# Leveraging Moderate User Data for News Recommendation

Dhruv Khattar, Vaibhav Kumar*, Vasudeva Varma
*Information Retrieval and Extraction Laboratory*
*International Institute of Information Technology Hyderabad*
Hyderabad - 500032, Telangana, India
Email: {dhruv.khattar, vaibhav.kumar}@research.iiit.ac.in, vv@iiit.ac.in

***Abstract*—It is very crucial for news aggregator websites which are recent in the market to actively engage its existing users. A recommendation system would help to tackle such a problem. However, due to the lack of sufficient amount of data, most of the state-of-the-art methods perform poorly in terms of recommending relevant news items to the users. In this paper, we propose a novel approach for Item-based Collaborative filtering for recommending news items using Markov Decision Process (MDP). Due to the sequential nature of news reading, we choose MDP to model our recommendation system as it is based on a sequence optimization paradigm. Further, we also incorporate factors like article freshness and similarity into our system by extrinsically modelling it in terms of reward for the MDP. We compare it with various other state-of-the-art methods. On a moderately low amount of data we see that our MDP-based approach outperforms the other approaches. One of the reasons for this is that the baselines fail to identify the underlying patterns within the sequence in which the articles are read by the users. Hence, the baselines are not able to generalize well.**

***Index Terms*—Collaborative Filtering, Markov Decision Process, News Recommendation, Semantic Similarity**

## I. Introduction

In the contemporary world, readers want to read news coming from a variety of sources. One of the reasons for this is that a single digest is not able to cover the different kinds of news that a user looks for. Also, reading news from a variety of sources provides alternate perspectives to a particular news event. Altogether, there are a variety of reasons to it. This is the scenario where news aggregators come into play. Such aggregators collect news from a variety of sources to present to its users. User engagement needs to be maximized and in order to do this articles of interest should be presented to the users. Specifically, this is very crucial for the websites which are recent to the market. In the initial stages such websites have very moderate user activity and one of their prime focus is not to lose their existing userbase. Hence, actively engaging its existing users becomes crucial. A recommender system would help in solving such a problem. However, lack of sufficient amount of data to generate good recommendations remains a problem.

Most commonly, methods for recommending news articles have been divided into two categories: Content-based filtering(CBF) and Collaborative filtering(CF). The latter is a major approach to this task [1] [2] [3] [4] [5]. Content-based recommendation [6] [7] uses features about items and/or users to recommend news items. Both have proven to work well in different scenarios but have their own challenges.

Typically, CF requires a considerable amount of history of interactions of the users before good quality recommendations can be provided. This is known as the cold start problem [8]. On the other-hand CBF uses different kinds of similarity measures between items and/or users to recommend news articles. However, none of these methods directly account for the freshness of the news articles.

Apart from this, one of the major issues is that of datasize required for learning the model. As mentioned earlier, typical CF requires a considerable amount of user history in order to come up with good recommendations. In recent years, the problem of recommendation has also been tackled using deep learning settings. However, usually the data on which such models train is very large and do not seem to work well on moderately low amount of data.

The sequence in which a user reads news articles has a lot of information present in it. Due to the sequential nature of news reading, the task of recommendation could be posed as a sequential prediction problem as well as a sequential decision problem. The latter method of modelling is similar to what we present here. In this paper, we propose a model which is similar to Item-based CF which leverages the information present in the sequence in which the articles are read by the users. Along with it, we also incorporate the aspect of freshness and similarity between different articles to capture the overall interests of users. The similarity between different articles is captured using a semantic measure. We use MDP to model this altogether.

## II. Related Work

There has been a lot of work on recommendation systems. Here, we look at some of the work which is related to our proposed approach.

In general, recommendation system can be divided into CF-based and CBF-based approaches. In CF-based recommendations, items are recommended to a user based on users sharing similar interests. CF-based systems can further be divided into

*Corresponding Author

Item-based, User-based or a hybrid of both these. Examples of this technique include nearest neighbour modelling [1], Bayesian Matrix Factorization, Restricted Boltzmann Machine(RBM) [4], AutoRec(autoencoder based collaborative filtering) [10].

Recently, researchers have developed deep learning based approaches to tackle the problem of recommendation. In [12], authors use Boltzmann Machines to learn similarity between items, and then combined this with Collaborative Filtering. In [10] [19], authors use Autoencoders for Collaborative Filtering. It has been shown that it outperforms the state-of-the-art CF techniques like Matrix Factorisation, RBM-CF and is comparable with LLORMA [13].

## III. Model Architecture

In this section, we describe the overall architecture of our model. Firstly, we mention the notations used. Next, we discuss the semantic measure used to find out the similarity between different news articles. Then, we discuss the construction of Markov Decision Process. Finally, we explain how we recommend articles to a user based on the decisions made by the MDP.

### A. Notation

We denote the set of articles by $N$. We treat each article as a state. We define the reading sequence of each user $u_i$ as $U_i$ = [$n_1$, $n_2$, ... , $n_t$]. We introduce a parameter $k$, which denotes the number of clicks after which an article $n_j$ is read after $n_i$ in a given sequence. For example: in the above mentioned sequence $n_2$ is read 1 click after $n_1$.

### B. Semantic Similarity

We use a method similar to that of Semantic Recommendation mentioned in [14] for finding out similarity between different news articles. In traditional forms of text comparison all words in the text are considered. In our results we see that using traditional forms of text comparison like cosine similarity (in KNN) does not help much. One of the reasons for this is that it becomes difficult for such methods to identify the relatedness between two different words. For example, there is no way to explicitly identify the relatedness between Roger Federer and Rafael Nadal using traditional cosine. Users who are interested in the former, will be interested in reading news about the latter as well. To overcome this we create an ontology which is based on the concepts (topics) which best describe the given article.

For each news article in our data, we have at least 1 and at most 5 topics that best describe that article. We call these topics as concepts and using these concepts we find the semantic similarity between two news articles. Suppose, we have two news articles $n_i$ and $n_j$. We define a concept set of a news article $n_i$ as all the topics that describe that article (given in the data). We refer to this concept set by $C(n_i)$. Therefore, the concept set of $n_i$ is:

$$C(n_i) = [c_{i1}, c_{i2}, c_{i3}, c_{i4}, c_{i5}]$$

Now, we define concept equivalence set of a concept $c$ as the set containing all the concepts which occurred at least once with $c$ across different news articles. Let us denote the concept equivalence set of $c$ by $CE(c)$. Therefore,

$$CE(c) = \cup_{n_i \in N_c} C(n_i)$$

where, $N_c$ is the set of all news articles which have $c$ as a concept. Now, we use Jaccard's similarity based on the concepts contained in the two news articles. Therefore,

$$ssim(n_i, n_j) = \frac{(\cup_l CE(c_{il})) \cap (\cup_l CE(c_{jl}))}{(\cup_l CE(c_{il})) \cup (\cup_l CE(c_{jl}))} \quad (1)$$

where, $c_{il}$ is the $l$th concept of article $i$.

One of the advantages of using such a measure is that, even for the users whose interaction with the website is very less, we can still come up with a plethora of related topics which might be of interest to them.

### C. Markov Decision Process

In [15], it has been argued that it is better to view the problem of recommendation as that of sequential optimization problem, and hence MDP is better suited for it. An MDP is by definition a four tuple: $\langle S, A, Rwd, TP \rangle$, where S is the set of states, A is the set of actions, $Rwd$ is the reward function, and $TP$ is the transition probability from one state to the other. The decision makers goal in MDP is to maximize its reward stream.

There are two problems that we come across while using an MDP for news recommendation. Firstly, the state space is too large because of the vast number of news articles. Typically, an MDP solver requires a matrix of size AxNxN, where N is the number of news articles. Hence, formulating the set of actions becomes crucial both for scalability as well as accuracy. In [15], the size of A becomes equal to N. We change this by introducing our own set of actions. This is discussed later in this section.

We treat each read news article as a state. The transition probability is denoted by $TP(n_i, n_j)$, where $n_i$,$n_j \in N$. We use $U_i$ of each user along with an exponential discounting function to calculate $TP(n_i, n_j)$ as follows:

$$TP(n_i, n_j) = \frac{count(n_i, n_j)}{\sum_{n \in N} count(n_i, n)} \quad (2)$$

$$count(n_i, n_j) = \sum_{k=0}^{K} trans_k(n_i, n_j) * e^{-(k-1)} \quad (3)$$

where, $trans_k(n_i, n_j)$ denotes the number of times exactly $k-1$ articles were read from $n_i$ to $n_j$.

We use exponential discounting with the assumption that the current article that is being read by the user will have partial effects on the type of articles that the user will read in the near future. Also, this helps us to tackle the problem of varying interests of user.

Now, given the reading sequence $U_i$ of each user, we further define action $A_i \in A$. $A_i$ denotes the action in which $i$ number of clicks were required for transitioning between two states.

For example: suppose a user read articles in the following sequence $n_1$, $n_2$, $n_3$, $n_4$. Then, an action $A_2$ over the state $n_2$ would lead us to the state $n_4$. Our action set consists of five actions ranging from $A_1$ to $A_5$. We then model our reward function.

As mentioned earlier, the lifespan of a news article is less. Hence, a user would have more incentive in reading an article which is published later in time than the one which is currently being read. Keeping this in mind, we define our first reward function as:

$$Rwd_1(A_i, n_j, n_k) = \begin{cases} \frac{dt(n_k) - dt(n_j)}{N} & trans_i(n_j, n_k) > 0 \\ 0 & otherwise \end{cases} \tag{4}$$

where $dt(n_i)$ denotes the discovery time of news article $n_i$. Secondly, to capture the similarity between two news article, we use the semantic similarity measure as follows:

$$Rwd_2(A_i, n_j, n_k) = \begin{cases} ssim(n_i, n_j) & trans_i(n_j, n_k) > 0 \\ 0 & otherwise \end{cases} \tag{5}$$

where $ssim(n_i, n_j)$ denotes the semantic similarity measure mentioned in 3.2. We then define a third reward function which is a weighted combination of the above two:

$$\begin{aligned} Rwd_3(A_i, n_j, n_k) = \alpha * Rwd_1(A_i, n_j, n_k) + \\ (1 - \alpha) * Rwd_2(A_i, n_j, n_k) \end{aligned} \tag{6}$$

where $\alpha$ is a hyperparameter. It is tuned using a cross validation set. We use the Policy Iteration method using the MDP-toolbox to calculate the optimal policy [17].

### D. Recommending Articles

The MDP gives us the information about the best action to be undertaken at a given state. To recommend articles we look at the reading history of each user. We denote the amount of history to be considered for recommendation as $RH$. For example if the reading sequence of a user was [$n_1$, $n_2$, $n_3$, $n_4$, $n_5$], and our chosen $RH$ is 2, then we only use the decisions given by the MDP for states $n_4$ and $n_5$. Here each decision is an action. An action leads to another set of states. We then choose the top 5 states which have maximum probabilities of being transitioned to from the current $RH$ state. For example, in the above sequence, suppose the decision on $n_4$ is $A_1$ and that of $n_3$ is $A_2$. We would then consider all those states which could be reached from $n_4$ by performing the action $A_1$. Similar would be the case for $n_3$ when executing an action $A_2$ over it. It could be possible that both $n_3$ and $n_4$ lead to the same state, say $n_s$. In such cases we associate the probability of $n_s$ with the state for which $TP(n_i, n_s)$ is maximum. Finally, we get a list of states with decreasing order of probability values. We select the top 5 states from this list as our set of recommended articles. This list is basically a ranked set of recommended articles.

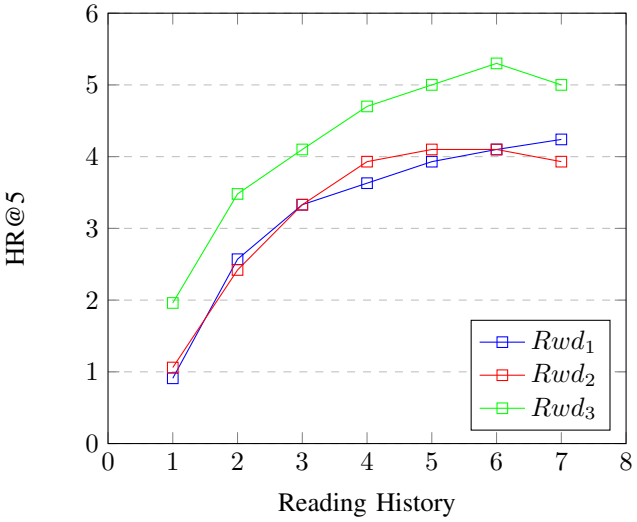


Fig. 1: Performance of our model on Validation data

| Method | HR@10 | NDCG@10 | P@5 | MAP |
|---|---|---|---|---|
| Most Popular | 1.16 | 0.33 | 0 | 0.1 |
| RegSVD | 1.83 | 0.833 | 0 | 0.16 |
| KNN | 1.83 | 1.16 | 0 | 0.23 |
| Bigram | 3.16 | 1.33 | 0.16 | 0.53 |
| Discounted Bigram | 5.33 | 1.83 | 0.5 | 0.86 |
| MDP($Rwd_1, RH = 6$) | 6.83 | 3.5 | 0.5 | 1.23 |
| MDP($Rwd_2, RH = 6$) | 7.16 | 3.66 | 0.5 | 1.26 |
| MDP($Rwd_3, rh = 6$) | 7.66 | 4.16 | 0.83 | 1.63 |

TABLE I: Performance of our model vs state-of-the-art models

## IV. Data and Experiments

For the purpose of our study we received data from a social network aggregation website called Veooz.com[1]. The data contains news articles read by users in a sequential manner. This was collected across a period of three months. The news articles were in English. Each news article came tagged with at least one and at most five topics. The way in which these topics are found out is proprietary to the website. We removed all the users who had read less than 6 articles. Finally, the data contained 660 users, 1826 unique articles. We randomly select data of 60 users as our cross validation set for learning the hyper parameters involved in MDP.

We use two different methods to test our model. Firstly, we remove the last 5 articles read by the users. The removed articles act as our testing set. We train our model on the remaining articles read by the user. We then recommend a ranked set of 5 articles to the user. We use two popular metrics for evaluating our system: precision at position 5 (P@5) and Mean Average Precision (MAP).

In the other setting, we exclude the last article read by the user and use the rest for training. We then recommend a ranked set of $x$ articles. The performance of the ranked list is judged by Hit Ratio (HR) and Normalized Discounted Cumulative gain (NDCG). As such, HR@$x$ intuitively measures whether the test item is present in the top-$x$ list, and the NDCG

[1]https://www.veooz.com

accounts for the position of the hit by assigning higher scores to hits at top ranks. We fine tune our hyperparameters using the cross validation set and thus, set $RH = 6$, $\alpha = 0.7$ and $K = 5$.

In order to make our results more convincing, we compare it with several state-of-the-art-methods. These include KNN as mentioned in [5] which uses similarity between the articles already read by the user and a new article to recommend news articles. RegSVD [20] is a matrix factorization based approach for recommending news articles. We then incorporate a bigram based model as mentioned in [16]. We modify the bigram based model to use our discounted probabilities as described in equations (2) and (3). We then finally evaluate our model with the three different reward functions mentioned. For some of the baselines we use the LibRec implementation [18]. Others are implemented in python.

## V. Results and Discussion

Table 1 summarizes the results. By looking at the performance of the most-popular baseline, we can say that popularity does not necessarily correlate with the users interests. Secondly, we see that both KNN based and MF-based methods perform poorly specially in the case of their P@5 accuracy. KNN uses the cosine similarity metric and hence is unable to generalize well because users have very less reading history. It isn't able to capture the actual interests of users. One of the reasons for the MF-based methods to perform poorly is their inability to identify the important latent factors. With such a kind of data, where there is no severe user activity, MF-based methods seem to fail to identify latent factors which are crucial to the process of recommendation.

Next, we see that the simple bigram and discounted bigram models perform better than the above discussed. One of the most important reasons for this is that, it explicitly takes into account the reading sequence of a user. These methods are similar to a language model and make use of the sequence to recommend a news article. Recommendation is posed as a sequence prediction problem in these cases.

We see that our MDP-based model outperforms the baselines. When modelled using $Rwd_3$, the MDP performs best. A combination of the information present in the reading sequence, along with the freshness and semantic similarity between the different read articles is fairly able to capture the users interests.

## VI. Conclusion

News aggregator websites, which are new to the market cannot generate a lot of data from user interaction. Most of the state-of-the-art methods heavily depend on huge amounts of data. Such websites would at least want their existing userbase to remain intact and in order to do so would deploy a recommender system. However, by the experiments conducted we can see that the baselines do not perform very well in such a scenario (with less data). Here we show that by using the sequential nature of news reading combined with other aspects like semantic similarity and freshness, we are fairly able to generalize and provide better recommendations. Better results show that the semantic similarity measure is able to capture the diverse interests of users. Also, posing the problem of recommendation as that of sequential optimisation provides us with better results. Another advantange of such a system is that if we solely use $Rwd_1$ to model our MDP, then the entire recommendation system becomes language agnostic. Hence, for news aggregators which combine news belonging to a variety of languages, this could be helpful.

## References


[1] Bell, Robert M., and Yehuda Koren. "Improved neighborhood-based collaborative filtering." KDD cup and workshop at the 13th ACM SIGKDD international conference on knowledge discovery and data mining. sn, 2007.

[2] Rennie, Jasson DM, and Nathan Srebro. "Fast maximum margin matrix factorization for collaborative prediction." Proceedings of the 22nd international conference on Machine learning. ACM, 2005.

[3] Salakhutdinov, Ruslan, and Andriy Mnih. "Bayesian probabilistic matrix factorization using Markov chain Monte Carlo." Proceedings of the 25th international conference on Machine learning. ACM, 2008.

[4] Salakhutdinov, Ruslan, Andriy Mnih, and Geoffrey Hinton. "Restricted Boltzmann machines for collaborative filtering." Proceedings of the 24th international conference on Machine learning. ACM, 2007.

[5] Sarwar, Badrul, et al. "Item-based collaborative filtering recommendation algorithms." Proceedings of the 10th international conference on World Wide Web. ACM, 2001.

[6] Linden, Greg, Brent Smith, and Jeremy York. "Amazon. com recommendations: Item-to-item collaborative filtering." IEEE Internet computing 7.1 (2003): 76-80.

[7] Liu, Jiahui, Peter Dolan, and Elin Rnby Pedersen. "Personalized news recommendation based on click behavior." Proceedings of the 15th international conference on Intelligent user interfaces. ACM, 2010.

[8] Su, Xiaoyuan, and Taghi M. Khoshgoftaar. "A survey of collaborative filtering techniques." Advances in artificial intelligence 2009 (2009): 4.

[9] Wang, Chong, and David M. Blei. "Collaborative topic modeling for recommending scientific articles." Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2011.

[10] Sedhain, Suvash, et al. "Autorec: Autoencoders meet collaborative filtering." Proceedings of the 24th International Conference on World Wide Web. ACM, 2015.

[11] Li, Ping, Trevor J. Hastie, and Kenneth W. Church. "Very sparse random projections." Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2006.

[12] Gunawardana, Asela, and Christopher Meek. "Tied boltzmann machines for cold start recommendations." Proceedings of the 2008 ACM conference on Recommender systems. ACM, 2008.

[13] Lee, Joonseok, et al. "Local low-rank matrix approximation." International Conference on Machine Learning. 2013.

[14] IJntema, Wouter, et al. "Ontology-based news recommendation." Proceedings of the 2010 EDBT/ICDT Workshops. ACM, 2010.

[15] Shani, Guy, David Heckerman, and Ronen I. Brafman. "An MDP-based recommender system." Journal of Machine Learning Research 6.Sep (2005): 1265-1295.

[16] Garcin, Florent, et al. "Personalized news recommendation based on collaborative filtering." Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01. IEEE Computer Society, 2012.

[17] Chads, Iadine, et al. "MDPtoolbox: a multiplatform toolbox to solve stochastic dynamic programming problems." Ecography 37.9 (2014): 916-920.

[18] Guo, Guibing, et al. "LibRec: A Java Library for Recommender Systems." UMAP Workshops. 2015.

[19] Strub, Florian, Jrmie Mary, and Romaric Gaudel. "Hybrid Collaborative Filtering with Autoencoders." arXiv preprint arXiv:1603.00806 (2016).

[20] Paterek, Arkadiusz. "Improving regularized singular value decomposition for collaborative filtering." Proceedings of KDD cup and workshop. Vol. 2007. 2007.