Multi-Rate Deep Learning for Temporal Recommendation

Yang Song
Microsoft Research Redmond
yangsong@microsoft.com

Ali Mamdouh Elkahky
Columbia University
ame2154@columbia.edu

Xiaodong He
Microsoft Research Redmond
xiaohe@microsoft.com

ABSTRACT

Modeling temporal behavior in recommendation systems is an important and challenging problem. Its challenges come from the fact that temporal modeling increases the cost of parameter estimation and inference, while requiring large amount of data to reliably learn the model with the additional time dimensions. Therefore, it is often difficult to model temporal behavior in large-scale real-world recommendation systems. In this work, we propose a novel deep neural network based architecture that models the combination of long-term static and short-term temporal user preferences to improve the recommendation performance. To train the model efficiently for large-scale applications, we propose a novel pre-train method to reduce the number of free parameters significantly. The resulted model is applied to a real-world data set from a commercial News recommendation system. We compare to a set of established baselines and the experimental results show that our method outperforms the state-of-the-art significantly.

1. INTRODUCTION

Recommendation systems are core components of many of the modern Internet services including News, E-commerce, online movie sites and more. Each recommendation scenario has its unique attributes which creates needs 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 unrealistic assumption in many scenarios, particularly in News or shopping related scenarios. For example, In a previous work [6], the authors showed that users who visited *spligle.de*, a popular German news portal, are likely to be interested in football-related news, the reason being that the data was collected around the dates of world cup 2014. While the trained model may work very well for a period of time, eventually after the event the model's performance will degrade significantly and therefore needs to be retrained.

One way to handle temporal changes in user's interests is to use only the recent interactions from the users in order to provide the most timely recommendation. While this may work for users of who have rich interactions with the systems, it will fail in the cold-start scenario. In addition, this approach needs constantly re-train the model, which is both expensive and difficult for performance evaluation.

A better approach is therefore to make use of both long-term user history as well as their short-term temporal interests. In this case, we assume that user's preferences are composed of two components: the long-term preference which reflects the *fairly stable* interests of the users based on their online activities; and the temporal interests which represents the users' current immanent need/interests.

Specifically, in this work, we propose a *multi-rate* temporal deep learning model that jointly optimizes long-term and short-term user interests to improve the recommendation quality. The model consists of several components: a Deep Semantic Structured Model (DSSM) [11] to model user static interests; two LSTM-based temporal models to capture daily and weekly user temporal patterns; and an LSTM temporal model to capture global user interests. The term multi-rate indicates the capability of our model which is able to capture user interests at different granularity, so that temporal dynamics at different rates can be effectively and jointly optimized.

2. RELATED WORK

In recommendation systems, temporal modeling of data is an important element in many tasks such as news recommendation [16, 15], movie recommendation [14, 23], and recently, music recommendation [12]. It has been shown to improve results over non-temporal model significantly. In this section, we will survey some of the recent approaches and refer readers to a more extensive review on temporal recommendation in [5].

In [14], the matrix factorization model is extended to allow each user to have a base latent vector and another set of time dependent vectors. A regularization term to enforce each time-dependent vector to be similar to the one in the previous time step is added to maintain the smoothness over time. A Bayesian extension to this work was introduced in [23] where regularization is replaced by prior distributions. By treating matrix factorization as an autoregressive model, the authors introduced a temporal matrix factorization approach [24] which aims at predicting next

SIGIR '16, July 17-21, 2016, Pisa, Italy

DOI: http://dx.doi.org/10.1145/2911451.2914726

steps from historical data. A simple and scalable Bayesian approach for News recommendations was proposed in [16]. The authors used a probability distribution of user's topic preferences at the current time which have priors that depends on both the user's historical preferences as well as the preferences of other users sharing the same geographic location. In [25], the authors used collaborative filtering technique to compute explicit user similarity function and extends the similarity function to incorporate temporal pattern similarities between users. Another approach presented in [2] divided user interaction with the system into discrete partitions, each of which represents the user interests in a particular time range. A similar approach that mines temporal sequential patterns from usage data was presented in [23] where the authors modeled the transition between (user, item, time) nodes and then used certain rules to mine meaningful temporal patterns and fused them with long term user interests. In [15], long-term user history was leveraged to provide coarse grain news recommendation for certain news groups. The short-term history of the user was then used to recommend specific news articles within the selected groups.

Using deep learning approaches for recommendation systems has recently received many attentions [20, 21, 22]. However, using deep learning for temporal recommendation has not yet been extensively studied. In [8], the authors proposed to use Recurrent Neural Networks (RNN) for recommending shopping items to users based on the user's current session history. A similar idea of session-based recommendation was proposed for music recommendation in [1] by modeling music playlist as sessions.

3. TEMPORAL DEEP SEMANTIC STRUCTURED MODEL (TDSSM)

The Deep Semantic Structured Models (DSSM) was proposed in [11] for ranking purpose and was later on extended to the recommendation scenarios in [6]. We briefly review DSSM here. Essentially, DSSM can be viewed as a multi-view learning model that often composes of two or more neural networks for each individual view. In the original two-view DSSM model, the left network represents the query view and the right network represents the document view. The input of each neural network can be arbitrary types of features, e.g., letter-tri-gram used in the original paper [11], or bag of unigram features used in [6]. Each input feature vector goes through non-linear transformations in the feedforward network to output an embedding vector, which is often much smaller than the original input space. The learning objective of DSSM is to maximize the cosine similarity between the two output vectors. During training, a set of positive examples and randomly sampled negative examples are generated in each minibatch to minimize the cosine loss on the positive examples. In [6], the authors used DSSM for recommendation where the first neural network contains user's query history (and thus referred to as user view) and the second neural network contains implicit feedback of items (e.g., News clicks, App downloads). The resulting model is named multi-view DNN (MV-DNN) since it can incorporate item information from more than one domains and jointly optimize all of them using the same loss function in DSSM.

In the MV-DNN model (and DSSM as well), both the user view and item view are *static* in the sense that the input features represent a holistic view of users interests during a certain period of time. The model thus lacks the ability of responding promptly to some scenarios where freshness and the temporal dynamics of the items is sometimes more important than the content relevance itself, e.g., News recommendation. Therefore, we propose to extend DSSM to include temporal information by incorporating recurrent neural networks (RNN) into the user-view of the model. Instead of modeling the entire user history using RNN which is difficult to train, our model combines the long-term user preferences which remains relatively stable over time, with the short-term temporal user interests using feedforward neural network as follows:

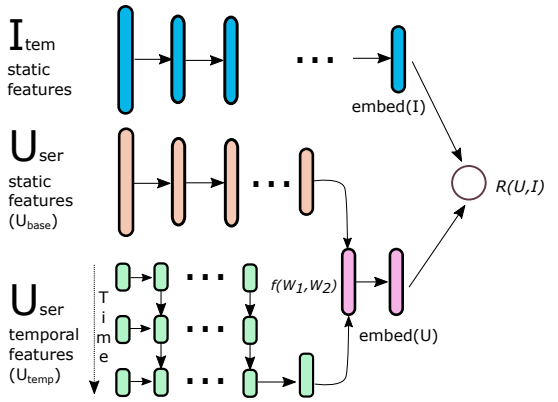

Figure 1: The temporal DSSM (TDSSM) model for temporal user modeling.

$$E(U, t_i) = f_{U,t_i}(E_{base}(U), E_{t_i}(U)) \quad (1)$$

where $E(U, t_i)$ is the output embedding of the user U at time t_i, which is a combination of the baseline user preferences $E_{base}(U)$ and the temporal user preference $E_{t_i}(U)$. Here f is a aggregation function which can take one of the three forms:

$$f(W_1, W_2) = \begin{cases} W_1 \odot W_2 & \text{elem-wise multiplication} \\ W_1 \odot A \odot W_2 & \text{weighted elem-wise multiplication} \\ [W_1; W_2] & \text{concatenation} \end{cases}$$

On the other hand, the content of the items often remain unchanged. This implies that we can use the same embedding for the items over different time period and thus makes the training tractable. The objective function can then be written as:

$$\min_{W_{user} W_{item}} -\log \prod_{user, item^+, t_i} p(item^+ | user, t_i) \quad (2)$$

where $p(item^+|user, t_i)$ is defined as:

$$p(item^+ | user, t_i) = \frac{e^{cos\left(E(user,t_i), E_{item^+}\right)}}{\sum_{\forall item} e^{cos(E(user,t_i), E_{item})}} \quad (3)$$

Figure 1 shows the architecture of the proposed model, namely the temporal DSSM (TDSSM). The key difference to the previous model is the introduction of the user temporal features network, where we use RNN to model user interests at different time spots. One design decision in this model is how we choose the granularity of each input time spot t_i. While modeling using smaller time spans (say hourly)

can capture more fine-grained interest changes, it will make the feature space very sparse and thus make the learning difficult. On the other hand, having time large spans may lead to less sparse feature spaces but also makes the model less adaptive and efficient for capturing temporal dynamics. The next section addresses this issue by extending the current TDSSM.

3.1 Mutli-Rate TDSSM

Instead of having temporal embedding $E(user, t_i)$ computed on a single granularity, we propose a new model to that aggregate temporal information at different rates, each of which is modeled using a separate RNN. *Fast-rate* RNNs are used to adapt to very recent user interests, while *slow-rate* RNNs are designed to model seasonal user interest shift. RNNs at different rates are combined using fully-connected feedfoward network. This design allows for the independence of each RNN in terms of feature aggregation, while also in principle jointly optimizes the parameters from all RNNs and the baseline model effectively. The resulted model is named multi-rate TDSSDM, or MR-TDSSM. One drawback of such model is the large number of model parameters from the RNNs. In the next section, we discuss how to use a technique called pre-train to reduce the free parameters and speed up the training process.

4. TRAINING SPEED-UP USING PRE-TRAIN

Pre-train is a known approach to make neural network training faster and more effective [9, 4]. Previously, pre-train was used to train deep networks in incremental fashion where training of the first layer is done separately then its output is taken as input to the next layer. Recently, it becomes popular to use pre-train of word embedding for NLP applications [17], by first training on a large unlabeled data set, then use the trained embedding in the target supervised task.

In this work, we use a similar idea as word embedding to initialize the embedding of user and item feature vectors via additional training data. To be concrete, we started by training a DSSM model in the same way as proposed in [6]. The resulted DSSM is then used to generate embedding for MR-TDSSM inputs.

The use of pre-train reduces the size of input feature space by a factor proportional to the ratio between the size of the original feature space (i.e. $\|X_{user}\| = D_{user}$ and $\|X_{item}\| = D_{item}$) to the size of the embedding space(i.e. $\|E_{user}\| = \|E_{item}\| = d$). In addition, it no longer requires the item view to go through another multi-layer neural networks. The reduction of the parameters in the input layers comes with the cost of converting those inputs from sparse vectors to dense ones during initialization. However, this operation actually improves the training efficiency since implementing efficient operations for sparse vectors is still a challenging task[3].

5. EXPERIMENTAL SETUP AND RESULTS

The data set used in our experiment comes from a commercial news portal which serves millions of daily users in a variety of countries and languages. The dataset consists of user's news click history between 04/01/2014 and 09/30/2014. The first four months are used for training the model while the last two months are used for testing. In order to get enough signal for the temporal model, we filtered out users who have less than 20 news clicks during those six months. This resulted in 64,669 users, 26.5 million training pairs of $(user, news)$ clicks for training and 13.3 million pairs for testing. For each test triplet $(user, news, time)$, we generate 9 random news from the same time span for evaluation. The evaluation objective is to see how well the model can rank the clicked news on top of those randomly sampled news. Therefore, we employee five popular metrics for evaluation: precision at position 1 (P@1) and 5 (P@5), Area Under Curve (AUC), Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR).

To make the result more convincing, we also compare with several state-of-the-art recommendation algorithms. Those include SVD++ and RegSVD [18], WRMF [10], CF-KNN [13] and AoBPR [19]. Note that matrix factorization techniques including SVD++, RegSVD, WRMF and CF-KNN cannot deal with out-of-matrix predictions thus during evaluation novel users are removed from these algorithms. We also added the popularity-based baseline by ranking news based on the number of clicks they received. For those baseline algorithms we used the implementation of LibRec from [7].

For DSSM implementation, we used the implementation of Sent2Vec from [11]. The output embedding is set to 300. To implement the TDSSM and MR-TDSSM, we used Theano [1] and Keras[2]. The RNNs in the models are implemented using LSTM in Keras. We also tried GRU but the results seem to be worse than LSTM. To verify whether the RNN model itself can achieve good performance for evaluation, we also trained an LSTM-only model that uses only recent user embedding. In addition, a variant of the LSTM-only model which adds the user static input as the input in the beginning of the model is also evaluated. Additionally, to verify the recency effect, we implemented a baseline which leverages the embedding from the immediate previous time spot to test the performance of the news on the current time spot.

For TDSSM we used previous two-week clicks as short-term history, with each day as a time spot. Therefore, the length of the LSTM for TDSSDM is 14. For MR-TDSSM, we implemented two LSTMs in different rates, where the fast-rate LSTM uses daily signals and the slow-rate LSTM uses weekly signals.

Table 1 summarizes the results. We first observe that the most-popular baseline performs no better than random guess, indicating that popularity does not necessarily correlates with user interests. Secondly, we can observe that most matrix factorization-based techniques performed quite poorly, with the only exception of WRMF [10] which takes implicit feedback as additional signals and thus performed well (0.195 P@1) on this data set. Now let's examine the performance of models that take short-term user interests into consideration. We first observe that by only using the immediate previous day's news click, the model (prev-day click) already shows better performance (0.211 P@1) than all MF-based methods, with competitive results against the DSSM baseline (0.229 P@1) which only leverages user long-term static features. On the other hand, LSTM-based methods (LSTM-only and LSTM-DSSM) failed to outperform

[1]https://github.com/Theano/Theano
[2]http://keras.io

Table 1: Performance comparison of our models vs. state-of-the-art methods.

Method	P@1	P@5	AUC	MAP	MRR
Most Popular	0.105	0.053	0.504	0.042	0.284
SVD++ [13]	0.157	0.072	0.647	0.056	0.296
RegSVD [18]	0.156	0.072	0.638	0.052	0.294
AoBPR [19]	0.168	0.076	0.659	0.064	0.315
UserKNN (k=80)	0.174	0.093	0.684	0.068	0.323
WRMF [10]	0.195	0.114	0.692	0.071	0.347
Prev-day Click	0.2109	0.119	0.727	0.074	0.358
DSSM [11]	0.229	0.126	0.763	0.085	0.379
LSTM-Only	0.189	0.102	0.688	0.07	0.329
LSTM-DSSM	0.196	0.116	0.691	0.073	0.349
TDSSM	0.231	0.134	0.779	0.091	0.381
MR-TDSSM	0.245	0.147	0.814	0.099	0.397

the DSSM model, which indicates that ignoring the long-term user interests may not lead to optimal performance. Since the short-term user history is often quite sparse, models like LSTM that has many training parameters cannot learn enough evidence from the sparse inputs. Finally, by combining long-term and short-term user interests, our proposed models TDSSM and MR-TDSSM successfully outperformed all the methods significantly. We can notice that by adding a slow-rate LSTM (weekly-based features) to the MR-TDSSM, it leads to great performance improvement over TDSSM with only one fast-rate LSTM component.

6. CONCLUSION

In this work we explored the use of deep learning for temporal news recommendation. We showed that deep learning can be effective in learning temporal recommendation models by combining traditional feedforward networks (DSSM) with recurrent networks (LSTM). To address the issue of different granularity, we also introduced the MR-TDSSM which had the ability to model sequence of data at multiple levels of rate. In addition, we showed that pre-train can significantly decrease the number of parameters in the model and make it efficiently trainable. Comparisons with several state-of-the-art methods indicated that both of models can significantly improve the recommendation performance.

Future work includes applying this model to different recommendation scenarios. One obvious domain is shopping recommendation where user's history of recently purchased items can be good indicators of their immediate purchasing interests. For the models, we also want to explore the attention-based memory network to further improve the performance of LSTM models.

7. REFERENCES

[1] Recurrent neural networks for collaborative filtering. http://erikbern.com/2014/06/28/recurrent-neural-networks-for-collaborative-filtering/. Accessed: 2016-01-18.

[2] L. Baltrunas and X. Amatriain. Towards time-dependant recommendation based on implicit feedback. In *Workshop on (CARS09)*, 2009.

[3] N. Bell and M. Garland. Implementing sparse matrix-vector multiplication on throughput-oriented processors. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, page 18. ACM, 2009.

[4] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. 2007.

[5] P. G. Campos, F. Díez, and I. Cantador. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 24(1-2):67–119, 2014.

[6] A. M. Elkahky, Y. Song, and X. He. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of WWW2015*, pages 278–288, 2015.

[7] G. Guo, J. Zhang, Z. Sun, and N. Yorke-Smith. Librec: A java library for recommender systems.

[8] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. *CoRR*, abs/1511.06939, 2015.

[9] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[10] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM'08*.

[11] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck. Learning deep structured semantic models for web search using clickthrough data. In *CIKM'13*.

[12] N. Koenigstein, G. Dror, and Y. Koren. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *Recsys'11*.

[13] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD'08*.

[14] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[15] 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, 2014.

[16] J. Liu, P. Dolan, and E. R. Pedersen. Personalized news recommendation based on click behavior. In *Proceedings of the 15th international conference on Intelligent user interfaces*, pages 31–40. ACM, 2010.

[17] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[18] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. 2007.

[19] S. Rendle and C. Freudenthaler. Improving pairwise learning for item recommendation from implicit feedback. In *WSDM'14*.

[20] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[21] A. Van den Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In *NIPS'13*.

[22] H. Wang, N. Wang, and D.-Y. Yeung. Collaborative deep learning for recommender systems. *arXiv preprint arXiv:1409.2944*, 2014.

[23] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. SIAM.

[24] H. Yu, N. Rao, and I. S. Dhillon. Temporal regularized matrix factorization. *CoRR*, abs/1509.08333, 2015.

[25] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Time-aware point-of-interest recommendation. In *SIGIR'13*.