News Recommender System Considering Temporal Dynamics and News Taxonomy

Shaina Raza
Department of Computer Science
Ryerson University
Toronto, Canada
shaina.raza@ryerson.ca

Chen Ding
Department of Computer Science
Ryerson University
Toronto, Canada
cding@ryerson.ca

***Abstract*—In the past, news recommender systems have been built to recommend list of news items similar to those that a user has accessed before (content-based); or similar to those that have been read by similar users (collaborative filtering). However, the highly volatile nature of the news content and the dynamic and evolving user preferences are either ignored or not taken into full consideration in these systems. In a news recommender system, it is very likely that a user's short-term interest or preference may have a sudden change due to an emerging social or personal event or breaking news while their long-term interests may change gradually or remain. For these long-term interests of the readers, it is often more appropriate to associate them with news categories than with individual news items. In this paper, we propose a biased matrix factorization model with consideration of both temporal dynamics of user preferences and news taxonomy to build a news recommender system. By conducting an extensive experiment on a collection of news data, we demonstrate the effectiveness of our proposed model against traditional matrix factorization models as well as other neural recommender baselines. The findings from our experiments show that news category is an important factor when readers choose news articles to read, and temporal factors with consideration of different temporal resolution also play a role in this process.**

Keywords—Temporal dynamics, Taxonomy, Categories, Biases, News recommendations, Temporal effects, Matrix factorization, Latent factors

I. INTRODUCTION

Nowadays news reading becomes much more prevalent in people's daily lives than it has ever been. Big names such as Yahoo!, Google, CNN have introduced online news portals which users can access anytime from anywhere to browse through different news categories to find most up-to-date news. However, the challenge for the news readers is to find the right content. With a massive amount of news available online, it becomes difficult and time-consuming to select interesting news to read. To overcome this information overload problem, a news recommender system offers promising solution to help users find most relevant news articles in a timely manner according to their interests.

In the current state of the art of news recommender systems, news stories that users have read in the past are used to infer their interests and preferences. As we know, there is usually frequent and dynamic content updating in a news domain, and it is also quite possible that a user's reading preferences would change over time. This change can be short-term due to breaking news or trending event, or due to a user's short-term interest drift. It can be long-term as the result of a change in a person's life such as a new stage in life, a new job. It can also be seasonal such as interest in sales news or travel news during holiday seasons. However, this type of temporal dynamics is often not taken into full consideration in current systems, although timestamps of the news reading activities are included in some models either as damping factors [1] or as time-related contextual features of news items [2].

On top of that, it is also important to consider the temporal resolution. Different resolution may cause different temporal effects in the news consumption patterns. Previous studies have reported the frequency change in reading activities when different temporal resolution is used. For example, in terms of hour of the day, the highest news consumption usually occurs in the evening from 6-8pm, followed by a peak at 7am or sometimes around the noon [3]. In terms of days of the week, news consumption is fairly flat except for a slight increase on Mondays and a slight decrease on Saturdays [3]. When analyzing user preferences, we should also take the temporal resolution into consideration, for example, the interest shift during the hour of the day, day of the week or week of the year.

Another issue is that they often associate the interest of a news reader with a specific news story or the keywords it contains. However, because the life span of a news item is usually very short, if we simply look at user's ratings on individual news items, it might not be easy to get a clear view of user's general interest for the benefit of the future recommendations. If we link these ratings to the metadata of the news items such as authors, sources, topics, or news categories, we may be able to identify the pattern. For example, if all Sports news got high ratings from a user, we may conclude that this user likes Sports news and recommending any Sports news could satisfy the user. Therefore, we should associate user interest (especially the long-term ones) with news category (based on a pre-defined news taxonomy) [4] or topic [1], not just a single item.

Common news categories include Sports, Politics, Entertainment, Travel, etc., and some sports-related topics could be Blue Jays, David Beckham, French Open, etc. As we can see from the example, when comparing between category and topic, category is better associated with user's long-term interest and is

This work is partially sponsored by Natural Science and Engineering Research Council of Canada (grant 2015-05555)

978-1-7281-0858-2/19/$31.00 © 2019 IEEE
978-1-7281-0858-2/19/$31.00 ©2019 IEEE

often more stable, whereas topic can be highly dynamic. For instance, French Open happens once a year, a reader interested in Sports (or more specifically Tennis) may only read news articles related to French Open for two weeks every year. So, if we use topics to capture a reader's interest, the topic "French Open" is considered as a short-term interest. However, if we use categories, the category "Sports" or sub-category "Tennis" could be considered as long-term interest because the reader also reads other sports or tennis news throughout the year. In short, categories are meant for broad grouping of the news and can help readers find the right type of content to read, whereas the topics are meant to describe specific details of news (but not as detailed as the keyword-level description).

Through this work, we would like to address these two above-mentioned challenges: (i) news consumption is driven by temporal factors, and (ii) news consumption is biased towards a certain type of news. We can model news readers' temporal reading interests by capturing their long-term and short-term interests. The long-term users' interests can be utilized to identify the type (category) of news that users are interested in, and then the short-term interests help the recommender system filter specific news items for individual users at that particular time.

In this paper, we propose a matrix factorization model that incorporates users' ratings driven by different temporal factors and captures information related to news categories. In that, we aim to model temporal dynamics of user behavior in the context of personalized news recommendations. We consider the effects of different temporal resolution (year, month, day, hour, minute, second) along with full timestamp of user's rating in our model and treat each time unit as one variant of the model. We also consider the category information (based on a pre-defined news taxonomy) in different ways, including hierarchical categories (categories with subcategories), flattened categories and a single-level category (only top-level categories). By incorporating time and category information, we can unveil the temporal effects in users' reading patterns.

In the past decade, the deep-learning techniques have achieved great success in image recognition, translation, and more recently various natural language understanding tasks. They have also been widely used in recommender systems. However, according to some recent studies [5] deep-learning based recommender systems may not achieve a better performance compared with some traditional models such as matrix factorization with optimized parameters. So, in the current stage, we would like to build our new addition on top of the biased matrix factorization model. In the experiment, we also include a few neural recommender systems as baselines. Our experimental results show that our fine-tuned model can achieve either better or comparable results compared to these neural recommenders. It also has the advantage on computational efficiency.

In summary, the contributions of this paper are:

- We propose a news recommender system that considers the temporal dynamics in user preferences and interests and utilizes the category information from a news taxonomy to improve the prediction accuracy;
- We implement a rich bias model that incorporates biases related to user-item interactions, the temporal bias in different time units and the categorical bias;
- We conduct a comprehensive experiment to investigate the performance of different variants of our proposed model against traditional as well as deep-learning baselines to find out which one gives us the best result under different circumstances.

The rest of the paper is organized as follows. Section II discusses the related work. Section III explains our proposed methodology. Section IV gives details about our experimental setup, evaluation metrics and information about different hyperparameters used in the model. Section V shows and analyzes the results. Finally, section VI gives the conclusion and briefly discusses future work.

II. Related Work

We review the state-of-the-art literature in this section.

A content-based recommender system tries to recommend newly published news articles based on user feedback data collected from the past reading history. The news profile and the user profile are often represented through well-known methods such as the vector space model (e.g. TF-IDF), topic model (e.g. Latent Dirichlet Allocation) or knowledge graph representation learning [6]. Content-based methods usually perform well when there are plenty of historical records.

In collaborative filtering approach, user modeling is based on similarity between the items or users in terms of their ratings and does not require the extraction of item features [7]. Well-known techniques include neighborhood (e.g., K-Nearest-Neighbors) [8] or model-based methods (e.g., matrix factorization) [9]. Collaborative filtering requires sufficient historical interactions from enough number of users to make effective recommendations.

Hybrid recommendation techniques combine the benefits of both the content-based and collaborative filtering methods to make better recommendations. There are some news recommender systems that make use of hybrid approach [8], [9] to tackle the shortcoming of each individual model. However, hybridization alone cannot improve recommendations in a news environment unless we consider the dynamic characteristics of news content and readers' preferences in the recommendation process.

Temporal dynamics in recommender systems refer to the concept of accurately capturing user preferences over time [10]. The well-known work in this field is based on collaborative filtering techniques. In one of the initial work, Ding and Li [11] computed the time weights on different items in such a way that the time function (with weights) is decayed on historical data. A different strategy without the time-decay weights was proposed by Koren [10], who modeled the temporal dynamics along the whole time period. Koren proposed a matrix factorization model timeSVD++, adding time-dependent rating bias for each user to improve the accuracy of recommendations. Xiong et al. [12] extended the matrix factorization model to include time as the third dimension (a tensor) which is a parameter-free approach.

There are also some research efforts that address temporal dynamics in content-based recommender systems. They often use graph-based approaches to exploit temporal influences for time-aware recommendation [6], [13], [14]. In a graph-based approach, the user-item matrix is represented as a bipartite graph and the time factor is included as a new node, thus making a tri-partite graph.

All the above-reviewed work is for general recommender systems. There is very limited work on news recommender systems that addresses the temporal dynamics. Lommatzsch et al. [2] discussed different recommendation frameworks with the capability of incorporating the dynamics of user preferences in a news recommender system. Li et al. [1] proposed a content-based recommendation approach using a user-item affinity graph to address temporal dynamics. Wang et al. [6] proposed a content-based deep learning model for CTR prediction to make news recommendations. They also used max-over-time-pooling (a sample-based discretization process in deep learning) to train their neural model for temporal dynamics.

In the state-of-the-art general recommender systems, the category information is incorporated in the matrix factorization model in two distinct ways: flat category structure [4] and category hierarchy [15]. Wang et al. [6] proposed to associate each word in the news titles with an entity found in a knowledge graph. Though not exactly the taxonomy information, their proposed DKN model can help distinguish knowledge entities in news content.

In our work, we use the collaborative filtering technique to incorporate temporal dynamics into our proposed model. The reason for choosing collaborative filtering over content-based approach is that we prefer the content-free nature of collaborative filtering algorithm so that we don't have to overload our models with bulks of news data. Also, the consumption data has recorded the temporal patterns of user preferences and it provides a rich source to analyze the dynamics in users' preferences so as to improve our recommendations.

Our work is different from the Koren model [10] that is based on stationary time periods, Ding and Li [11] model where older consumption data is always discarded and Wang et al. [6] model that includes the subset of data within time-decay windows. Our model is built using the technique of temporal analysis in which we propose to include time varying factors along with full temporal ordering while modeling users' preferences. For the taxonomy information, our work is different from Koenigstein et al. [4] model in which the taxonomy is compressed beforehand, Sun et al. [15] model where the hierarchical information is represented as the combined effects of users and items, and Wang et al. [6] model where the knowledge entities cannot easily and accurately capture users' long-term preferences because they are similar to topics which are often more suitable for representing users' short-term interests. In our proposed model, we consider the biases associated with the category information at different levels (parent category, sub-category) and the temporal effects on how a user can be influenced by his preferred categories.

To the best of our knowledge, there is no work in news recommender system that simultaneously considers temporal dynamics and category information through biased modeling to represent readers' short-term and long-term preferences. Though our model is based on simple method, i.e. matrix factorization, we have trained our models with hyperparameter optimization to see if they can achieve better or comparable performance compared with computationally complex neural models.

III. METHODOLOGY

In this section, we first formally define the problem and the notations that we will be using in this paper. Then, we explain our proposed recommendation model and its different variants.

A. Problem Definition

Here we formally define the problem we would like to solve in this work. Given users, items, news taxonomy and timestamp, we want to build a news recommender system in which we first predict the ratings of a specific user on unrated news items with the consideration of news taxonomy and the temporal information associated with user ratings, and then recommend a list of news items to the target user based on predicted ratings.

B. Notations

We have used following set of notations shown in Table I.

TABLE I. NOTATIONS

Notation	Description
P	Set of users
Q	Set of items
r_{ui}	Rating given by user *u* to item *i*
$\hat{r}_{ui}$	Predicted rating from user *u* on item *i*
p_u	Vector of user *u*'s latent factors
q_i	Vector of item *i*'s latent factors
$q_i^T p_u$	Dot product of user and item factors to show the interaction between user *u* and item *i*
μ	Overall average rating
b_{ui}	Bias (observed deviation) involved in rating $\boldsymbol{r_{ui}}$
b_u	Bias of user *u*, a user-specific bias component
b_i	Bias of item *i*, an item-specific bias component
$b_{i,c(i)}$	Bias of item *i* associated with the its category *c*
$b_{i,sc(i)}$	Bias of item *i* associated with its subcategory *sc*
$b_{i,fc(i)}$	Bias of item *i* associated with the flattened category *fc* – category concatenated with subcategory
b_c	Bias of category *c*, a category-specific bias component
$b_{c,i(c)}$	Bias of category *c* associated with item *i* which belongs to category *c*
$b_{c,sc(c),i(c)}$	Bias of category *c* associated with item *i* which belongs to a subcategory *sc* of category *c*
b_{fc}	Bias of flattened category *fc*
$b_{fc,i(fc)}$	Bias of flattened category *fc* associated with item *i* which belongs to *fc*
$b_{u,t(u,i)}$	Bias of user *u* associated with the timestamp *t* of rating on item *i* by user *u*
x_i	Side information on item *i* for modeling any additional features
y_u	Side information on user *u* for modeling any additional features
w_u	Weight term for user *u*
w_i	Weight term for item *i*
m_i	Weight vector for side information vector $\boldsymbol{x_i}$
λ_2	Regularization parameter for interaction term $q_i^T p_u$
$t_y,\ t_m,\ t_d,$ $t_h, t_{min},\ t_S$	Timestamp *t* with time units in year 'y', month 'm', day 'd', hour 'h', minute 'min', and second 's' respectively

There are two major types of data we have used to create our models. One is the observation data that is the portion of the dataset used for training the model, e.g., in a user-item rating matrix, user and item are the main observation data. The other is the side information that refers to any additional features beyond the user and item information, e.g., time of the day when the user rated the item can be included as the side or contextual information with the user or item data. For every notation related to bias, when subscript has two parts (separated by comma), the first part refers to main observation data whereas the second part refers to the side information associated with the observation data. For example, in $b_{i,c(i)}$, item i is the main observation data whereas the category of the item c(i) is considered as the side information of item i. The difference between $b_{i,c(i)}$ and $b_{c,i(c)}$ is that in the former, item is the main observation data with category as the side information and vice versa in the latter.

C. Proposed Model

In this section, we discuss our proposed model step by step to show how a basic matrix factorization model [16] is enhanced to add novel information in the form of temporal dynamics and news taxonomy in a couple of ways.

1) *Biased matrix factorization with consideration of news taxonomy*

Matrix factorization is one of the most popular recommendation algorithms which got its first recognition in Netflix competition [16]. Matrix factorization can be used to discover the latent features that exhibit in the interactions between two different types of entities (e.g., users and items). It approximates the rating given by a user on an item, following the rule [16] as shown in equation (1):

$$\hat{r}_{ui} = q_i^T p_u \tag{1}$$

The interaction term $q_i^T p_u$ describes the interaction between the user and the item, i.e., the degree of user's preference on the item. The observed variations in rating values happen to occur because of the effects associated with either the users or items known as biases. The bias term only depicts the effect of one entity on the output and does not consider the interaction between a user and an item. A user's bias represents the user's tendency to give higher or lower ratings than the average. Similarly, an item's bias refers to how the item is rated compared to the average across all items. In our work, we provide the biases for both the observation data and the side information.

The bias term refers to a user or item's bias towards higher or lower ratings [16]. For example, a consistently highly-rated item would have a higher bias coefficient, whereas a consistently low-rated item would have a lower coefficient to account for this bias. These coefficients can be learned when training the model. A baseline predictor, which represents the effects associated with biases without involving user-item interactions [10], for an unknown rating r_{ui} is denoted by b_{ui} as shown in equation (2):

$$b_{ui} = \mu + b_i + b_u \tag{2}$$

In our work, we propose a novel bias model that involves item biases associated with news taxonomy. For example, news items under a certain news category may get higher ratings than the average. In other words, we may have a news category or sub-category where all the news items are rated higher than average, which shows the biases attached to that category. The reason of including category-related bias into the matrix factorization model is to undo the effects of the biases on a certain category of news items so that all the news items seem to behave more or less similarly in the system. We expand equation (2) to include new category-related biases as shown in equation (3):

$$b_{ui} = \mu + b_i + b_u + b_{i,c(i)} \tag{3}$$

In equation (3), the new addition is the shared bias component between the items and the news category. In the next section, we will consider different ways to define category-related bias terms, and we will also consider different ways to combine observation data and side information when including category in the model.

2) *Adding temporal dynamics into biased matrix factorization model*

The news recommender system is characterized by a distinctive property of timeliness, which may cover many different aspects such as novelty, trend, popularity and the timely delivery of the news. Each user's rating is marked by a timestamp. We include this temporal information to model evolving users' preferences. Users do change their ratings over time. For example, a reader might be quite interested in reading news stories related to politics and tend to give an average rating of '4' to news items under politics. However, after some time, changes might happen (e.g., change in family structure, location, new interests), which are reflected in his ratings, so he might start to give an average rating of '3' to political news. We further expand equation (3) to include time-dependent factors associated with user's rating as shown in equation (4):

$$b_{ui} = \mu + b_i + b_u + b_{i,c(i)} + b_{u,t(u,i)} \tag{4}$$

In equation (4), we consider the time-varying information in the form of biases associated with user's ratings on items. The bias component is common to all ratings of the user in the same timestamp when he rated item i. We define different temporal resolution to include in our proposed model. We have the main timestamp t that we add as the bias for the user's rating on an item. Then, we isolate the temporal components attributed to different time resolution and add them as the fine-grained contextual condition (side information). Our proposed model can include any of these different combinations of time-varying factors: full timestamp t, timestamp t with contextual condition year t_y, month t_m, day t_d, hour t_h, minute t_{min}, or second t_s. Thus, we propose that $b_{u,t(u,i)}$ can be any of these time-varying factors. In general, we can use coarser-level temporal resolution such as year, month or day to capture the steadily changing temporal dynamics. To capture the rapidly changing temporal effects, we can use finer-level temporal resolution such as minute or second.

Our full bias model with time-varying factors is already shown in equation (4), where the category c, and the temporal information t are the side (contextual) information which are passed as vectors along with both user and item vectors. If we represent the side information vector for item i as $\boldsymbol{x_i}$ and user u as $\boldsymbol{y_u}$, we can modify our equation (1) to include the side information as shown in equation (5):

$$\hat{r}_{ui} = q_i^T p_u + x_i + y_u \tag{5}$$

We derive different variants of our model by passing different sets of information (observations, side information and biases for both) and hyperparameters (regularization terms, number of iterations, number of latent factors) into the model. We also propose one unique temporal model corresponding to each unique time-varying factor. Our objective for defining different temporal models is to test different ways of modeling user preferences and to identify the model with best recommendation accuracy in different scenarios. We have not given any specific names to these temporal models because the timing information is shared and common. However, in the later part of this section, we differentiate our temporal models based on varying biases on the items and the news taxonomy.

Apart from temporal modeling through different time-varying factors, we classify our models into three groups: basic temporal model, item-dominant temporal models, and category-dominant temporal models. In the first group, news taxonomy is not considered. In the second and third groups, news taxonomy is considered, and different category-related bias terms are defined. The difference between these two groups is that the input to the former is a user-item rating matrix and to the latter is a user-category matrix. This classification of temporal models is based on the arrangement of biases in the model. Below, we give explanation to each group of models.

Simple Matrix Factorization Model

M1: Simple user-item biased model

$$b_{ui} = \mu + b_i + b_u + b_{u,t(u,i)} \tag{6}$$

This is a plain matrix factorization model with item and user biases. In this model, we consider user *u* and item *i* as the main observation data whereas the time-varying factor is the side information obtained through the user-item interactions. The biases b_i and b_u are associated with the observation data whereas $b_{u,t(u,i)}$ is the bias associated with the side information.

Item-Dominant Models

M2: Item-category-subcategory biased model

$$b_{ui} = \mu + b_i + b_u + b_{i,c(i)} + b_{i,sc(i)} + b_{u,t(u,i)} \tag{7}$$

This model is characterized by the addition of category as the side information whose bias is then added to the item bias. We also add the bias of fine-grained subcategory information with the item bias. In this model, b_i and b_u refers to biases with the observation data whereas $b_{i,c(i)}$, $b_{i,sc(i)}$ and $b_{u,t(u,i)}$ are the biases of the side information. The assumption here is that a news item is characterized by the categories it belongs to in the form of a linked taxonomy.

M3: Item-category biased model

$$b_{ui} = \mu + b_i + b_u + b_{i,c(i)} + b_{u,t(u,i)} \tag{8}$$

In M3 model, we have the item bias associated with just one level of category (no subcategory). The assumption here is that the inherent relationships among categories (when considering subcategories) may lead to complexity in terms of item modeling and may affect the accuracy of the model. So, for the sake of simplicity, we compute item biases under one level of category.

M4: Item-flattened-category-subcategory biased model

$$b_{ui} = \mu + b_i + b_u + b_{i,fc(i)} + b_{u,t(u,i)} \tag{9}$$

In M4 model, we have the item bias associated with the side information from the linked taxonomy. Like M3, the assumption here is to make the model less complex but at the same time benefit from rich information of news taxonomy. Unlike M2 in which category and subcategory information are passed as separate biases, here we compress (flatten) the news taxonomy and associate it with the item bias.

Category-Dominant Models

M5: Category-item-subcategory biased model

$$b_{ui} = \mu + b_c + b_{c,i(c)} + b_{c,s(c),i(c)} + b_u + b_{u,t(u,i)} \tag{10}$$

M5 model requires taking category information as a set of observations, and the item and subcategory information is added as the side information. This model is different from M2 model where the bias of the news taxonomy is added with the item bias. In M5, the biases of item and subcategory are added with the category bias.

Typically, in the matrix factorization model, we take a user-item rating matrix as the input. We now consider the user-category matrix in M5 as well as other models in this group. The user-category matrix is built by taking the ratings of the user for a particular category associated with a news item. The intention is to focus on the general information need of news readers. For example, while making recommendations for news items under a certain category, we could tell user's tendency to give higher or lower ratings than average to that category. Subcategory is a finer level depiction of a category, e.g., Soccer is a subcategory under category Sports. So, we can add subcategory information to reflect finer-level tendency of users towards a certain group of news items.

M6: Category-item biased model

$$b_{ui} = \mu + b_c + b_{c,i(c)} + b_u + b_{u,t(u,i)} \tag{11}$$

Different from M5 model, the M6 model does not include the subcategory information. Here we have the item added as the bias of category. The purpose here is the same as in M3, which is to reduce model' complexity.

M7: Flattened-category-subcategory-item biased model

$$b_{ui} = \mu + b_{fc} + b_{fc,i(fc)} + b_u + b_{u,t(u,i)} \quad (12)$$

Similar to M4, here we compress the category-subcategory information. The item bias is added to bias of the flattened categories. The intention here is to simplify the model and at the same time benefit from the rich taxonomy information. Each of our models [M1-M7] is tested on all the time-varying factors (different temporal resolution), thereby making a combination of 49 variants. In the later experiment section, we choose to demonstrate the results by taking the most appropriate time-varying unit for each model.

3) *Regularization Terms*

As we increase the number of latent factors in our model, it could improve accuracy. However, the number of factors may become too high at which point the model begins to overfit. In order to avoid overfitting, we add the L2-norm (ridge regression) [17] as the regularization term, which adds the squared magnitude of coefficient as penalty term to the loss function as shown in equation (13).

$$min_{P,Q} = \sum_{u,i,r_{ui}} (\hat{r}_{ui} - r_{ui})^2 + \lambda \left(||Q||_2^2 + ||P||_2^2 \right) \quad (13)$$

We have also used different values of λ to control the strength of L2-norm discussion in section V.

4) *The Complete Proposed Model*

Our complete matrix factorization model is defined in equation (14) as follows:

$$\hat{r}_{ui} = b_{ui} + q_i^T p_u + x_i + y_u \quad (14)$$

Here, b_{ui} is our temporal biased model shown in equation (4) and could be elaborated by any of our proposed models [M1-M7]. The $q_i^T p_u$ is the personalization model [16] that explains the interaction of a user with an item. We have also captured the higher-order interactions (x_i, y_u) among our dataset features beside the user-item factors. This makes our model capable of training latent factors from more combination of features in the model.

In equation (14), the side information for the item and the user is represented by x_i and y_u respectively. We also introduce weights (coefficients) in our model for user-item interactions where w_u and w_i are the weights for the user and item respectively. The weight vectors for the item's side information x_i is represented by m_i and for user's side information y_u is represented by n_u. Our final model after including weights is shown in equation (15):

$$\hat{r}_{ui} = b_{ui} + w_u + w_i + n_u y_u + m_i x_i + q_i^T p_u \quad (15)$$

Thus, we can state that our model is able to make rating predictions as a weighted combination of user and item latent factors, side information, biases and their combinations.

5) *Stochastic Gradient Descent (SGD)*

SGD is a popular approach to learn latent factors where the algorithm loops through all the ratings (make several passes through the training set) and updates the relevant latent factors each time [16]. Since our dataset spans over months, we can demonstrate that news readers' preferences over certain items or categories tend to drift at a certain rate. Inspired by the experiment of Koren [10], we also use SGD optimization to tune the model parameters. We test different combinations of time-varying factors in our model using SGD to train a single training example per epoch. Later SGD orders all training examples by their time. This continues till the model training is complete so that the learned parameters begin to reflect the latest time point. In this way, we can learn the latest time point that best suits our test set. The observation data, side information, biases and weights for all models [M1-M7] are incorporated directly into the SGD and are updated using the rule as in equation (16).

$$min_{w_u,w_i,n_u,m_i,P,Q} = \sum_{u,i,r_{ui}} (\hat{r}_{ui} - r_{ui})^2 + \lambda_1 \left(||w_u||_2^2 + ||w_i||_2^2 + ||n_u||_2^2 + ||m_i||_2^2 \right) + \lambda_2 \left(||Q||_2^2 + ||P||_2^2 \right) \quad (16)$$

In our models, we have used two lambda constants λ_1 and λ_2 associated with L2-norm shown in equation (16). The constant λ_1 is the parameter of regularization for linear terms (weight vectors for the interactions and side information), whereas λ_2 is parameter to be used with user-item interactions (with biases).

IV. Evaluation

In this section, we describe the dataset used in the experiment and explain evaluation procedure and metrics

A. Dataset

It was not a trivial task to find a suitable dataset to evaluate our proposed recommendation model because most of the standard datasets available for news recommender research are either too small, sparse, or void of temporal information or rating data from the users. After some extensive research, we found that the New York Times data would be the most suitable to our problem in terms of the number of users, news articles, temporal information, news taxonomy and rich user-item features. For the experiment, we collected the news data using New York Times API1 by retrieving the URL of the articles. We used a Python package2 that serves as API Wrapper for New York Times to retrieve the comments on news articles. Those comments were retrieved with respect to the timeline of retrieved news data. We thoroughly analyzed our dataset and kept users who have commented on at least 10 items, and the news items that were commented by at least 10 users. This has given us more than 240,000 users and over 2 million comments.

We used a rule-based sentiment analysis tool VADER [18] that treats each comment as a document and returns a correlated rating which is an overall score of three types: neutral, positive

1 https://developer.nytimes.com/indexV2.html

2 https://github.com/AashitaK/nyt-comments

and negative scores, corresponding to the text. Here, we tokenized the comment text and represented it as a bag-of-word vector to get its sentiment score. Since time is the critical part of our problem definition, we divided our dataset into multivariate (more than one time-dependent variable) time-series data. Through time-series data, we were able to add fine-grained timing information in the form of contextual conditions to introduce time-varying factors into our models. Time-varying factors helped us investigate the temporal patterns and evolution of user preferences. Among available news item features, we chose news ID, category and subcategory information of the news item. As for the user features and interactions, we chose user ID, comment ID, comment body (text) and most importantly the time when the comment was given by the user. Table II presents some statistics of our dataset.

TABLE II. DATASET STATISTICS

Characteristics	Description
Domain	News
Users	241050 (unique)
Items	10,000 (unique)
Interaction type	Comments
# of interactions	2,000,000
Features	34
News features	16
Time duration	2 years
Taxonomy	2-level (category and subcategory)

B. Baselines

We have compared our models with two types of baselines: (i) Non-neural traditional matrix factorization and (ii) Deep learning models. The baseline models were tuned using the optimal settings as shown in the respective papers. In addition to these, we have also compared our proposed models [M1-M7] with each other to show the effectiveness of incorporating different factors. Our baselines are listed in Table III.

TABLE III. BASELINE MODELS

	Baseline	Description
Simple Models	Basic matrix factorization [16]	Non-temporal non-taxonomy version of our M1 model
	timeSVD++ [10]	A temporal matrix factorization model
	A modified timeSVD++	A modified version of timeSVD++ which works on user-category matrix instead of user-item matrix
Deep learning models	DKN [6]	Deep Knowledge-Aware Network for News Recommendation, a content-based deep model for CTR prediction
	NCF [19]	Neural Collaborative Filtering, a deep learning recommender that generalizes the matrix factorization with multi-layer perceptron
	WD [20]	Wide & Deep Learning for Recommender Systems, a hybrid recommender that jointly trains linear models and deep neural networks
	RBM [21]	Restricted Boltzmann Machine, a generative neural network collaborative filtering model
	DeepFM [22]	Deep Factorization Machine, a collaborative filtering based neural network for CTR prediction with the power of factorization machines and deep learning

C. Hyperparameters and Evaluation metrics

We followed the conventional 70-10-20 proportions for splitting our dataset into training-validation-test sets and repeated the splitting process a few times and randomly. We have tuned our models using different hyperparameters on the validation set. All our models were trained using same set of parameters to have consistency. In that, we trained the models in a number of iterations (in the range of [10-50]) through the observed data by running the SGD solver. Our implementation of SGD uses an adaptive learning rate that has a constant multiplicative step size that varies by trying several options on a subset of the data. We defined the number of latent factors in the range of [5-50] and the regularization weights for both λ_1 and λ_2 in the range of [0. 000001 - 0.01]. We stopped training our models when there was no further improvement.

We have used two types of evaluation metrics based on our subtasks as shown in problem definition: to predict and to recommend. To evaluate our prediction task, we have used root mean squared error (RMSE) as our accuracy metric. To evaluate the recommendation quality, we have used precision @k (i.e., the proportion of top k results that are relevant) and recall @k (i.e., the proportion of all relevant results included in the top k results) as metrics. We have also used F1-score @k that returns the harmonic mean of precision and recall values. Here, we have considered four k values: 10, 20, 50, and 100.

V. EXPERIMENTAL RESULTS

Once we finished tuning the parameters using the validation set, we chose the best set of parameters and re-built our final model using the training and validation sets. We report our results using the test set as shown below.

A. Comparison of Time-Varying Factors

We have analyzed the effects of time-varying factors on users' preferences in our final model shown in equation (15), to find out which model gives us better accuracy. For the evaluation, we have empirically passed different temporal resolution as time-varying factors into our models [M1-M7], which were then tuned by varying weights on linear regularization term λ_1.

Our results for the RMSE values of these models are shown in Table IV.

TABLE IV. RMSE OF DIFFERENT TEMPORAL MODELS

	t_y	t_m	t_d	t_h	t_{min}	t_s	t
M1	0.8281	0.6554	0.7076	**0.6252**	0.6357	0.6481	0.6610
M2	0.7250	0.6733	0.6636	0.6635	0.6906	**0.6535**	0.6631
M3	0.9040	0.8482	0.6585	0.6301	0.6377	**0.6229**	0.6724
M4	0.7411	0.6626	0.6584	0.7303	**0.6404**	0.6621	0.7152
M5	0.5991	0.5987	0.5988	0.5989	0.5999	0.6001	0.5989
M6	0.6345	0.6128	0.6245	**0.6025**	0.6421	0.6222	0.6254
M7	0.6275	0.6570	**0.6020**	0.6139	0.6673	0.8091	0.6230

We have shown the best accuracy for each model in bold in Table IV. In M5, we did not see much difference among the different values of RMSE, so we did not highlight any one. The M1 model (simple matrix factorization) shows better accuracy when we pass hour as the contextual condition to the timestamp

(of rating). The item-dominant models M2 and M3 show better accuracy using the time unit second, whereas M4 shows better accuracy using minute. The category-dominant models M6 and M7 are close in terms of RMSE values and show better result when we pass hour and day respectively as the contextual conditions to the timestamp. In the M5 model, there is little difference in results but coarser-level time units: month, day, hour and the full timestamp (alone) give slightly better result. Overall, category-dominant models are better and M5 has the lowest RMSE among all the models.

As can be seen in Table IV, different time varying factors perform differently (or have different impacts) in each model [M1-M7]. Based on this result, for each model, to achieve best RMSE values, we should choose different time-varying factors. It also shows that previous models, only considering a single timestamp, without considering the effect of time units, may not achieve optimal performance.

We also evaluate the performance of our models with time-varying factors using F1 @ k as shown in Figure 1.

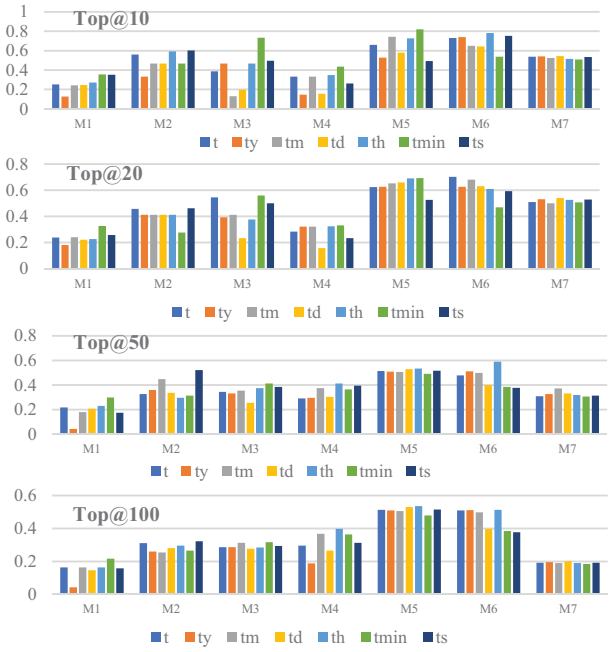

Fig. 1. F1-score varying @ k for our temporal models

Our results from Figure 1 can be summarized as follows. M1 shows its highest F1-score varying at all k values with minute as the time unit. The highest F1-score at all k values for M2 is seen when we consider second as the time unit. M3 shows the best F1-score at all k with minute as the time unit. The highest F1-score for M4 and M5 is seen when we pass minute as the time unit for k values 10, 20 and hour for k values 50, 100. In general, M6 performs the best with time unit as hour, and month achieves the best F1-score for k=20. M7 shows its highest F1-score varying at all k values with day as the time unit.

Overall, M5 and M6 give the best results in terms of precision-recall (skipped due to page limit), F1 and RMSE values. The general performance from category-dominant models is better than item-dominant models in all the metrics. Our results also reveal that item-dominant models work better when we include fine-level time units such as second and minute, and category-dominant models work better with coarser-level time units such as hour and day. The coarsest-level time units such as month or year in general do not perform well, although day as time unit sometimes achieves a high recall. Also, the difference between different time units is smaller for item-dominant models.

B. *Parameter Tuning*

We have used two regularization terms: λ_1 to control the effects of linear terms and λ_2 for adding weights to the interactions. For explanation, we demonstrate the results of our parameter tuning on one of our best models M5 (h). Since interactions are integral part of our experiment, we first tune λ_2 while keeping the default value of λ_1 (1e-10) as shown in Figure 2 (a). Once we find the best value of λ_2 i.e. 0.00001, we fix it and begin tuning the value of λ_1 for M5 as shown in Figure 2(b).

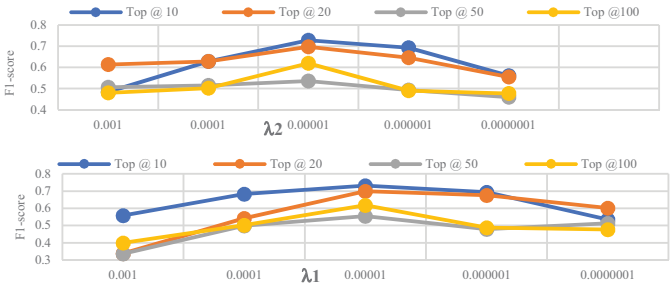

Fig. 2. (a). F1-score for varying λ_2 (b) F1-score for varying λ_1 on M5

We find the best values for both λ_1 and λ_2 as 0.00001 and therefore we attach them to our interaction terms and add them to the loss function in equation (16).

C. *Comparison of Our Proposed Models*

We first choose the most appropriate time-varying factor and the optimal setting for each model and then compare performance of all the proposed models. The major difference between them is the way how category information is considered in the bias term. Our results in the previous section reveal that item-dominant models [M2-M4] work better when we choose finer-level time unit such as second or minute. This is justifiable because in a news recommender system, users' preferences over news item tend to evolve fast. For the category-dominant models [M5-M7], coarser-level time unit such as hour or day tends to work better. This is also reasonable because users' preferences over news categories usually remain stable for some time and when there is a change, the change is not abrupt. For M1, we have chosen minute as the time unit since it achieves better results in terms of RMSE and precision-recall values. These time units are added as the contextual condition along with the main timestamp in each model.

We have plotted the average precision-recall pairs varying @ k for all the proposed models [M1-M7] using the most appropriate time unit as shown in Figure 3. As we can see in Figure 3, M2 model that considers taxonomy as the side information and associates both category and subcategory biases

with item, has the highest scores among the item-dominant models [M2-M4] for almost all k values. The only exception is a little higher recall value for M3 when k=20. M3 model that only considers top-level categories usually performs better than M4 model in which a flattened category/sub-category is considered. However, when k value is getting bigger, M4 starts to outperform M3, and when k=100, the performance of M4 is better than that of M3.

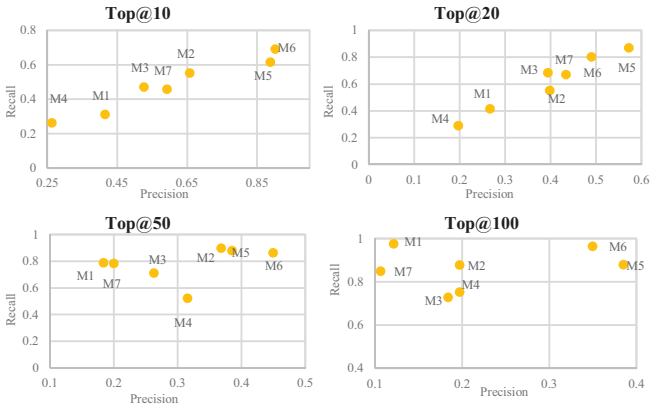

Fig. 3. Precision-Recall varying @k for our proposed models

M5 model that computes the temporal ratings of the users over the items with more complete taxonomy information (category with subcategory), and M6 model that considers only top-level categories, generally achieve a similar performance, which in many cases is much better than that of M7 model, especially on precision values. This conclusion is more or less consistent with the one on item-dominant models.

Overall M5 and M6 models which consider category as the main observation data give us the highest precision-recall ratio among all. The M2 model which considers item bias associated with category hierarchy also consistently shows the third highest precision-recall ratio for most k values. Again, it indicates that user's selection of news items is influenced by his preference on news categories.

Next, in Table V, we compare our models (temporal) with their non-temporal versions in terms of both RMSE values and F1-scores at various k values.

TABLE V. RMSE, F1OF OUR MODELS (TEMPORAL VS. NON-TEMPORAL)

	Model	RMSE	F1@10	F1@20	F1@50	F1@100
Non-Temporal	M1	0.7654	0.0970	0.1257	0.1605	0.1809
	M2	0.7800	0.1387	0.1413	0.1501	0.1609
	M3	0.7859	0.1395	0.1964	0.1404	0.1455
	M4	0.7795	0.1031	0.1067	0.1044	0.1161
	M5	0.7856	0.2233	0.2755	0.2882	0.2907
	M6	0.7811	0.1551	0.1697	0.1484	0.1661
	M7	0.7812	0.0386	0.0960	0.1737	0.1501
Temporal	M1 (h)	0.6357	0.3567	0.3254	0.2983	0.2163
	M2 (s)	0.6535	0.6007	0.4633	0.5225	0.3223
	M3 (s)	0.6229	0.4972	0.5006	0.3841	0.2939
	M4 (s)	0.6621	0.2632	0.2347	0.3937	0.3126
	M5 (h)	0.5989	0.7277	**0.6906**	0.5354	**0.5355**
	M6 (h)	**0.6025**	**0.7828**	0.6090	**0.5913**	0.5132
	M7 (h)	0.6139	0.5166	0.5270	0.3185	0.1894

For our temporal models, we configure them using the most appropriate time-varying factor (shown in Figure 1). The name of each temporal model is followed by the abbreviation of its best time unit, e.g., M2(s): M2 with second. As we can see clearly, all the temporal models perform better than their corresponding non-temporal versions. The category-dominant models are better than item-dominant models. Among all models, M5 (h) and M6 (h) perform the best.

D. Comparison with other Baselines

In this section, we compare our best-performing models (M5 and M6) with the baseline models. The original timeSVD++ works with user-item matrix. We modified it to include the taxonomy. We refer to original timeSVD++ as S2-I and modified timeSVD++ (user-category matrix) as S2-C. Table VI shows the results.

TABLE VI. RMSE AND F1-SCORES OF OUR MODELS AND BASELINES

	Model	RMSE	F1@10	F1@20	F1@50	F1@100
Our Models	M5 (h)	0.5989	0.7277	**0.6906**	0.5354	0.5355
	M6 (h)	0.6025	**0.7828**	0.6090	0.5913	0.5132
Baselines	S2-I	0.6901	0.3610	0.4058	0.4417	0.4610
	S2-C	0.6945	0.3659	0.4219	0.4412	0.4483
	NCF	0.5827	0.1273	0.1613	0.2047	0.2734
	DKN	**0.5195**	0.6842	0.6673	**0.7184**	**0.7268**
	RBM	0.8300	0.2652	0.1911	0.1040	0.1313
	WD	0.9516	0.0334	0.0390	0.0894	0.0888
	DeepFM	0.5781	0.5964	0.5422	0.5537	0.5456

DKN demonstrates the best accuracy in three metrics (RMSE, F1@50, F1@100), whereas our models beat DKN on F1@10 and F1@20 when k value is small. It indicates that our model has a good top N recommendation accuracy when N value is small. This actually aligns with the real scenarios in which users normally only have patience to check the first 10 or 20 recommended results. In all metrics, our models achieve similar results as the second best-performing baseline – DeepFM. We further compare our models with the baselines in Figure 4 on their precision-recall values.

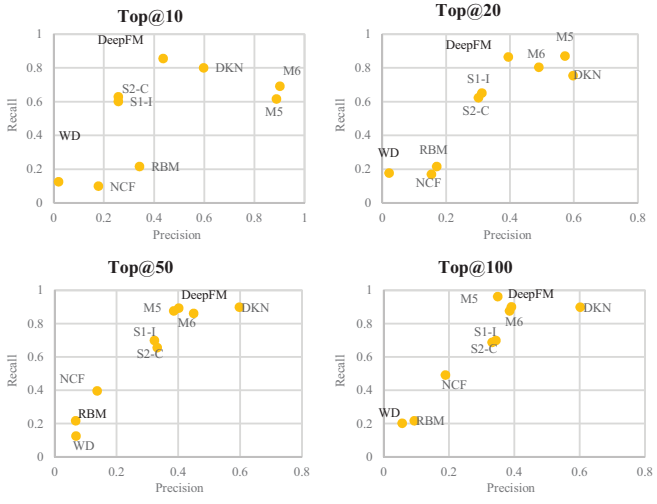

Fig. 4. Precision-Recall for M5, M6 and the baselines

The results show that our models consistently perform better than both variants of timeSVD++ and three state-of-the-art deep-learning models (RBM, NCF, WD). Among the deep-learning models, DKN and DeepFM show the best results. DKN demonstrates the best precision, recall values when k value is 50 and 100. DeepFM achieves the second-best results for these two k values. Its recall value is comparable to that of DKN, and the precision value is lower. For lower k values, our M5 and M6 models outperform DKN on precision when k value is 10 and recall when k value is 20 respectively, and as shown in Table VI, a higher overall F1 score for both of the k values.

Compared to these computationally complex neural recommenders, our model is simpler, with a comparable and sometimes better result. Also, we can observe an obvious difference in model training. It takes a significant amount of effort to configure and train the neural models. The training time is much longer. It usually takes a few minutes to train the model. As a comparison, our proposed models take a few seconds to build.

VI. Conclusion

In this paper, we have proposed a rich biased matrix factorization model with terms and hyperparameters exploiting the information from news taxonomy and temporal dynamics of news ratings. We have varied our models with different factors and extensively compared them with different traditional as well as computationally advanced deep-learning models. The results show that our temporal models perform better than traditional baselines. A few variants of our temporal models outperform some deep-learning models. To sum up, using our proposed model, we can not only satisfy users' reading interest but also address their preferences according to temporal dynamics. We can achieve good results using simpler but well-tuned techniques compared to complex models.

There are a few directions we would like to work on in the future. First, we may conduct a live experiment, asking users to give feedback on the recommendation results to further improve the model. Second, in our current work, when collecting data, we only considered users who have left some comments on different news articles. As a next step, we would like to include users who have left only one or two comments. We will explore ways to generate good recommendation results for them (also for users who may not leave any comment), possibly by imputing the existing sparse ratings. We also plan to incorporate detailed category hierarchy up to n (more than two) levels into the study. Furthermore, we would like to investigate our models with levels of hierarchy in linear complexity. Finally, we would like to build our model (the idea of considering temporal dynamics and news taxonomy) on top of the deep learning recommenders to see whether it can further improve the accuracy and whether it can outperform the matrix-factorization based models.

References

[1] L. Li, L. Zheng, F. Yang, and T. Li, "Modeling and broadening temporal user interest in personalized news recommendation," *Expert Systems with Applications*, 41(7): 3168–3177, Jun. 2014.

[2] A. Lommatzsch, B. Kille, and S. Albayrak, "Incorporating Context and Trends in News Recommender Systems," in *Proceedings of the International Conference on Web Intelligence*, 2017, pp. 1062–1068.

[3] F. Bentley, K. Quehl, J. Wirfs-Brock, and M. Bica, "Understanding Online News Behaviors," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, Paper No. 590.

[4] N. Koenigstein, G. Dror, and Y. Koren, "Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy," in *Proceedings of the 5th ACM Conference on Recommender Systems*, 2011, pp. 165-172.

[5] M.F. Dacrema, P. Cremonesi, and D. Jannach, "Are we really making much progress? A worrying analysis of recent neural recommendation approaches," *arXiv:1907.06902 [cs]*, 2019.

[6] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: Deep Knowledge-Aware Network for News Recommendation," in *Proceedings of the 2018 World Wide Web Conference, 2018,* pp. 1835-1844.

[7] L. Li, D. Wang, T. Li, D. Knox, and B. Padmanabhan, "SCENE: a scalable two-stage personalized news recommendation system," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval,* 2011, pp. 125-134.

[8] N. Jonnalagedda, S. Gauch, K. Labille, and S. Alfarhood, "Incorporating popularity in a personalized news recommender system," *PeerJ Comput. Sci.*, vol. 2, p. e63, Jun. 2016.

[9] G. Sottocornola, P. Symeonidis, and M. Zanker, "Session-based news recommendations," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 1395–1399.

[10] Y. Koren, "Collaborative filtering with temporal dynamics," in *in Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 447–456.

[11] Y. Ding and X. Li, "Time weight collaborative filtering," in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, 2005, p. 485-492.

[12] L. Xiong, X. Chen, T. Huang, J. Schneider, and J.G. Carbonell, "Temporal collaborative filtering with Bayesian Probabilistic Tensor Factorization," in *Proceedings of the 2010 SIAM International Conference on Data Mining*, 2010, pp. 211–222.

[13] L. Xiang, Q. Yuan, S. Zhao, L. Chen, X. Zhang, Q. Yang, and J. Sun, "Temporal recommendation on graphs via long- and short-term preference fusion," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010, p. 723-732.

[14] Q. Yuan, G. Cong, and A. Sun, "Graph-based Point-of-interest Recommendation with geographical and temporal influences," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, 2014, pp. 659–668.

[15] Z. Sun, G. Guo, and J. Zhang, "Learning hierarchical category influence on both users and items for effective recommendation," in *Proceedings of the Symposium on Applied Computing*, 2017, pp. 1679–1684.

[16] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer*, 42(8): 30–37, Aug. 2009.

[17] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of statistical software*, 33(1):1, 2010.

[18] C.J. Hutto and E. Gilbert, "VADER: a parsimonious rule-based model for sentiment analysis of social media text," in *Proceedings of the 8th International AAAI Conference on Weblogs and Social Media*, 2014, p. 10.

[19] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 173–182.

[20] H.-T. Cheng *et al.*, "Wide & Deep Learning for Recommender Systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 2016, pp. 7-10.

[21] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted Boltzmann machines for collaborative filtering," in *Proceedings of the 24th International Conference on Machine Learning*, 2007, pp. 791–798.

[22] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A Factorization-Machine based Neural Network for CTR Prediction," *arXiv:1703.04247 [cs]*, Mar. 2017.