Word Semantics based 3-D Convolutional Neural Networks for News Recommendation

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

***Abstract*—Deep neural networks have yielded immense success in speech recognition, computer vision and natural language processing. However, the exploration of deep neural networks for content based recommendation has received a relatively less amount of inspection. Also, different recommendation scenarios have their own issues which creates the need for different approaches for recommendation. One of the problems with news recommendation is that of handling temporal changes in user interests. Hence, modelling temporal behaviour in the domain of news recommendation becomes very important.**

In this work, we propose a recommendation model which uses semantic similarity between words as input to a 3-D Convolutional Neural Network in order to extract the temporal news reading pattern of the users. This in turn improves the quality of recommendations. We compare our model to a set of established baselines and the experimental results show that our model performs better than the state-of-the-art by 5.8% (Hit Ratio@10).

***Index Terms*—Word Semantics, 3-D CNN, Content-based Recommendation, News Recommendation**

I. Introduction

The web provides instant access to a wide variety of online news. Hence, it becomes desirable to have a recommender system that would point a user to the most relevant items and thus would maximize the user engagement with the site. Each recommendation scenario has its unique attributes which creates the need for different approaches in building recommendation systems. For example, news recommendation is more focused on the freshness of the content, while movie recommendation may put more emphasis on the content relatedness. In addition, user interests constantly evolve over time. While many existing techniques assume the user preferences to be static, this seems to be an unrealistic assumption in many scenarios, particularly in news or shopping related scenarios. Hence, it becomes crucial to model the temporal nature of news reading.

A major approach to the task of recommendation is called collaborative filtering [5] [**?**] which uses the users past interaction with the item to predict the most relevant content. Another common approach is content-based recommendation [11], which uses features between items and/or users to recommend new items to the users based on the similarity between features. However, among the various approaches for collaborative filtering, matrix factorization [1] is the most popular one, which projects users and items into a shared latent space, using a vector of latent features to represent a user or an item. Thereafter, a users interaction with an item is modeled as the inner product of their latent vectors.

However, some recent research has given rise to adoption of word embedding based techniques in content-based recommendation scenarios. Authors in [2], use Word2Vec [3] in order to create a user profile based on the learned embeddings. The experimental results of this work show that a model based on word embeddings is comparable to that of well-performing algorithms based on Collaborative Filtering and Matrix Factorization. This prompts us towards the advantages of using word embeddings in recommendation scenarios.

Although, the above mentioned work tackles the problem of recommendations in general, none of them attempt to model the temporal changes that might occur in a user's interest. Typically, in a news recommendation scenario users interests keep evolving over time. It might be possible that a user who reads news articles pertaining only to politics may suddenly develop interest in sports due to various reasons. Hence, it becomes very crucial to account for the dynamic changes in interests as well as come up with better recommendations.

In this work, we come up with an approach that uses user-item interactions and the content of the news to capture the similarity between users and items (news). We only focus on implicit feedback (which indirectly reflects the preference of the users) provided by the users, i.e whether they have read a given article or not and in what sequence were those articles read by them. We use a specific amount of reading history for each user and compute a 3-D tensor for the same. We then feed this as input to a 3-D Convolutional Neural Network (CNN) [4]. Authors in [4] use 3-D CNN model for action recognition. It has been shown that 3-D convolutions are better in extracting features from the temporal dimensions. An example of 3-D convolutions can be seen in Fig. 1. We adapt the 3-D CNN to capture the temporal changes in users interests.

To summarize, the contributions of this work is as follows:

1) We present a novel 3-D CNN based model for news recommendation in which we utilize the user-item based interaction as well as the content of the read news articles.

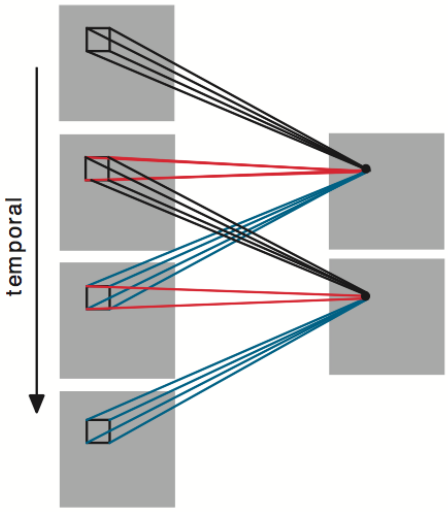

Fig. 1: An illustration of 3D convolution. In the figure the size of convolutional kernel is 3, and the set of weights are color coded so that shared weights are in the same color.

2) We perform experiments in order to demonstrate the effectiveness of our model for the task of news recommendation and show that our method performs 5.8% better than the baseline in terms of Hit Ratio@10.
3) We also compare the performance of 3-D CNN with that of its 2-D variant in order to show that temporal features are better extracted by the former.

II. Related Work

There has been a lot of work on recommender systems with a myriad of publications. In this section we attempt to review work that is closely associated to ours.

Collaborative Filtering Collaborative Filtering is an approach of making automatic prediction (filtering) about the interests of a user by collecting interests from many related users. Some of the best results are obtained based on matrix factorization techniques [5]. Collaborative Filtering methods are usually adopted when the historical records for training are scarce.

Content-based Filtering Content-based recommender systems try to recommend items similar to those a given user has liked in the past [7] [6]. The common approach is to represent both the users and the items under the same feature space. Then similarity scores could be computed between users and items. The recommendation is made based on the similarity scores of a user towards all the items. The Content-based Filtering methods usually perform well when users have plenty of historical records for learning.

Hybrid of CF and Content-based Filtering As a first attempt to unify Collaborative Filtering and Content-based Filtering, (Basilico and Hofmann 2004) proposed to learn a kernel or similarity function between the user-item pairs that allows simultaneous generalization across either user or item dimensions. This approach would do well when the user-item rating matrix is dense [8]. However in most current recommender system settings, the data is rather sparse, which would make this method fail.

Implicit Feedback Implicit Feedback originated from the area of information retrieval and the related techniques have been successfully applied in the domain of recommender systems [9] [10]. The implicit feedbacks are usually inferred from user behaviors, such as browsing items, marking items as favourite, etc. Intuitively, the implicit feedback approach is based on the assumption that the implicit feedbacks could be used to regularize or supplement the explicit training data.

3-D CNN In [4], authors develop a novel 3-D CNN model for action recognition. Unlike the 2-D CNN, in which convolutions are applied over the 2-D feature map, this model extracts features from both spatial and temporal dimensions by performing 3-D convolutions, thereby capturing the motion information encoded in multiple adjacent frames of a video.

Although the model is very useful in capturing temporal patterns, it has not been utilized in any recommendation scenarios. Such a model could be utilized for capturing the temporal interests from the user's reading behaviour and make recommendations accordingly.

III. Dataset

For this work we use the dataset published by CLEF NewsREEL 2017. CLEF NewsREEL provides an interaction platform to compare different news recommender systems performance in an online as well as offline setting [12]. As a part of their evaluation for offline setting, CLEF shared a dataset which captures interactions between users and news stories. It includes interactions of eight different publishing sites in the month of February, 2016. The recorded stream of events include 2 million notifications, 58 thousand item updates, and 168 million recommendation requests. The dataset also provides other information like the title and text of each news article, time of publication etc. Each user can be identified by a unique id. For our task, we needed to find out the sequence in which the articles were read by the users along with its content. Since, we rely on implicit feedback we only need to know whether an article was read by a user or not.

IV. Model Architecture

In this section we briefly provide the description of our model. We divide the model into three parts, 3-D Tensor, 3-D Convolutions and Score Aggregation. We then explain the training criteria for the model.

A. 3-D Tensor

Similarity Tensor is a three-dimensional structure (as seen in Fig. 2) where each element M_{ijk}, denotes the similarity between the j-th word of the i-th read article (of the user reading history) denoted by w_{ij} and k-th word of the article which is to be considered for recommendation (i.e test article) denoted by v_k :

$$M_{ijk} = w_{ij} \otimes v_k \tag{1}$$

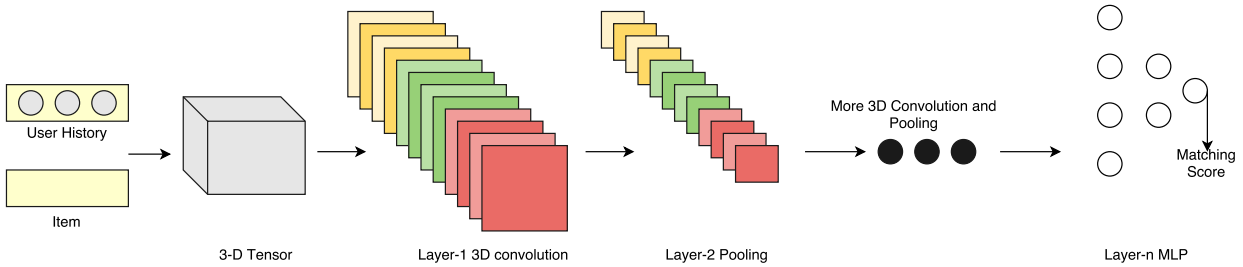

Fig. 2: Model Architecture of 3D CNN for Recommendation

where $\otimes$ stands for a general operation to obtain the similarity. We then define the general operation between w_{ij} and v_k to be the cosine similarity of their respective word embeddings.

We use Word2Vec [3] in order to learn word embeddings. We concatenate the title and text of the news article. We then compute the similarity tensor by finding out the similarity between each word of the article in the user history with that of the item that is to be considered for recommendation. One can think of the similarity tensor as a stacked representation of 2-D matrices, where each stack (2-D matrix) is a similarity matrix between an article in the user history and the test item. For example: suppose we choose the reading history for each user to be 4, then M_{234} would represent the similarity between the word embeddings of the 3rd word in the 2nd article of the user history and the 4th word in the article that is to be considered for recommendation.

B. 3-D Convolution

Based on the Similarity Tensor, we then conduct 3-D convolution to extract features that would depict the changing/evolving interests of the users. The model consists of 3-D convolution layers and pooling layers. Kernel size in each convolutional layer are the major hyper parameters. In text processing, the size of the kernel determines the number of words we want to compose together as well as the extent of temporality we would like to consider. Besides, pooling sizes in each pooling layer are also important which decide how large area we want to take as a unit.

Formally in a 3D CNN, the value at position (x, y, z) on the j[th] feature map of the i[th] layer is given by,

$$v_{ij}^{xyz} = \tanh(b_{ij} + \sum_{m} \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{pqr}^{ijm} x_{(i-1)m}^{(x+p)(y+q)(z+r)}) \tag{2}$$

where, where R_i is the size of the 3D kernel along the temporal dimension, w_{pqr}^{ijm} is the (p, q, r)[th] value of the kernel connected to the m[th] feature map in the previous layer.

C. Score Aggregation

After a series of convolutional layers followed by pooling layers, three additional fully connected layers are used to aggregate the information into a single matching score/target value. In this paper we use 128 hidden units for the first hidden layer followed by 64 units for the second hidden layer. We use ReLU as the activation function for these layers. For the final output unit we use the Logistic Function.

D. Training

Typically in matrix factorization, to learn the model parameters, existing pointwise methods [13] perform regression with a squared loss. This is based on the assumption that observations are generated from a Gaussian distribution. The final output layer has the predicted score $\hat{y_{ux}}$, and training is performed by minimizing the pointwise loss between $\hat{y_{ux}}$ and its target value y_{ux}. Considering the one-class nature of implicit feedback, we can view the value of y_{ux} as a label 1 meaning the item x is relevant to a user u, and 0 otherwise. The prediction score $\hat{y_{ux}}$ then represents how likely an item x is relevant to u. Hence in order to constrain the values between 0-1 we use the logistic function. We then define the likelihood function as,

$$p(\gamma, \gamma^- | M, \Theta_m) = \prod_{(u,i)\in\gamma} \hat{y_{ui}} \prod_{(u,j)\in\gamma^-} (1 - \hat{y_{uj}}) \tag{3}$$

, where γ, γ^- represent the positive (observed interactions) and negative (unobserved interactions) samples/items, M represents the similarity tensor and Θ_m represents the parameters of the model. Taking the negative log likelihood we reach,

$$L = -\sum_{u,i\in\gamma\cup\gamma^-} y_{ui} \log \hat{y_{ui}} + (1 - y_{ui})(1 - \log \hat{y_{ui}}) \tag{4}$$

This is the objective function to minimize, and its optimization can be done by performing stochastic gradient descent (SGD). Careful readers might have realized that it is the same as the binary cross-entropy loss, also known as log loss. By employing a probabilistic treatment for, we address recommendation with implicit feedback as a binary classification problem.

V. Experiments

As mentioned earlier we use the data provided by CLEF NewsReel 2017. We choose users who have read in between 10-15 (inclusive) articles for training and testing our model for item recommendation. The frequency of users who have read more than 15 articles varies extensively and hence we restrict ourselves to the upper bound of 15. We set the lower bound to 8 since we need some history in order to capture the changing user interests. However, for future work we would like to investigate how changing the lower bound affects the performance of our model.

Evaluation Protocol: For each user we held-out her latest interaction as the test set and utilized the remaining data for training. We then recommend a ranked list of articles to each user. The performance of a ranked list is judged by Hit Ratio (HR) and Normalized Discounted Cumulative gain (NDCG). Without special mention we truncate the ranked list at 10 for both metrics.

Baselines: We compare our method with several others. First we look at item popularity based method (ItemPop). In this we recommend the most popular items to the user. We then evaluate User-to-User (U2U-KNN) and Item-to-Item (I2I-KNN) by setting the neighbourhood size to 80. We then compare it with Singular Value Decomposition (SVD). We also implement Word Embeddings based Recommendations as in [2] and Keyword based Vector Space Model (Key-VSM) as mentioned in [7]. Further in order to demonstrate the effectiveness of our model, we also compare it with its 2-D CNN variant.

Parameter Settings: We implemented our proposed model using Keras [14]. We then construct our training set as follows:

1) We first define the reading history. We denote the reading history by h.
2) Leaving the latest article read by each user, the remaining articles are used as positive samples.
3) Corresponding to each positive sample, we randomly sample 4 negative instances (articles which the user did not read).

We then randomly divide the training set into training and validation set in a 4:1 ratio. This helps us to ensure that the two sets do not overlap. We tuned the hyper-parameters of our model using the validation set. We use a batch size of 256. In the first layer of the model we apply a 3-D Convolution of size 3x3x3, followed by a max pooling layer with a pooling size of 2x2x2. We then repeat 3-D convolution with the same kernel size followed by a pooling layer with pooling size of 1x2x2. We experimented with different variations in kernel and pooling sizes. The above mentioned seemed to be performing the best.

VI. Performance Comparison

From Fig. 3 it can be clearly seen that 3-D CNN outperforms the respective baselines in terms of the Hit Ratio. We also see that the NDCG scores of 3-D CNN is high as well. One can argue that this might be a result of using the loss function in Eq. 4. Further it can be clearly noticed that U2U, I2I and SVD do not perform well. One reason for this could be the sparsity of the data. In presence of sparse data these methods fail to capture relevant information. The low performance of Word Embedding based Recommendations suggests that a representation of words alone is not effective in profiling the user. The model also outperforms Key-VSM [7] as well as its 2-D CNN variant. Key-VSM and 2-D CNN are fairly able to capture the users interests but are still not at par with 3-D CNN. This shows that in tasks such as news recommendation, where the user interests keep varying, 3-D CNN is better in accounting for the temporal changes.

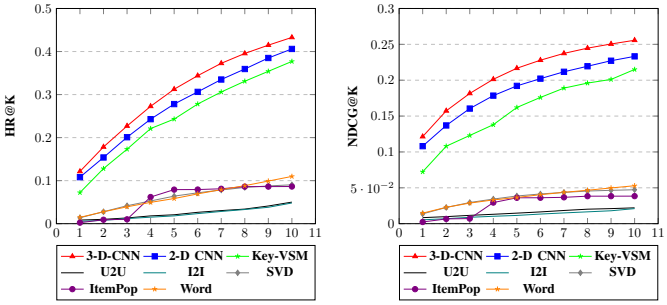

Fig. 3: Performance of our model vs state-of-the-art models

VII. Conclusion and Future Work

In this work we explore the idea of using semantic similarity in combination with 3-D CNN for capturing temporal changes in the users interests in order to provide better recommendations. For future work we would like to come up with a model which combines the aspects of collaborative filtering with that handling temporal changes in users interests. We would also like to experiment on CNN in combination with Recurrent Networks in order to profile the interests of a user and come up with better recommendations.

References

[1] Koren, Yehuda. "Factorization meets the neighborhood: a multifaceted collaborative filtering model." Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008.

[2] Musto, Cataldo, et al. "Learning word embeddings from wikipedia for content-based recommender systems." European Conference on Information Retrieval. Springer International Publishing, 2016.

[3] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems. 2013.

[4] Ji, Shuiwang, et al. "3D convolutional neural networks for human action recognition." IEEE transactions on pattern analysis and machine intelligence 35.1 (2013): 221-231.

[5] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009).

[6] Saia, Roberto, Ludovico Boratto, and Salvatore Carta. "Semantic Coherence-based User Profile Modeling in the Recommender Systems Context." KDIR. 2014.

[7] Lops, Pasquale, Marco De Gemmis, and Giovanni Semeraro. "Content-based recommender systems: State of the art and trends." Recommender systems handbook. Springer US, 2011. 73-105.

[8] Basilico, Justin, and Thomas Hofmann. "Unifying collaborative and content-based filtering." Proceedings of the twenty-first international conference on Machine learning. ACM, 2004.

[9] Kelly, Diane, and Jaime Teevan. "Implicit feedback for inferring user preference: a bibliography." ACM SIGIR Forum. Vol. 37. No. 2. ACM, 2003.

[10] Oard, Douglas W., and Jinmook Kim. "Implicit feedback for recommender systems." Proceedings of the AAAI workshop on recommender systems. Menlo Park, CA: AAAI Press, 1998.

[11] Kompan, Michal, and Mria Bielikov. "Content-Based News Recommendation." EC-Web. Vol. 61. 2010.

[12] Hopfgartner, Frank, et al. "Benchmarking news recommendations: The clef newsreel use case." ACM SIGIR Forum. Vol. 49. No. 2. ACM, 2016.

[13] Mnih, Andriy, and Ruslan R. Salakhutdinov. "Probabilistic matrix factorization." Advances in neural information processing systems. 2008.

[14] Chollet, Franois. "Keras." (2015).