Neural Network-Based Collaborative Filtering for Question Sequencing

Lior Sidi 200434447 and Hadar Klein 200623049

Abstract— **E-Learning systems (ELS) and Intelligent Tutoring Systems (ITS) take a major part in today's education programs. Sequencing questions is the art of generating a personalized quiz for a target learner. A personalized test will enrich the learner's experience and will contribute to a more effective and efficient learning process. in this paper we used the Neural Collaborative Filtering (NCF) model to generate question sequencing and compare it to a pair-wise memory-based question sequencing algorithm - Edurank. the NCF model showed significantly better ranking results than the Edurank model with an Average percision correlation score of 0.85 compared to 0.8.**

I. INTRODUCTION

E-Learning systems (ELS) and Intelligent Tutoring Systems (ITS) take a major part in today's education programs. Learners can study from their laptop and gain education from top lectures with Massive Open Online Courses (MOOCs). Learners can access many different educational resources such as lectures, summaries, exercises and exams. Recommender systems (RS) personalize ELS to learners in order to enrich their experience so they could learn more effectively and efficiently while keeping them motivated [KMIN15].

One of the many challenges that RS and ELS deal with is generating a personalized test for a target learner. The main motivation behind personalized tests is to avoid frustration of the learner from too easy or difficult questions under certain context. In order to create a personalized test the RS system should consider the learner's personalized difficulty, capabilities, context, learning styles and habits.

In order to generate a personalized test one must first assess the difficulty level of all the questions. The main article that this paper is based on addresses this issue with personalized sequencing of questions [SKG14]. The next stage of the personalized test is to decide the order of questions that are presented to the target learner.

Personalized sequencing in ELS is the learner's path through a collection of learning objects. Sequencing is an important part of the Sharable Content Object Reference Model (SCORM), an e-learning software product standard, and is applied by different mechanisms such as schedule-based sequencing, artificial intelligence based sequencing, collaborative learning or customized learning [KMVI+17].

In the first section "Related work" we will review different aspects of recommendation systems in E-learning and explain in detail the question sequencing task. We will review the Edurank algorithm, a memory based algorithm for questions sequencing. In the "Method" section we present a novel approach for sequencing questions by using the Neural Collaborative Filtering (NCF) model. In the "Evaluation" section we will examine the different parameters in the NCF network and finally compare the optimized model with the traditional Edurank model using the Algebra dataset [KBCS10]. In the "Results" sections we will present the evaluation results. The NCF got significantly better results then the Edurank with SAP score of 0.85 compare to 0.8 and with positive Spearman's rho score of 0.27. In the "Discussion" section we will interpret the results and in the "Conclusions" section that follows we will point out major conclusions and propose future work.

II. RELATED WORK

A. Recommendation Systems in E-learning

A common approach to generate item rankings is to use Collaborative Filtering (CF) methods. Of these, most methods order items for target users according to their predicted ratings. Thai-Nghe et al, proposed to predict student performance using matrix factorization [TNDH+11]. Their methods address specific latent vectors that explain when a learner is guessing or when he made a true mistake. Furthermore, they used the tensor factorization algorithm to combine the learners knowledge improvement over certain time-context. They applied their algorithms on the KDD 10 dataset [KBCS10] and the results seemed promising compared to other classification methods.

Another CF approach relies on the similarity between item ratings of different users to directly compute the recommended ranking over items. Segal, Ktzir and Gal applied this approach and suggested the personalized pre-ordering of questions by difficulty using CF and social choice methods [SKG14]. A more detailed explanation of their study is presented in the next sub-section.

Wang, Wang and Yeung proposed using collaborative filtering with a deep neural network architecture for recommender systems [HW14]. They combined ratings (sparse data) and auxiliary information such as item content information (can also be sparse) to generate new user-item ratings. Their collaborative deep learning model (CDL), preforms deep representation learning for the content information and collaborative filtering for the ratings (feedback) matrix. The CDL model's recall measure out-preformed other matrix factorization and SVD like methods.

He et al, presented the Neural network based Collaborative Filtering framework (NCF) [HLZ+17]. As apposed to the previous work, they used the ratings information without the content data. They proposed a model based algorithm using two embedding layers of user latent factors and item latent factors. The concatenated product of these two embedding

layers is then used as a the first layer in a deep multi-layer neural architecture ("the neural collaborative filtering layers"), to map the latent vectors to prediction scores. The NCF model needs a lot of training data to be accurate but it can reveal complex latent features (more than the simple SVD model). The NCF architecture can be seen in figure 1.

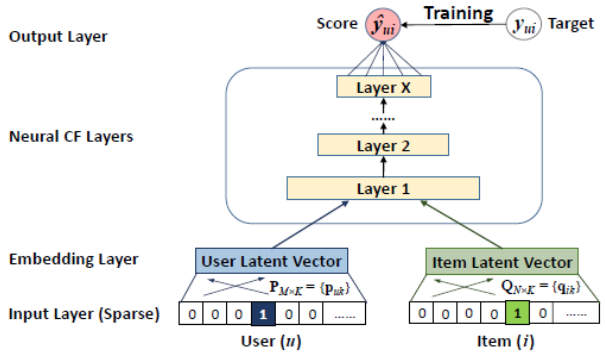

Fig. 1. Neural Network-based Collaborative Filtering Architecture

Cheng et al, presented the Wide and Deep Learning model for Recommender Systems (WDL) [CKH+16]. The WDL model combines the generalization strength of deep neural networks with the memorization of feature interactions (through a wide set of cross-product feature transformations) which are effective and interpretable. The wide component is a generalized linear model while the deep component is a feed-forward neural network. The WDL architecture can be seen in figure 2.

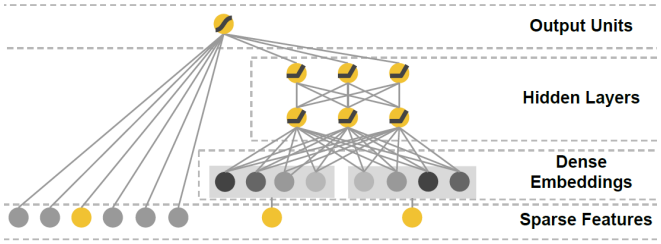

Fig. 2. The Wide (left side) and Deep (right side) model

B. Question Sequencing - The EduRank study

The EduRank study introduces an algorithm that uses collaborative filtering and social choice in order to produce a personalized ranking of new questions sequenced by their difficulty. Therefore, this study addresses the first step in generating a personalized test.The Edurank study presents some key concepts:

- Partial order - Let $\succ_j$ be the partial oder of a set of questions for student j. If $q_n \succ_j q_m$ then q_n is more difficult than q_n for student j.
- NDPM - The Normalized Distance based Performance Measure (NDPM) is used for evaluating a proposed system ranking to a reference ranking. It differentiates between correct orders of pairs, incorrect orders and ties. NDPM is used once in the evaluation of the algorithm [SKG14].
- AP Rank Correlation - the Average Precision correlation metric (AP or SAP) is also used for evaluating a proposed system ranking to a reference ranking but it gives more weight to errors over items that appear at higher positions in the reference ranking. AP is used once in the EduRank algorithm itself and for the second time in the evaluation of the algorithm [SKG14].

The Edurank study used two real world educational datasets:

1) The Algebra 1 dataset that was published in the KDD cup 2010 by the Pittsburgh Science of Learning Center (PSLC) [KBCS10]. This dataset contains about 800,000 answering attempts by 575 students, collected during 2005-2006. The features extracted for each question were: question ID, the number of retries needed to solve the problem by the student, and the duration of time required by the student to submit the answer.
2) The K12 unpublished dataset obtained from an e-learning system installed in 120 schools and used by more than 10,000 students. This dataset contains about 900,000 answering attempts in various topics including mathematics, English as a second language, and social studies. A non-personalized difficulty ranking from 1-5 for each question was supplied by a domain expert. this feature will be referred as CER. The features extracted for each question were: question ID, the answer provided by the student, the associated grade for each attempt to solve the question, CER score, TBR score which is the mastery level of the student on the question's topic.

In the EduRank Algorithm every question was labeled with a difficulty degree comprising of the first attempt grade, the number of retries and for the PLSC dataset the elapsed time solving the question was also considered. The Edurank model is a memory-based algorithm because it needs to save all the users similarities in order to generate a rating, the difficulty score. The EduRank algorithm is presented in algorithm 1.

Algorithm 1: The original EduRank Algorithm

Input : Set of students S.
Set of questions Q.
For each student $s_j \in S$, a partial ranking $\succ_j$ over $T_j \subseteq Q$.
Target student $s_i \in S$.
Set of questions L_i to rank for s_i.

Output: a partial order $\hat{\succ}_i$ over L_i.

foreach $q \in L_i$ **do**
  $c(q) = \sum_{q_l \in L \setminus q} rv(q, q_l, S)$
end
$\hat{\succ}_i \leftarrow \forall (q_k, q_l) \in \binom{L_i}{2}, q_k \hat{\succ}_i q_l$ iff $c(q_k) > c(q_l)$
return $\hat{\succ}_i$

The Edurank algorithm uses the known partial ranking $\succ_j$ for each student s_j over the group of question which that student has answered (T_j). The output of the algorithm is the partial order over the group of questions that a target student (s_i) has not yet answered (L_i). For each question q in L_i a Copland score is calculated ($c(q)$). the Copland score is a representative of the difficulty rank of q in L_i and it is described in equation 1.

$$c(q) = \sum_{q_l \in L \setminus q} rv(q, q_l, S) \tag{1}$$

$rv(q, q_l, S)$ is the aggregated voting of the order of (q, q_l) amongst all of s_i's neighbors. $rv(q, q_l, S)$ is described in equation 2. The aggregated voting can be perceived as a competition between q and q_l.

- q beats q_l if the number of wins of q over q_l computed over all of s_i's neighbors is higher than the number of loses. In this case $rv(q, q_l, S) = 1$.
- If the opposite occurs then $rv(q, q_l, S) = -1$.
- and if the number of wins equals the number of losses then $rv(q, q_l, S) = 0$.

$$rv(q, q_l, S) = sign(\sum_{j=S \setminus i} s_{AP}(T_i, \succ_i, \succ_j) \cdot \gamma(q, q_l, \succ_j) \tag{2}$$

The neighbor j's win or loss of q over q_l is expressed by equation 3

$$\gamma(q, q_l, \succ_j) = \begin{cases} 1 & \text{if } q \succ_j q_l \\ -1 & \text{if } q_l \succ_j q \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Every neighbor's win or loss of q over q_l is normalized by a similarity measure between that neighbor and s_i. The similarity measure is notated as $s_{AP}(T_i, \succ_i, \succ_j)$. It is based on the similarity between s_i and s_j regarding s_i's known partial ranking $\succ_i$ over the group of question which have been already answered (T_i). The similarity score is based on an AP Rank Correlation metric where disagreements over questions that are perceived more difficult are more heavily penalized.

The EduRank algorithm's performance was compared to other ranking algorithms using the NDPM and AP scores. The other algorithms were CER, TBR, a KNN method using the Pearson correlation (denoted UBCF), a matrix factorization method using SVD (denoted SVD), and EigenRank which are all explained in [SKG14]. EduRank outperformed all the other algorithms. The EduRank algorithm is simple and due to its' collaborative filtering feature the execution time is near the UBCF and better than SVD and EigenRank. The EduRank Algorithm does not acquire any user intervention in order to inquire the finalized ranking for the unseen question and keeps that list of unseen question for future use. The EduRank algorithm creates a static ranking for a target student with given known answers. In order to calculate new difficulty scores based on the a student's new answers the whole algorithm is run again and all similarity measures and partial rankings are calculated in he entire dataset. Edurank is a memory based algorithm which makes it hard to use on real world high scaled datasets without compromising run time performance. We propose using deep collaborative methods which are model based to gain more accurate personalized difficulty scores and to speed up the prediction of new difficulty scores based on user interaction.

III. Method - Neural Collaborative Filtering

Neural network based Collaborative Filtering (NCF) captures the user to item interaction with a deep learning model by replacing the inner product of the matrix factorization with a neural network architecture [HLZ+17].

As presented in Figure 1, the NCF is a multilayer network:

1) The inputs are the users and the items as two separate one hot vectors.
2) Each input vector is connected to an embedding layer that functions as a latent factor vector.
3) The embedding layers are then joined together to by concatenation to form the first layer in the neural collaborative filtering layers (stacked layers of weights and neurons).
4) the final layer is connected to one output neuron which predicts the difficulty score.

Algorithm 2: The NCF Algorithm

Input : Set of students S.
Latent factor size k
Number of hidden layers l (depth)
activation functions A
Set of questions Q.
Set of known difficulty scores D
Target student $s_i \in S$.
Set of questions L_i to rank for s_i.

Output: a partial order $\hat{\succ}_i$ over L_i.

$newNCF \leftarrow NCF(A, l, k)$; `// Create NCF model with l hidden layers and the wanted activation functions A.`

$newNCF.Fit(S, Q, D)$; `// Train NCF model on all known difficulty scores D.`

foreach $q \in L_i$ **do**

  $d(i, q) = newNCF.predict(s_i, q)$; `// predict the difficulty score for student i and question q`

end

$\hat{\succ}_i \leftarrow \forall (q_k, q_l) \in \binom{L_i}{2}, q_k \hat{\succ}_i q_l$ iff $d(i, q_k) > d(i, q_l)$

return $\hat{\succ}_i$

As presented in algorithm 2, the network is trained on user-item previous rankings. Then new items for a target user can be ranked. the users are students and the items are questions. The user-item ratings are the personalized questions difficulty scores.

In order to use the NCF model for question sequencing we train the network on the students' previous answers and

their pre-calculated user difficulty score. For ranking new questions we apply the model on the user and questions we wish to rank and get the predicted difficulty rank for each question. In the finale step, questions are sorted by their difficulty and presented to the student.

One of the most important aspect in the NCF model is the architecture configuration such as activation function, the latent factor size (width), the amount of the CF layer stacking (depth). In the evaluation section we present the model parameters importance and finally compare the optimized model with the Edurank memory-based algorithm.

IV. Evaluation

We implemented the Edurank model and the NCF model in python 2.7 using the pandas and Keras libraries. we used a standard laptop to run the experiment: Dell Ultrabook, Intel i7-6600U 2.6GHz, 16GB ram. We evaluated the models based on the same dataset used in the Edurank study, the Algebra 1 dataset. For the evaluation stage we filtered 250,000 question's answers and evaluate the question sequencing algorithms on 4 random questionnaires from 3 different users.

In this paper we conducted two types of experiments and evaluations:

- NCF parameter tuning:
  1) Wide vs. Deep - The Wide architecture comprises of the concatenated embedding layers and an output neuron. The Deep architecture comprises of l neural collaborative filtering layers in between the concatenated embedding layers and the output neuron. We tried $l = 1, 2, 4, 8$.
  2) Embedding size k - We evaluated k between 20 and 80.
  3) Activation function A - We evaluated $A = tanh, linear, relu$.
- Evaluation of the optimized NFC ranking vs. the Edurank ranking. We compared the Edurank algorithm with memory size of the 5 most similar students to an optimized NCF algorithm. We used The true difficulty ranking as the referenced ranking and compared the Edurank and NCF to it. The evaluation Metrics used were the Average correlation score (SAP) and the Spearman's rho score (SR). The SAP was used in the evaluation of the Edurank algorithm [SKG14]. The Spearman's rho score is a statistical correlation measure between two ranked variables [SG11]. Spearman's rho value is a continues value between 1 and -1 where 1 is a perfect ranking correlation between the predicted ranking and the referenced ranking. The Spearman's rho calculation can be seen in equation 4 where d_i is the difference between the two ranks of each observation an n is the number of observations.

$$SR = 1 - \frac{6 \sum_i d_i^2}{n(n^2 - 1)} \tag{4}$$

V. Results

A. NCF tunning

We started with evaluating the NCF architecture parameters. We set the architecture with fixed configurations of one epoch, 1024 batch size and 0.25 for dropout rate. the training of the model is optimized with 'Adadelta' optimizer with a mean square error loss function.

In Figure 3 we present the parameters tuning results and evaluation. The X axis is the parameter value we evaluated and the Y axis is the ranking results (Spearman's rho and SAP), In the two bottom graphs there is also a gray line that represent the training duration per value.

The top graph evaluates different activation functions with $l = 4$ and $k = 32$. As presented in the graph, the 'relu' and 'tanh' activation functions (SAP=0.825, SR=0.1) outperform the simple 'linear' function (SAP=0.822,SR=0.014). We choose the 'tanh' for the rest of the evaluation due to it's slightly higher AP rank and shorter fit and rate duration.

Fig. 3. NCF Parameters Tuning

The middle graph evaluates k, the factoring size in the embedding layer (width sizes). As expected, higher k values correlate with better ranking. The training duration also increases substantially when increasing the factor size k, therefore we choose a cost effective size of $k = 40$ that combines performance (low training duration) and high ranking results.

The bottom graph evaluates the network depth l, the number of stacked CF layers. Surprisingly deeper networks with more then one layer had lower Spearman's rho score and couldn't breach the 0.84 SAP score. The wide network with no CF layers (a simple factorized model) showed the worst results with only 0.02 Spearman's rho score. Therefore, the usage of at least one CF layer is necessary.

To conclude, the final optimized NCF network parameters are $k = 40$, $l = 1$, and 'tanh' as the activation function.

B. NCF and Edurank comparison

We compared the optimized CNF model with the known Edurank model. as demonstrated in Figure 4, the CNF model outperform the Edurank model with mean 0.86 SAP score and 0.27 Spearman rho compared to 0.81 and -0.22 in the Edurank algorithm.

Fig. 4. rank results for Edurank and the CNF algorithms

To test the significance of the results we used pairwise t-test on the student-questionnaire results (total of 12 cases, 3 students per 4 questionnaires). in Figure 5 we present the tests results. In both tests for the Spearman's rho and SAP score the t-statistic is bigger then the absolute value of the t-critical and the p-value is less than 0.005 meaning that we reject the null hypothesis and the NCF rankings are significantly better then the Edurank rankings.

t-Test: Paired Two Sample for spearman Means

	Edurank rho	*NCF rho*
Mean	-0.21904219	0.2652495
Variance	0.014594317	0.0232317
Observations	12	12
Pearson Correlation	-0.54352263	
Hypothesized Mean Difference	0	
df	11	
t Stat	-6.97550395	
P(T<=t) one-tail	1.1717E-05	
t Critical one-tail	1.795884819	
P(T<=t) two-tail	2.34341E-05	
t Critical two-tail	2.20098516	

t-Test: Paired Two Sample for SAP Means

	Edurank SAP	*NCF SAP*
Mean	0.809667732	0.8586214
Variance	0.002625296	0.0023083
Observations	12	12
Pearson Correlation	0.841704279	
Hypothesized Mean Difference	0	
df	11	
t Stat	-6.03514054	
P(T<=t) one-tail	4.24378E-05	
t Critical one-tail	1.795884819	
P(T<=t) two-tail	8.48756E-05	
t Critical two-tail	2.20098516	

Fig. 5. Pairwise t-test results between Edurank and NCF ranking results

VI. Discussion

The optimized NCF model scored higher (SAP=0.86, SR=0.27) than the Edurank model (SAP=0.81, SR= -0.22). Deep learning showed good results compare to the memory based algorithm under real world circumstances where the memory size of the model is limited. Therefore we recommend the optimized NCF model to evaluate unseen personalized question difficulty scores.

Evaluating deep learning architectures for a given dataset is an endless game due to the fact that there could be many different possibilities to design the architecture and train the network. To make matters more complicated, the architecture could change according to the dataset. In our evaluations we addressed the most critical parameters that effect the NCF network: the the embedding size (k, "width") and the number of hidden CF layers (l, "depth"). We encountered a trade of between increasing the width and gaining better ranking scores on one hand and decreasing the performance (longer training time) on the other hand. We chose a cost effective k value that considers both performance and ranking scores. Regarding the depth size we were surprised to see a decrease in the ranking scores as the number of hidden layers, l, grew. This result might have been a side effect of the vanishing gradient syndrome of deep neural networks. We propose more experiments with different activation functions (such as relu) to test this theory.

The Edurank originally was implemented in Java Mahout (map reduce), the implementation in this paper ran locally with python and also demanded many code optimization and parallelization to work in reasonable throughput.

One of the main advantages of the NCF on the memory-based algorithm is that it supports retraining efficiently. This advantage will allow deploying the NCF model in a real world environment that can be retrained regularly.

VII. Conclusion

In this paper we used a neural collaborative filtering (NCF) model to sequence questions by difficulty and optimized it accordingly. We compared the NCF model with with the Edurank algorithm with limited memory size, a memory based algorithm for question sequencing. the NCF model

showed significantly better results in ranking questions than the Edurank algorithm.

The NCF has two additional advantages over the Edurank memory based algorithm. First, the NCF model has the ability to represent complicated connections between users and items. Second, It can be adaptively retrained by the user's feedback to tune the question sequencing. The retraining of the NCF model can be done in batches and requires less computational complexity than the Edurank model.

We believe the next natural step should be to investigate the combination of a deep and wide architecture as proposed in [CKH+16].There is much more to investigate and research in this domain. It would be interesting to use different kinds of architectures such as recurrent neural network to recommend a future sequence of questions. It would also be interesting to add content based features to the entire network and check any influence on the rating.

As mentioned, it is possible to use the NCF as a part of an on-line experiment where the model can be retrained on the users feedback, it will be interesting to examine the online model and compare it to the offline one.

REFERENCES

[CKH+16] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & Deep Learning for Recommender Systems. *arXiv preprint*, 2016.

[HLZ+17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural Collaborative Filtering. *Www*, 2017.

[HW14] Dit-Yan Yeung Hao Wang, Naiyan Wang. Collaborative Deep Learning for Recommender Systems. *Arxiv*, 2014.

[KBCS10] Kenneth R Koedinger, Ryan S J Baker, Kyle Cunningham, and Alida Skogsholm. A Data Repository for the EDM community : The PSLC DataShop. *Handbook of Educational Data Mining*, 2010.

[KMIN15] Aleksandra Klašnja-Milićević, Mirjana Ivanović, and Alexandros Nanopoulos. Recommender systems in e-learning environments: a survey of the state-of-the-art and possible extensions. *Artificial Intelligence Review*, 2015.

[KMVI+17] Aleksandra Klašnja-Milićević, Boban Vesin, Mirjana Ivanović, Zoran Budimac, and Lakhmi C. Jain. *E-Learning Systems*, volume 112 of *Intelligent Systems Reference Library*. Springer International Publishing, Cham, 2017.

[SG11] Guy Shani and Asela Gunawardana. Evaluating recommendation systems. *Recommender systems handbook*, 2011.

[SKG14] Avi Segal, Ziv Katzir, and Kobi Gal. EduRank : A Collaborative Filtering Approach to Personalization in E-learning. *Proceedings of the 7th International Conference on Educational Data Mining*, 2014.

[TNDH+11] Nguyen Thai-Nghe, Lucas Drumond, Toms Horváth, Alexandros Nanopoulos, and Lars Schmidt-Thieme. Matrix and Tensor Factorization for Predicting Student Performance. In *Csedu (1)*, 2011.