafman, "An mdp-based recommender system," *JMLR*, vol. 6, no. Sep, pp. 1265–1295, 2005.

[6] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *WWW*. ACM, 2010, pp. 811–820.

[7] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng, "Learning hierarchical representation model for nextbasket recommendation," in *SIGIR*, 2015, pp. 403–412.

[8] Y. Koren, "Collaborative filtering with temporal dynamics," in *KDD*, 2009, pp. 447–456.

[9] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *RecSys*. ACM, 2010, pp. 79–86.

[10] D. Jannach and M. Ludewig, "When recurrent neural networks meet the neighborhood for session-based recommendation," in *RecSys*, 2017, pp. 306–310.

[11] M. Ludewig and D. Jannach, "Evaluation of session-based recommendation algorithms," *User Modeling and User-Adapted Interaction*, vol. 28, no. 4-5, pp. 331–390, 2018.

[12] D. Garg, P. Gupta, P. Malhotra, L. Vig, and G. Shroff, "Sequence and time aware neighborhood for session-based recommendations: Stan," in *SIGIR*, 2019, pp. 1069–1072.

[13] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

[14] Y. K. Tan, X. Xu, and Y. Liu, "Improved recurrent neural networks for session-based recommendations," in *DLRS*. ACM, 2016, pp. 17–22.

[15] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *CIKM*, 2017, pp. 1419–1428.

[16] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "Stamp: short-term attention/memory priority model for session-based recommendation," in *SIGKDD*, 2018, pp. 1831–1839.

[17] R. Devooght and H. Bersini, "Collaborative filtering with recurrent neural networks," *arXiv preprint arXiv:1608.07400*, 2016.

[18] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, "Recurrent recommender networks," in *WSDM*. ACM, 2017, pp. 495–503.

[19] M. Wang, P. Ren, L. Mei, Z. Chen, J. Ma, and M. de Rijke, "A collaborative session-based recommendation approach with parallel memory modules," in *SIGIR*, 2019.

[20] Y. Zhu, H. Li, Y. Liao, B. Wang, Z. Guan, H. Liu, and D. Cai, "What to do next: Modeling user behaviors by time-lstm." in *IJCAI*, 2017, pp. 3602–3608.

[21] A. Beutel, P. Covington, S. Jain, C. Xu, J. Li, V. Gatto, and E. H. Chi, "Latent cross: Making use of context in recurrent recommender systems," in *WSDM*, 2018, pp. 46–54.

[22] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *ACM SIGIR*. ACM, 2018, pp. 95–104.

[23] H. Jing and A. J. Smola, "Neural survival recommender," in *WSDM*. ACM, 2017, pp. 515–524.

[24] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *AAAI*, vol. 33, 2019, pp. 346–353.

[25] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou, "Graph contextualized self-attention network for session-based recommendation," in *IJCAI*, 2019, pp. 3940–3946.

[26] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *WSDM*. ACM, 2019, pp. 555–563.

[27] P. Ren, Z. Chen, J. Li, Z. Ren, J. Ma, and M. de Rijke, "Repeatnet: A repeat aware neural recommendation machine for session-based recommendation," in *AAAI*, vol. 33, 2019, pp. 4806–4813.

[28] G. Zheng, F. Zhang, Z. Zheng, Y. Xiang, N. J. Yuan, X. Xie, and Z. Li, "Drn: A deep reinforcement learning framework for news recommendation," in *WWW*, 2018, pp. 167–176.

[29] G. d. S. P. Moreira, F. Ferreira, and A. M. da Cunha, "News session-based recommendations using deep neural networks," *arXiv preprint arXiv:1808.00076*, 2018.

[30] S. Okura, Y. Tagami, S. Ono, and A. Tajima, "Embedding-based news recommendation for millions of users," in *SIGKDD*, 2017, pp. 1933–1942.

[31] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: Deep knowledge-aware network for news recommendation," in *WWW*, 2018, pp. 1835–1844.

[32] C. Wu, F. Wu, M. An, T. Qi, J. Huang, Y. Huang, and X. Xie, "Neural news recommendation with heterogeneous user behavior," in *EMNLP-IJCNLP*, 2019, pp. 4876–4885.

[33] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[34] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[35] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[37] A. Mnih and G. E. Hinton, "A scalable hierarchical distributed language model," in *NIPS*, 2009, pp. 1081–1088.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[39] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston *et al.*, "The youtube video recommendation system," in *RecSys*. ACM, 2010, pp. 293–296.

[40] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, no. 1, pp. 76–80, 2003.