HRAM : A Hybrid Recurrent Attention Machine for News Recommendation

Dhruv Khattar, Vaibhav Kumar*
International Institute of Information Technology
Hyderabad, India
{dhruv.khattar,vaibhav.kumar}@research.iiit.ac.in

Vasudeva Varma, Manish Gupta†
International Institute of Information Technology
Hyderabad, India
{vv,manish.gupta}@iiit.ac.in

Abstract

Popular methods for news recommendation which are based on collaborative filtering and content-based filtering have multiple drawbacks. The former method does not account for the sequential nature of news reading and suffers from the problem of cold-start, while the latter, suffers from over-specialization. In order to address these issues for news recommendation we propose a Hybrid Recurrent Attention Machine (HRAM). HRAM consists of two components. The first component utilizes a neural network for matrix factorization. While in the second component, we first learn the distributed representation of each news article. We then use the historical data of the user in a sequential manner and feed it to an attention-based recurrent layer. Finally, we concatenate the outputs from both these components and use further hidden layers in order to make predictions. In this way, we harness the information present in the user reading history and boost it with the information available through collaborative filtering for providing better news recommendations. Extensive experiments over two real-world datasets show that the proposed model provides significant improvement over the state-of-the-art.

Keywords

Matrix Factorization, Recurrent Neural Networks, News Recommendation

ACM Reference Format:
Dhruv Khattar, Vaibhav Kumar and Vasudeva Varma, Manish Gupta. 2018. HRAM : A Hybrid Recurrent Attention Machine for News Recommendation. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3269206.3269311

1 Introduction

The web provides instant access to a wide variety of online news. Hence, it becomes desirable for news aggregators to have a recommendation system that would point a user to the most relevant items and thus would maximize the user engagement with the site and minimize the time for finding relevant content.

A popular approach to the task of recommendation is collaborative filtering (CF) [2, 14, 16] which uses the user's past interaction with the item to predict the most relevant content. Amongst the various approaches for collaborative filtering, matrix factorization [10], is the most popular one, which projects users and items into a shared latent space and uses the inner product of the latent vectors to model interaction between users and items. Another common approach is content-based recommendation, which uses features between items and/or users to recommend new items to the users based on the similarity between features.

However, both the above mentioned methods have multiple drawbacks. Methods based on collaborative filtering do not harness the information present in the sequence in which the articles were read by the user. The sequence can help us analyze the overall interests as well as the changing interests of the users. Apart from this, collaborative filtering models also suffer from the problem of cold start. On the other hand, content-based models suffer from the problem of over specialization.

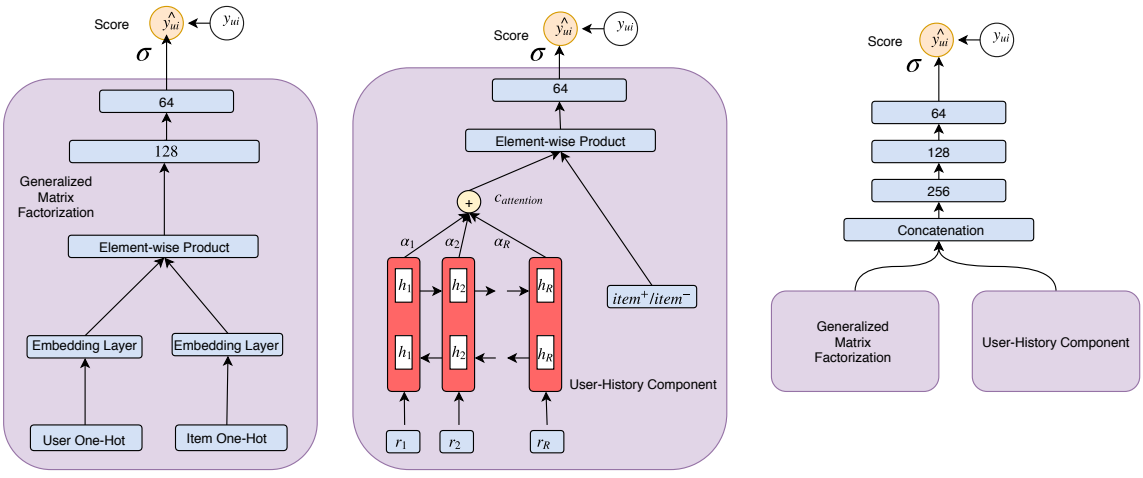
A news recommendation system should capture the overall interests of the user as well as their changing interests. It should also handle item cold start to deal with the overwhelming amount of fresh articles published each day. We propose the Hybrid Recurrent Attention Machine (HRAM) to tackle these problems.

As illustrated in Fig. 1, HRAM consists of two components. The first component is similar to Generalized Matrix Factorization as presented in [6]. We use one-hot encodings for both users and items as inputs to the component. This helps us to discover the underlying user-item interaction patterns. For the second component (User-History Component), we first learn a distributed representation for each news article using doc2vec [11] by combining the title and text of each news article. We then use the historical data of the user in a sequential manner and feed it to an attention based recurrent layer. The part dealing with attention allows the model to attend to articles in a differential manner, discriminating the more from the less important ones and thus enabling the model to adapt to the changing user preferences. Using a distributed representation for each article allows us to tackle the problem of item-cold start as well. We then fuse the two components together by concatenating their outputs followed by further hidden layers. Lastly, we use a logistic unit in order to compute the relevance of an article given a user. We pose the problem of recommending articles as that of binary classification in order to train our model parameters over implicit feedback data. We then perform extensive experiments over two real-world datasets to show the effectiveness of our model.

*Author had equal contribution. He can also be contacted at vaibhav2@andrew.cmu.edu
†Author is also a Principal Applied Researcher at Microsoft.

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3269311



(A) Recommendation using Generalized Matrix Factorization

(B) Recommendation using User-History Component

(C) Hybrid Recommendation Attention Machine (HRAM)

Figure 1: Model Architecture

2 Related Work

There has been extensive study on recommendation systems with a myriad of publications.

Traditional Recommendation Systems

Recommendation systems in general can be divided into collaborative filtering based systems and content based systems. In collaborative filtering, an item is recommended to a user if similar users liked that item. Examples of such techniques include Bayesian matrix factorization [15], matrix completion [14], Restricted Boltzmann Machines (RBMs) [16], nearest neighbor modeling [2] etc. Another common approach for recommendation is content-based recommendation. In this approach, features from user's profile and/or items are extracted and are used for recommending items to users.

Neural Network-based Recommendation Systems

There has been some work on exploring neural networks for recommendation systems. In [16], a two-layer RBM was used to model users' explicit ratings on items. Recently, auto-encoders have become a popular choice for building recommendation systems [3, 18, 19]. AutoRec [18] learns hidden structures that can reconstruct a user's ratings given her historical ratings as inputs. In terms of user personalization, this approach is similar to the item-item models [12, 17] that represent a user using features of her rated items. While previous work has focused on modeling CF, they have modeled the *observed* ratings data only. As a result, they can easily fail to learn users preference from the positive-only implicit data. Deep Semantic Structured Model (DSSM) [9] which was originally used for ranking web documents has been extended to recommendation scenarios in [4], where the first neural network contains user's query history and the second neural network contains implicit feedback on items. The resulting model is named multi-view DNN. However, it is not directly adaptable for news recommendation because it requires a lot of information outside the news domain. In [21] a collaborative denoising auto-encoder (CDAE) for CF with implicit feedback is presented. In contrast to the DAE-based CF [19], CDAE additionally plugs a user node to the input of auto-encoders for reconstructing the user's ratings. While CDAE is solely based on item-item interaction, our proposed model is based on user-item interactions. The Neural Collaborative Filtering (NCF) model [6] replaces the user-item inner product with a multi-layer perceptron architecture that can learn an arbitrary function from the given data which can then be used for generating recommendations.

3 Model Architecture

In this section, we present our Hybrid Recurrent Attention Machine. We first explain the two components individually followed by the fusion of both these components which is our proposed model. Finally, we discuss how we learn the parameters of the model.

3.1 Generalized Matrix Factorization (GMF)

This is similar to what has been used in [6] for Matrix Factorization. As can be seen from Fig. 1(A), we use the one-hot encodings of users and items as inputs to the network. This is then followed by an embedding layer which helps us obtain a vector for each user and item. Let us denote the user vector by p_u^{mf} and the item vector by q_i^{mf}. We first compute an element-wise product between these two vectors, followed by multiple fully connected layers, and then use that as input to an activation function which gives us the score for an article. This can be written as,

$$\hat{y_{ui}} = a_{out}(\Phi_1(p_u^{mf} \odot q_i^{mf})) \quad (1)$$

where a_{out} and Φ represent the activation function (logistic function) and non-linear transformation caused by multiple fully connected layers respectively. Note that if we use an identity function for a_{out} and Φ, we will be able to recover the Matrix Factorization model. Using such a model helps us to retain the advantages of collaborative filtering associated with news recommendation.

3.2 User-History Component

Distributed Representation of News Articles: To understand the interests of each user, we first need to understand the information present in the content of the news articles. For this purpose,

we learn a 300-dimension distributed representation [11] for each news article by combining the title and text of the news articles. Learning such a representation allows us to (1) capture the overall semantics of the news article, and (2) enables us to come up with a representation for new news articles as well as of articles with varying lengths.

Attention-based Recurrent Network: The overview of the component can be seen from Figure 1(B). One of the major challenges in news recommendation is that of understanding and adapting to the changing interests of the user and start recommending articles as soon as they are published.

The sequence in which the articles are read by a particular user gives us a lot of information about her interests. We use the learned representations of the articles in the user history as inputs to Bidirectional LSTMs. LSTMs have been shown to be capable of effectively capturing long-range dependencies in text [8, 20]. Bidirectional LSTMs can capture both past and future information effectively. Bidirectional LSTMs along with attention [1] allow the model to attend to articles in a differential manner, discriminating the important ones from the less important ones and thereby providing the model an overall summary of the user interests. In order to train the model, we first choose a window of size R (context size/user history). We slide this window across the historical reading data of each user. Articles present in each window are used as inputs to the bidirectional LSTM. Restricting the size of the window allows the model to adapt to the changing interests of the user, as predictions are only made considering her latest few interactions.

Finally, we compute an element-wise product between the output of the attention part and the distributed representation of the candidate article. We use the logistic function to compute the final output. Overall we can write the score assigned to each article as,

$$\hat{y_{ui}} = a_{out}(\Phi_2(p_u^{attention} \odot q_i^{content})) \tag{2}$$

where, $p_u^{attention}$ represents the output after the attention layer for each user, $q_i^{content}$ represents the learned embedding for an article, $\odot$ stands for the element-wise product, Φ represents a non-linear transformation caused by the hidden layer and a_{out} stands for the activation function (logistic function) at the output layer. With this approach we are able to utilize the content of the news articles, handle changes in users' interests as well as address item cold start.

3.3 HRAM

Finally, we fuse the GMF and User-History Component and call it the Hybrid Recurrent Attention Machine (HRAM). This can be seen in Fig. 1 (C). We concatenate the output of the GMF component with the output of the User-History Component. This is followed by hidden layers of sizes 256, 128, 32 respectively. We finally use the logistic function as the activation function to make predictions. Overall this can be represented as follows.

$$\Phi^{GMF} = \Phi_1(p_u^{mf} \odot q_i^{mf}), \tag{3}$$

$$\Phi^{Historical} = \Phi_2(p_u^{attention} \odot q_i^{content}) \tag{4}$$

$$\hat{y_{ui}} = a_{out}(\Psi([\Phi^{GMF}, \Phi^{Historical}])) \tag{5}$$

where $[\Phi^{GMF}, \Phi^{Historical}]$ represents the concatenation of the two vectors, Ψ represents the non-linear transformation caused by multiple fully connected hidden layers after concatenation. The final score for an article i given a user u is represented by $\hat{y_{ui}}$. From Equations 3-5, we can see how we use the information based on collaborative filtering, as well as the content based historical user preferences in order to come up with recommendations.

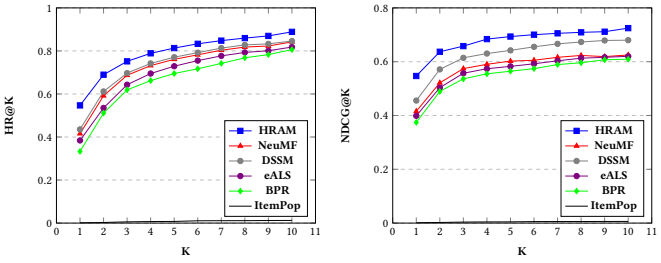


Figure 2: HR and NDCG over Malayalam Dataset

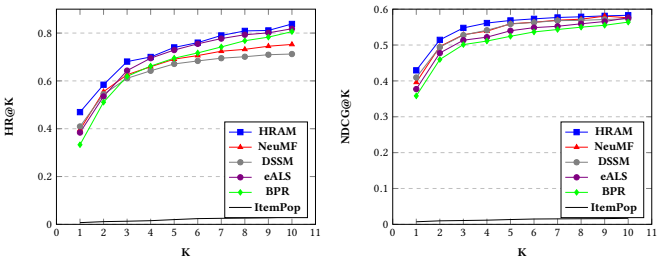


Figure 3: HR and NDCG over Indonesian Dataset

3.4 Parameter Learning

Since, we have data based on implicit feedback we pose the problem of recommendation as that of binary classification. As mentioned earlier, we use the logistic function at the output layer in order to constrain the values between 0 and 1. We use binary cross-entropy as our loss function which can be minimized using Stochastic Gradient Descent.

4 Experiments

We conduct experiments to answer the following research questions

- How does the proposed model compare with the state-of-the-art methods?
- How does the proposed model work when we vary the number of negative samples for learning the parameters?
- How does our model perform for the item cold start cases?

We make the code publicly available[1].

Dataset We use two datasets provided by a popular news aggregation website, NewsPlus[2]. The first dataset contains a list of articles (along with the content) read by 10297 users in an Indian language, Malayalam. The second dataset contains a list of articles (along with content) read by 22848 users in Indonesian. Due to proprietary issues, the dataset cannot be released publicly.

Evaluation Protocol To evaluate the performance of the recommended item, we use the leave-one-out evaluation strategy which has been widely adopted in literature [7, 13]. For each user we held-out her latest interaction as the test instance and utilized the

[1] https://github.com/dhruvkhattar/HRAM
[2] https://www.veooz.com

remaining data for training. Since it is time consuming to rank all items for every user during evaluation, we followed the popular strategy [4, 10] that randomly samples 100 articles that the user has not read, and ranking the test article among the 100 articles. The performance of the ranked list is judged using two metrics: Hit Ratio (HR) and Normalized Discounted Cumulative gain (NDCG) [5].

Baselines

- **ItemPop** [13]: It recommends most popular items.
- **BPR** [13]: This method optimizes the matrix factorization method with a pairwise ranking loss, which is tailored to learn from implicit feedback.
- **eALS** [7]: This is a state-of-the-art matrix factorization method for item recommendation. It optimizes the squared loss (between actual item ratings and predicted ratings) and treats all unobserved interactions as negative instances, weighting them non-uniformly by item popularity.
- **DSSM** [9]: It is a Deep Neural Network optimized for ranking. We adapt it for our task by passing on the article representation as the input.
- **NeuMF** [6]: It is a state-of-the-art neural matrix factorization model. It also models recommendation generation using implicit feedback as a binary classification problem.

Parameter Settings We randomly divide labeled data into training and validation in a 4:1 ratio. We randomly initialize the network. We use a batch size of 256 and use AdaDelta as the optimizer.

Performance Comparison Fig. 2 shows the performance of various recommendation systems where the ranking position K ranges from 1 to 10. Our model shows consistent improvements over the other methods across all positions in case of HR as well as NDCG. At HR@10, our model shows an improvement of around 4% (Malayalam) and 3% (Indonesian) over the state-of-the-art. However, the performance of the baselines varies over the two different datasets. Further we noticed the following performance trend for variations of our own model: HRAM >User-History Component >GMF. We do not present details due to lack of space. It may not be futile to conclude that by considering the historical user data of a user and boosting it with the information available through collaborative filtering, HRAM is able to make better recommendations.

We also varied the recurrent units used by User-History component and observed the trends as BiLSTM >LSTM >GRU >RNN. Moreover, adding an attention layer to BiLSTM led to the best performance. When we varied the negative samples used for training the model parameters, we observed that for the Malayalam dataset the performance decreases when we use more than two negative samples for every positive sample. For the Indonesian dataset the performance increases for up to four negative samples per positive sample, after which it starts decreasing. We also found out that using a window size of eight, i.e., a context/reading size of eight provided us the best results.

For evaluating our model over the item-cold start cases, for both the datasets, we segregated users who had read a new news article in the end, i.e., the last article they read was never read by any other user before they read it. We observed that for both the datasets, the HR@10 was ~0.5. We saw a gradual increase in the hit rate as we increased the value of K. This promises us that our model is well suitable for handling the item cold-start problem as well.

5 Conclusion

In this work, we proposed the Hybrid Recurrent Attention Machine for News Recommendation which utilizes the historical reading data of the user for understanding the user preference and also uses the information available through collaborative filtering for making better recommendations. We conducted extensive experiments over two real-world datasets to show that our proposed model provides significant improvement over the state-of-the-art. Further, we demonstrated its effectiveness in handling the cold start problem. In future, we would like to explore more on user profiling methods by considering various user profile factors like demographics, age etc., and incorporate it into the present system. This would also potentially help in solving the user cold-start problem.

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Robert M Bell and Yehuda Koren. 2007. Improved Neighborhood-based Collaborative Filtering. In *KDD*. 7–14.

[3] Minmin Chen, Zhixiang Xu, Fei Sha, and Kilian Q Weinberger. 2012. Marginalized Denoising Autoencoders for Domain Adaptation. In *ICML*. 767–774.

[4] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. In *WWW*. 278–288.

[5] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware Explainable Recommendation by Modeling Aspects. In *CIKM*. 1661–1670.

[6] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proc. of the* 26^{th} *Intl. Conf. on World Wide Web (WWW '17)*.

[7] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *Proc. of the* 39^{th} *Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*. ACM, 549–558.

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comp.* 9, 8 (1997), 1735–1780.

[9] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning Deep Structured Semantic Models for Web Search using Clickthrough Data. In *CIKM*. 2333–2338.

[10] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *KDD*. 426–434.

[11] Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *ICML*. 1188–1196.

[12] Xia Ning and George Karypis. 2011. Slim: Sparse Linear Methods for Top-n Recommender Systems. In *ICDM*. 497–506.

[13] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proc. of the* 25^{th} *Conf. on Uncertainty in Artificial Intelligence*. AUAI Press, 452–461.

[14] Jasson DM Rennie and Nathan Srebro. 2005. Fast Maximum Margin Matrix Factorization for Collaborative Prediction. In *Proc. of the 22nd Intl. Conf. on Machine Learning*. ACM, 713–719.

[15] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo. In *Proc. of the* 25^{th} *Intl. Conf. on Machine Learning*. ACM, 880–887.

[16] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann Machines for Collaborative Filtering. In *Proc. of the* 24^{th} *Intl. Conf. on Machine Learning*. ACM, 791–798.

[17] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proc. of the* 10^{th} *Intl. Conf. on World Wide Web*. ACM, 285–295.

[18] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet Collaborative Filtering. In *Proc. of the* $24^{th} on World Wide Web. ACM$, $111 - -112$.

[19] Florian Strub and Jeremie Mary. 2015. Collaborative Filtering with Stacked Denoising AutoEncoders and Sparse Inputs. In *NIPS Workshop on ML for eCommerce*.

[20] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *NIPS*. 3104–3112.

[21] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-n Recommender Systems. In *WSDM*. 153–162.