Neural Expert Recommendation

Seyedeh Mina Mousavifar
sem311@mail.usask.ca
University of Saskatchewan
Saskatoon, SK, Canada

ABSTRACT

In community question and answer (CQA) platforms, such as Stack Overflow(SO), recommending relevant experts for answering questions is crucial. Personalized recommendation models users' preference for items based on their past interactions, known as Collaborative filtering(CF)[19]. The most popular collaborative filtering technique is matrix factorization (MF), which utilizes a vector of latent features to represent a user or an item that can model user's interaction on an item as the inner product of their latent vectors[12]. In this paper, I implemented a neural network recommender(NCF)[8] based on implicit feedback to suggest experts in CQA platforms. This approach combines traditional matrix factorization and multi-layer perceptron to perform the recommendation, which generalizes the MF method and captures the non-linearity in the latent space. Finally, I conducted extensive experiments on a large real-world dataset and benchmarked the performance of NCF on the expert recommendation task. Empirically, the recommendation in SO using NCF indicated that the inner-product of latent features of questions and experts with different weights offers better performance.

KEYWORDS

community question answering, expert recommendation, neural matrix factorization

ACM Reference Format:
Seyedeh Mina Mousavifar. 2020. Neural Expert Recommendation. In *CMPT898: Deep Learning and Applications course, Winter, 2020, University of Saskatchewan, SK.* , 5 pages.

1 INTRODUCTION

Proper coordination between questions and the potential answerers is the core function in CQA platforms. However, SO lacks coordination, which adversely affects knowledge acquisition. An expert might encounter various new inquiries without being able to distinguish their question of interest rapidly, and a question may experience several low-quality answers without receiving any proper response in a limited timeframe. A recent study on Stack Overflow [1] demonstrates that these platforms have highly committed domain experts, who target not only the requester question but also a long-lasting answer to a broader crowd. Therefore, expert recommendation is crucial in SA by bringing experts' attention to relevant questions effectively and instantly.

Expert recommender systems are algorithms aimed at suggesting relevant questions to experts. Collaborative filtering(CF)[19] models experts' preference for questions based on their past interactions. These past expert-question interactions are sufficient to detect similar experts or similar questions and make predictions based on these estimated proximities. The most popular collaborative filtering technique is matrix factorization (MF), which utilizes a vector of latent features to represent an expert or a question that can model an expert's interaction on a question as the inner product of their latent vectors[12]. MF assumes that each dimension of the latent space is independent of each other and linearly combining them with the same weight. However, neural network recommender(NCF) [8] consists of two separate models, Generalized Matrix Factorization(GMF) and Multi-Layer Perceptron(MLP) to generalize the MF method and capture the non-linearity in the latent space respectively. This method aims to learn the complex structure of expert-question interaction data by allowing varying importance of latent dimensions and learning non-linear functions for the interaction. I summarize the contributions of this project as follows:

- I examined the state-of-the-art algorithm for collaborative filtering based recommendation in Stack Overflow by conducting extensive experiments on a large real-world dataset.
- I explored the importance of latent features against the interaction decomposition function and highlighted the critical aspects of future deep recommenders in Stack Overflow.

2 LITERATURE REVIEW

Recommendation on CQA platforms has mainly centred on linking questions to platform users proficient in a particular question domain. Various research efforts have attempted to find experts for questions in different CQA platforms using several approaches such as graph-based algorithms [18], statistical models [16], social network analysis or link analysis [13], and ranking metrics [27]. However, recommending experts in CQA platforms with Machine Learning algorithms can't capture the complicated structure of these networks. Dijk et al. [3] transformed early detection of topical expertise issues into a classification problem. They extracted textual, behavioural, and time-aware features from the SO dataset to predict whether a user will be an expert - users with ten or more accepted answers - in the future. They implemented and evaluated the following algorithms: Gaussian Naïve Bayes, Random Forest, and Linear Support Vector. Their research revealed that Random Forest performed better than Gaussian Naïve Bayes and Linear Support Vector, which verify the needs for more complex models to deal with this problem.

Recent works [4, 21, 23, 24, 26] have investigated deep learning models for the recommendation. They primarily used DNNs for modelling auxiliary information, such as the textual description of items [23], acoustic features of music [21, 24], cross-domain behaviours of users [4], and the valuable data in knowledge bases

[26]. The most related work is [25], which presents a collaborative denoising autoencoder (CDAE) for CF with implicit feedback. In contradiction to the DAE-based CF [20], CDAE also plugs a user node to autoencoders' input to build the user's ratings. CDAE is equal to the SVD++ model [11] when the identity function is applied to activate the hidden layers of CDAE. Separated from CDAE, NCF [8] utilizes a two-pathway architecture, modelling user-item interactions with a multi-layer feed-forward neural network. This allows NCF to learn an arbitrary function from the data, capturing more complexity and expressive than the fixed inner product function. However, the application of this method in SO, which doesn't yield proper recommendation using Machine Learning algorithms, hasn't been explored yet.

3 ALGORITHM DESCRIPTION

The model introduced as NeuMF[8] provides a general framework that is generic and can express and generalize matrix factorization. This model is the fusion of two separate models, Generalized Matrix Factorization(GMF) and Multi-Layer Perceptron(MLP). Since NeuMF solely focuses on collaborative filtering, it uses the interaction matrix, which is the one-hot encoding of expert and question IDs as input. So the interaction matrix is a sparse matrix, where ui is the interaction between a specific expert u and a question i. The embedding layer is after the input layer, a dense layer that decomposes the sparse matrix into expert and question latent vectors. The target value y_{ui} is a binarized 1 or 0, as 1 means expert u has answered question i, and 0 otherwise. The prediction score $\hat{y}_{ui}$ then represents how likely the question i will be answered by the expert u. The training is performed by minimizing the pointwise loss between the predicted score of $\hat{y}_{ui}$ and its target value of y_{ui}. Then the likelihood function is defined as binary cross-entropy loss, also known as log loss. So, the recommendation problem can be addressed as a binary classification problem.

$$p(\gamma, \gamma^{-}|\mathbf{P}, \mathbf{Q}, \theta_f) = \Pi_{(u,i)\in\gamma}\hat{y}_{ui}\Pi_{(u,i)\in\gamma^{-}}(1-\hat{y}_{ui})$$

Where $\mathbf{P} \in R^{M\times K}$ and $\mathbf{Q} \in R^{N\times K}$ denotes the latent factor matrix for experts and questions and θ_f denotes the model parameters of the interaction function f.

By taking the negative logarithm of the likelihood:

$$\begin{aligned} L &= -\sum_{(u,i)\in\gamma} log(\hat{y}_{ui}) - \sum_{(u,i)\in\gamma^{-}} log(1-\hat{y}_{ui}) \\ &= -\sum_{(u,i)\in(\gamma\cup\gamma^{-})} y_{ui}log(\hat{y}_{ui}) + (1-y_{ui})log(1-\hat{y}_{ui}) \end{aligned}$$

3.1 Generalized Matrix Factorization(GMF)

GMF aims to generalize the MF method [12] by allowing the interaction function determined from a large family of factorization models[15]. The mapping function of the first layer of this model is:

$$\phi(\mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u \odot \mathbf{q}_i$$

Where, the expert latent vector $\mathbf{p}_u$ and the question latent vector $\mathbf{q}_u$ equals $\mathbf{P}^T\mathbf{v}_u^U$ and $\mathbf{Q}^T\mathbf{v}_v^V$ respectively. Then the output layer of this model is:

$$\hat{y}_{ui} = a_{out}(\mathbf{h}^T(\mathbf{p}_u \odot \mathbf{q}_i))$$

Where the output layer uses the sigmoid activation function with the binary cross-entropy loss function. If we use the uniform vector 1 as $\mathbf{h}$ and the identify function for a_{out}, this model performs the same as MF. Therefore, this model is the generalized version of MF.

For training GMF, mini-batch Adaptive Moment Estimation(ADAM)[10] is used due to its efficiency in tuning the learning rate for each parameter by slightly changing the learning rate for frequent parameters and significantly altering learning rate for the infrequent parameter. Moreover, I initialized the parameters using a random initialization from a Gaussian distribution (with a mean of 0 and a standard deviation of 0.01).

3.2 Multi-layer Perceptron(MLP)

GMF concatenate experts' and questions' features. However, this concatenation may not capture the interactions between expert and question features. So, I utilize a standard multilayer perceptron to learn the interaction between experts and questions latent features. For learning the interaction between expert and question latent features $\mathbf{p}_u$ and $\mathbf{q}_i$, instead of learning the element-wise product in GMF, the following method will be applied:

$$\begin{aligned} \mathbf{z}_1 &= \phi(\mathbf{p}_u, \mathbf{q}_i) = \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix} \\ \phi_2(\mathbf{z}_1) &= a_2(\mathbf{W}_2^T\mathbf{z}_1 + \mathbf{b}_2) \\ &\dots \\ \phi_2(\mathbf{z}_{L-1}) &= a_L(\mathbf{W}_L^T\mathbf{z}_{L-1} + \mathbf{b}_L) \\ \hat{y}_{ui} &= \sigma(\mathbf{h}^T\phi_L(\mathbf{z}_{L-1})) \end{aligned}$$

Where $\mathbf{W}_x$, $\mathbf{b}_x$, and a_x denote the weight matrix, bias vector, and activation function for the x-th layer's neural network.

Furthermore, I will use ReLu[4] for activation with the binary cross-entropy loss function. ReLu is chosen because of its proven resistance to saturation and being suitable for sparse data[5]. Furthermore, the sigmoid function suffers from saturation, and the tanh function is a rescaled version of sigmoid with lower saturation. For training MLP, mini-batch Adaptive Moment Estimation(ADAM)[10] is used due to its competence in tuning the learning rate. Moreover, I initialized the parameters using a random initialization from a Gaussian distribution (with a mean of 0 and a standard deviation of 0.01) until convergence. For network structure design, a standard solution is to adopt a tower model, where the bottom layer is the largest. Each successive layer has fewer neurons to learn more abstractive features of the data by using a small number of hidden units for higher layers [6]. I empirically implement the tower structure, halving the layer size for each successive upper layer. This model employs three hidden layers, and the factors of [8, 16, 32, 64] are tested for the number of neurons for the last layer.

3.3 Fusion of MLP and GMF(NeuMF)

So far, I have developed two models, GMF, which uses a linear kernel to model the latent feature interactions and MLP, which uses a non-linear kernel to learn the interaction function from data. For adding more flexibility to the fused model, I allow GMF and MLP to learn separate embeddings and combine the two models by concatenating their last hidden layer. Consequently, NeuMF combines the linearity of MF and non-linearity of deep neural networks for modelling expert–question latent structures.

$$\phi^{GMF} = \mathbf{p}_u^G \odot \mathbf{q}_i^G$$

$$\phi^{MLP} = a_L(\mathbf{W}_L^T(a_{L-1}(...a_2(\mathbf{W}_2^T \begin{bmatrix} \mathbf{p}_u^M \\ \mathbf{q}_i^M \end{bmatrix} + \mathbf{b}_2)...)) + \mathbf{b}_L)$$

$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \begin{bmatrix} \phi^{GMF} \\ \phi^{MLP} \end{bmatrix})$$

Where $\mathbf{p}_u^G$ and $\mathbf{p}_u^M$ are expert embeddings for GMF and MLP model, and $\mathbf{q}_i^G$ and $\mathbf{q}_i^M$ are for question embeddings. Finally, each model parameter can be calculated with standard back-propagation. The architecture of NeuMF is presented in figure 1.

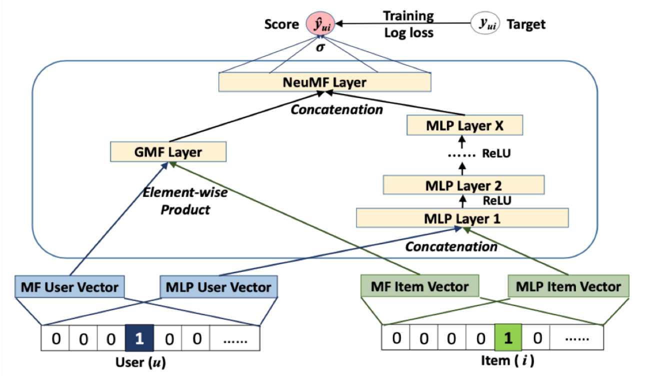

Figure 1: Neu MF architecture

Two approaches are tested for training NeumF, firstly, pre-training MLP and GMF separately and initializing parameters of NeuMF with the learned parameters, and secondly, training NeuMF without pre-training. For the pre-trained model, the NeuMF parameters will be initialized by pre-trained models of GMF and MLP. So, the weights of the two models are concatenated with:

$$\mathbf{h} \leftarrow \begin{bmatrix} \alpha \mathbf{h}^{GMF} \\ 1 - \alpha \mathbf{h}^{MLP} \end{bmatrix}$$

Where $\mathbf{h}^{GMF}$ and $\mathbf{h}^{MLP}$ denote the $\mathbf{h}$ vector of pre-trained GMF and MLP model and α is a hyper-parameter showing the trade-off between the two models. For training NeuMF, the Stochastic Gradient Descent(SGD) [17] is used for optimization because the pre-trained parameters utilize ADAM optimizer and have the momentum information implicitly, which mitigates the need for using momentum-based optimizers.

Training NeuMF without the pre-trained GMF and MLP models uses Adaptive Moment Estimation(ADAM)[10] for optimization with randomly initializing parameters from a Gaussian distribution (with a mean of 0 and a standard deviation of 0.01).

4 RESULTS

4.1 Data Description

I use Stack Overflow data dump, which has approximately 1 million users and 19 million posts. However, I selected 2019 posts and considered users who have answered more than 15 questions during the last year as experts because of computational power limitations. Moreover, the expert who has the accepted answer is only counted as the expert, not other answerers due to the low score of the significant portion of answers for the questions which adversely affect expert identification. I split the data randomly, by taking 80% as the training set and 20% for the testing. Furthermore, I applied negative sampling to obtain negative interaction due to a lack of experts who didn't answer the question. Negative sampling[14] is a method used for machine learning models that have numerous negative observations than positive ones. For each question, I randomly sampled 4 experts who didn't answer the question as the negative samples.

4.2 Evaluation Method

To evaluate the performance of the expert recommender, for each question in the test set, 100 negative experts are randomly sampled, and the model rank all users, including the user who has truly answered the question based on prediction. For instance, in the labelling of our data, the expert who answered the question has label 1 and others are zero, so the model should rank it higher than others. Afterwards, I applied rank-based metrics such as Hit Ratio(HR) and Normalized Discounted Cumulative Gain (NDCG)[7] to evaluate the expert recommender. These metrics are calculated for the top 10 rank list. HR indicates whether the test expert is present in the top 10, and NDCG [7] considers the position of the hit by measuring higher scores for the top ranks.

I used this method and metrics so that I can compare it to [8] result. Then, I calculated both metrics for each test user and reported the average score. I conducted this experiment for 20 iterations and chose the model with the best NDCG as the best iteration. Moreover, I explored the performance of NeuMF with both pre-trained and unpre-trained GMF and MLP models.

4.3 Baseline

I used Matrix Factorization method [12] as the baseline for my model, which is the de-facto approach to latent factor model-based recommendation that was used as the baseline for the related paper. This model performance highly relies on the choice of user-interaction function. This method uses direct factorization of the incomplete matrix into low-rank U and V, with an L1 sparsity penalty on the elements of U and an L2 penalty on the elements of V to complete the matrix. I solved the factorization by gradient descent and set the learning rate to 0.001 so that small steps are taken towards the minimum without jumping over it. Furthermore, the number of latent factors is set to default 10, which is recommended by the literature [12]. In this project, NeuMF aims to enhance this model by generalizing this user-interaction function using deep learning in the expert recommendation domain.

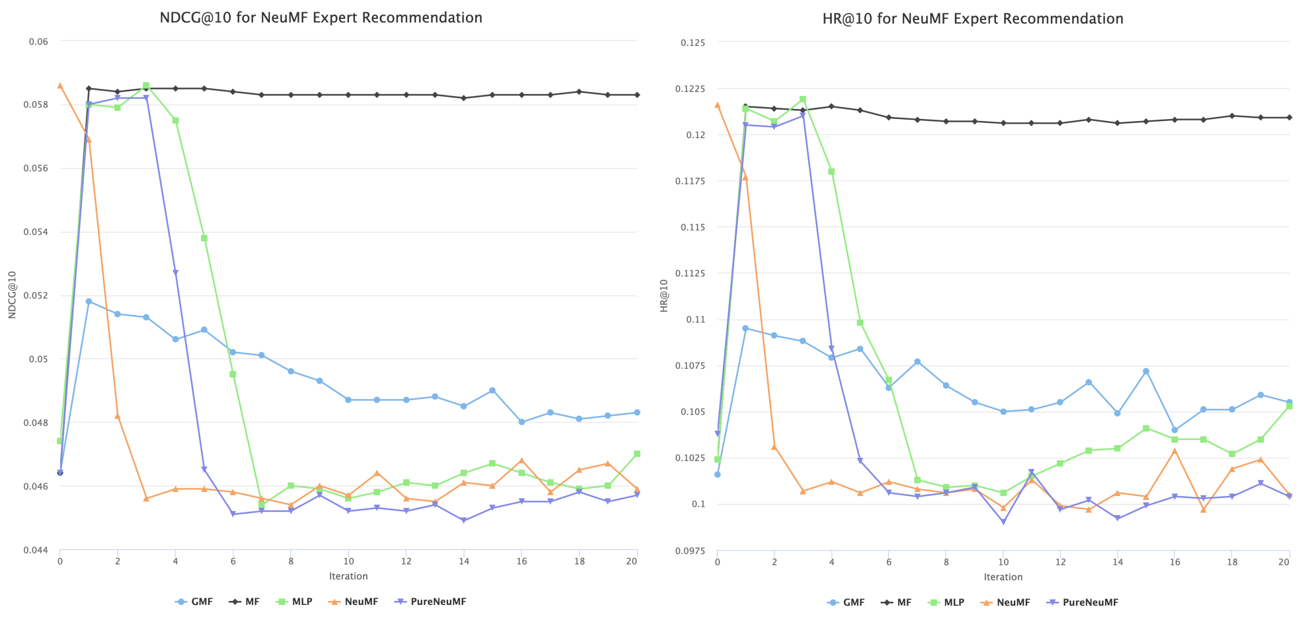

Figure 2: Recommendation Performance of NCF framework in SO

Table 1: Best Performance of Each NCF model

Model	NDCG@10	HR@10
GMF	0.0518	0.1095
MLP	0.0586	0.1219
NueMF	0.0586	0.1216
PureNeuMF	0.0582	0.121
MF	0.0585	0.1215

4.4 Results

Figure 2 demonstrates the performance of the NCF framework in SO. MLP has the best NDCG and HR between the 5 models. Table 1 indicates the best performance of these models. MLP has better performance than MF as the baseline. However, results indicated that GMF is prone to overfitting with the lowest metrics and NueMF with pre-trained parameter initialization has the same performance as MLP. Consequently, these results exhibited that the latent factors have different weights, and expert and question latent are linearly related to each other.

5 DISCUSSION

The results indicated that the linear inner product is sufficient for learning the expert-question interaction data, but these latent features have distinctive weights. Although NueMF had better performance than MF, the enhancement is so minute. Each of MLP and GMF models took four hours to converge using 4 GPUs, NueMF with pre-trained parameters converged in 1 day, and NueMF without pre-trained parameters complete execution in 3 days. Thus, MF only demanded 5 minutes to factorize the interaction matrix. Therefore, the computational overhead of NCF models is substantial without improving the performance dramatically. [2] proposed that many deep learning models propose tiny advancement in performance with thoroughly investigating the best hyper-parameters and compared their models to the normal baseline without tweaking the hyper-parameters for the best performance. In the paper [2], authors wholly investigated the NCF framework and realized that this method didn't have better performance than eALS[9], the state-of-the-art machine learning recommender. In this project, my results projected the same outcome as [2], where NueMF improves MF by only 0.1%.

6 CONCLUSION AND FUTURE WORK

In this project, I investigated the application of the collaborative filtering NCF framework in SO platform. The results projected that latent features were not equally important, which further indicated the need to utilize content-based filtering recommenders in SO to consider textual features of questions and experts' field of interest. Moreover, attention-based neural networks [22] is a probable venue for the future of expert recommenders due to its capacity to identify the critical latent features in the interaction matrix. The resources for this project is available on Github [1].

REFERENCES

[1] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2012. Discovering value from community activity on focused question answering sites. *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 12* (Aug 2012), 850–858. https://doi.org/10.1145/2339530.2339665

[2] Maurizio Ferrari Dacrema, Simone Boglio, Paolo Cremonesi, and Dietmar Jannach. 2019. A Troubling Analysis of Reproducibility and Progress in Recommender Systems Research. arXiv:cs.IR/1911.07698

[3] David Van Dijk, Manos Tsagkias, and Maarten De Rijke. 2015. Early Detection of Topical Expertise in Community Question Answering. *Proceedings of the 38th*

[1]https://github.com/Minam7/NeuMF_SO

International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR 15 (Apr 2015), 995–998. https://doi.org/10.1145/2766462.2767840

[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 *Proceedings of the 24th International Conference on World Wide Web* (Florence, Italy) *(WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 278–288. https://doi.org/10.1145/2736277.2741667

[5] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep Sparse Rectifier Neural Networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*, Geoffrey Gordon, David Dunson, and Miroslav Dudík (Eds.), Vol. 15. PMLR, Fort Lauderdale, FL, USA, 315–323. http://proceedings.mlr.press/v15/glorot11a.html

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. arXiv:cs.CV/1512.03385

[7] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. TriRank: Review-Aware Explainable Recommendation by Modeling Aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management* (Melbourne, Australia) *(CIKM '15)*. Association for Computing Machinery, New York, NY, USA, 1661–1670. https://doi.org/10.1145/2806416.2806504

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web* (Perth, Australia) *(WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 173–182. https://doi.org/10.1145/3038912.3052569

[9] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2017. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. arXiv:cs.IR/1708.05024

[10] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. arXiv:cs.LG/1412.6980

[11] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada, USA) *(KDD '08)*. Association for Computing Machinery, New York, NY, USA, 426–434. https://doi.org/10.1145/1401890.1401944

[12] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37. https://doi.org/10.1109/MC.2009.263

[13] Duen-Ren Liu, Yu-Hsuan Chen, Wei-Chen Kao, and Hsiu-Wen Wang. 2013. Integrating expert profile, reputation and link analysis for expert finding in question-answering websites. *Information Processing & Management* 49, 1 (2013), 312–329. https://doi.org/10.1016/j.ipm.2012.07.002

[14] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*. Curran Associates, Inc., 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[15] Steffen Rendle. 2010. Factorization Machines. In *Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM '10)*. IEEE Computer Society, USA, 995–1000. https://doi.org/10.1109/ICDM.2010.127

[16] Fatemeh Riahi, Zainab Zolaktaf, Mahdi Shafiei, and Evangelos Milios. 2012. Finding expert users in community question answering. *Proceedings of the 21st international conference companion on World Wide Web - WWW 12 Companion* (2012). https://doi.org/10.1145/2187980.2188202

[17] H. Robbins and S. Monro. 1951. A stochastic approximation method. *Annals of Mathematical Statistics* 22 (1951), 400–407.

[18] Hiemstra Djoerd Zaragoza Hugo Rode Henning, Serdyukov Pavel. 2007. Entity ranking on graphs: Studies on expert finding. *CTIT Technical Report Series* (2007).

[19] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-Based Collaborative Filtering Recommendation Algorithms. In *Proceedings of the 10th International Conference on World Wide Web* (Hong Kong, Hong Kong) *(WWW '01)*. Association for Computing Machinery, New York, NY, USA, 285–295. https://doi.org/10.1145/371920.372071

[20] Florian Strub and Jérémie Mary. 2015. Collaborative Filtering with Stacked Denoising AutoEncoders and Sparse Inputs. In *NIPS Workshop on Machine Learning for eCommerce*. Montreal, Canada. https://hal.inria.fr/hal-01256422

[21] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2643–2651. http://papers.nips.cc/paper/5004-deep-content-based-music-recommendation.pdf

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[23] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2014. Collaborative Deep Learning for Recommender Systems. arXiv:cs.LG/1409.2944

[24] Xinxi Wang and Ye Wang. 2014. Improving Content-Based and Hybrid Music Recommendation Using Deep Learning. In *Proceedings of the 22nd ACM International Conference on Multimedia* (Orlando, Florida, USA) *(MM '14)*. Association for Computing Machinery, New York, NY, USA, 627–636. https://doi.org/10.1145/2647868.2654940

[25] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining* (San Francisco, California, USA) *(WSDM '16)*. Association for Computing Machinery, New York, NY, USA, 153–162. https://doi.org/10.1145/2835776.2835837

[26] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 353–362. https://doi.org/10.1145/2939672.2939673

[27] Cai Deng He Xiaofei Yueting Zhuang Zhou Zhao, Qifan Yang. 2016. Expert finding for community-based question answering via ranking metric network learning. *IJCAI International Joint Conference on Artificial Intelligence* (2016), 3000–3006.