A Content-Based Recommendation System Using Neuro-Fuzzy Approach

1st Tomasz Rutkowski
Senfino Technologies
Warsaw, Poland
tomasz.rutkowski@senfino.com

2nd Jakub Romanowski
Senfino Technologies
Czestochowa, Poland
jakub.romanowski@senfino.com

3rd Piotr Woldan
Senfino Technologies
Czestochowa, Poland
piotr.woldan@senfino.com

4th Paweł Staszewski
Senfino Technologies
Czestochowa, Poland
pawel.staszewski@senfino.com

5th Radosław Nielek
Polish-Japanese Academy of Information Technology
Warsaw, Poland
nielek@pjwstk.edu.pl

6th Leszek Rutkowski, Fellow, IEEE
Institute of Computational Intelligence
Czestochowa University of Technology
Czestochowa, Poland,
and Information Technology Institute,
University of Social Sciences,
90-113 Lodz, Poland
leszek.rutkowski@iisi.pcz.pl

***Abstract*—In this paper, we present our novel approach to recommender systems based on a neuro-fuzzy approach. The neuro-fuzzy approach allows for deciding to recommend or not to recommend processed items for a user. By using it, we can understand the decision through analyzing rules of decision paths. Our method gives a possibility to learn and simulate users decisions based on their actions in our test environment. Finally, a rank list of top-rated items is delivered to the user based on simulated rank for each of them. We develop our AI framework to perform tests with the use of CUDA technology. Additionally, we develop a user interface in the form of a web application. It gives the possibility to perform simulations of real users. To compare our approach with a deep learning based method, we perform tests on the MovieLens 20M Dataset. It should be noted that the architecture of the data module of our system allowed for reasonably easy integration with MovieLens data.**

***Index Terms*—recommendation systems, neuro-fuzzy structure, MovieLens Dataset**

I. Introduction

In the world of multimedia and the growing amount of electronic services, the choice of suitable products has become a troublesome and time-consuming issue. Due to the high availability of products or services, choosing a movie to watch, a mobile phone or a car that meets all of the requirements consumes a considerable amount of time. As a result, time becomes the greatest value for a human being; moreover, its saving gains an appropriate value. The solution to this problem became recommendation systems. An appropriately tailored recommendation system for a specific industry allows for presenting an offer corresponding to the user preferences. In recent years, many recommendation systems for multimedia such as music, photos or films have been created [1] [2] [3]. Recommendation systems have touched on other spheres of life like corporate media or the scientific world [4] [5] [6]. Such systems are becoming ubiquitous in every area of life. The preferences and values resulting from the users behavior in the system allow for classifying him to a corresponding user model. Despite information about the users preferences and his context in a given recommendation system, one of the most valuable information is the rating of the presented product. The rating gives information to the system about how much the presented product is appropriate for a given user and should be treated as having an explicit impact [7].

A recommendation system is any system that offers items in a personalized way to a specific user or guides him to the product best suited to his profile [8] [9] [10]. In the literature several techniques for building recommendations systems have been developed including:

- collaborative filtering - it is the most commonly implemented technique [11]. Such systems recommend items by identifying other users with similar taste. Recommendation of new items is based on user-user, user-item, or item-item similarities. The major problem with this technique is known under the name cold start - the system cannot draw any inferences for users or items about which it has not collected enough information.
- content-based techniques - the recommender attempts to recommend items similar to those a given user preferred in the past [12]. In this case, the recommendations are based on information on the content of a given item, not on other users' opinion as in the case of collaborative filtering. These techniques are vulnerable to overfitting, but their great advantage is not needed for data on other users [13].

This work is partially supported by Polish National Science Centre grant 2015/19/B/ST6/03179

978-1-5090-6020-7/18/$31.00 ©2018 IEEE

- hybrid approach - it relies on a combination of many different recommendation methods [14]. The final goal of this approach is to obtain the most accurate list of predictions and as a result, a more precise specification of the user's profile.

To improve the effectiveness of recommendation systems, techniques based on complementing the thematic information data sets from external databases are often used. It allows for a more accurate perception of the problem. An additional knowledge database may contain information from websites, external APIs [15] [16], data from charts, information collected from experts, statistical data [17], etc.

One way to facilitate the recommendation process is to properly model input data. Preparation of data in the appropriate format can significantly improve the processing in the recommendation system. The most commonly used methods are described in [18] and [19].

Without any doubt, almost all recommendation systems developed in the literature are focused on providing the highest accuracy of prediction. Consequently, recommender systems sacrifice their interpretability in favor of a correct rating. However, in many situations, we would like to understand why a certain recommendation has been made.

Motivated by the above discussion, this paper presents a new content-based recommendation system developed by implementing a neuro-fuzzy approach leading to interpretable recommenders. By properly performing pre-processing of lexical data (e.g. movie genres) and transforming them into values of linguistic variables, we can construct a neuro-fuzzy system.

The initial fuzzy rules are generating using the Wang-Mendel method [20], which divides the input space into fuzzy regions and then applies a table-lookup scheme to extract interpretable rules. The resulting system is fine-tuned, using the back propagation method [21]. Parallely, we have also developed our implementation of a deep-learning content-based recommender system characterized by a very high accuracy but lack of interpretability. The presented comparison results surprisingly show that a not-interpretable deep-learning approach only slightly outperforms the interpretable neuro-fuzzy approach. To our best knowledge, the method presented in this paper is the first successful implementation of neuro-fuzzy techniques in content-based recommender systems.

Although this paper is concerned with a specific application dealing with the MovieLens data, it has a great potential to solve other problems, in particular when interpretability is a very important matter. The remainder of this paper is organized as follows. In Section 2, the recommender system is described in details, including description of the neuro-fuzzy structure, subsequent steps of the preprocessing stage and the rule generation process. In Section 3, numerical simulations are described to demonstrate the effectiveness of our approach. Finally, conclusions are drawn in Section 4.

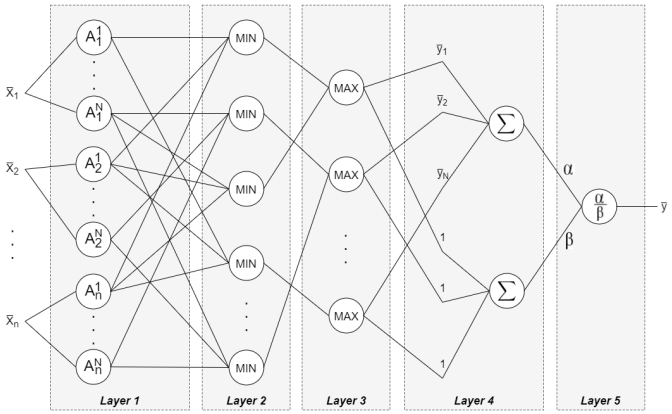

Fig. 1. The neuro-fuzzy structure

II. Neuro-fuzzy recommender system

A. Fuzzy system

In fuzzy systems, the rules base, see e.g. [22], sometimes called a linguistic model, is a set of fuzzy rules $R^{(k)}$, for $k = 1, ..., N$, of the form

$$R^{(k)} : \textbf{IF } x_1 \text{ is } A_1^k \textbf{ AND } x_2 \text{ is } A_2^k \textbf{ AND...AND } x_n \text{ is } A_n^k$$
$$\textbf{THEN } y_1 \text{ is } B_1^k \textbf{ AND } y_2 \text{ is } B_2^k \textbf{ AND...AND } y_m \text{ is } B_m^k$$

where N is the number of fuzzy rules, A_i^k - fuzzy sets such as

$$A_i^k \subseteq \mathbf{X}_i \subset \mathbf{R}, i = 1, ..., n,$$

B_j^k - fuzzy sets such as

$$B_j^k \subseteq \mathbf{Y}_j \subset \mathbf{R}, j = 1, ..., m,$$

$x_1, x_2, ..., x_n$ - input variables of the linguistic model, while

$$[x_1, x_2, ..., x_n]^T = \mathbf{x} \in \mathbf{X}_1 \times \mathbf{X}_2 \times ... \times \mathbf{X}_n,$$

$y_1, y_2, ..., y_m$ - output variables of the linguistic model, while

$$[y_1, y_2, ..., y_m]^T = \mathbf{y} \in \mathbf{Y}_1 \times \mathbf{Y}_2 \times ... \times \mathbf{Y}_m.$$

Symbols $\mathbf{X}_i, i = 1, ..., n$ and $\mathbf{Y}_j, j = 1, ..., m$, denote the spaces of input and output variables respectively. We assume that particular rules $R^{(k)}$, $k = 1, ..., N$, are related to each other using the "or" logical operator. Moreover, in this paper we assume that the output fuzzy sets are singletons. In the paper fuzzy rules will be generated using a slight modification of the Mendel-Wang method, see e.g. [21]. An exemplary scheme of the neuro-fuzzy structure is shown in Fig. 1. For a detailed description of various structures of neuro-fuzzy systems the reader is referred to [22].

B. Preprocessing

The first of the neuro-fuzzy approach issues that we challenged was the description of nominal and numerical variables in a unified manner. The major problem was how to transform lexical values of nominal features, e.g., movies genres, to be incorporated into the neuro-fuzzy structure design. To be

properly processed, the input data of the neuro-fuzzy system should be values of linguistic variables. Numeric values are normalized to the range 0-1.

In cases of nominal features, we analyze ranked samples of the user and based on this, we calculate the rate of importance for individual features. The example of transforming nominal values into a numerical form is presented in Figs. 2 - 4, based on the genres of films rated by one user. This stage of sample processing is repeated for each nominal feature and each user. Generated importances w, for each movie category, are base values needed to generate an input signal to the system. This stage contains the following steps:

1) Counting distinct genres relative to their positions in negative and positive ranked samples. In this example, we analyze occurrences of the genre up to the 3rd place, see Fig. 2.

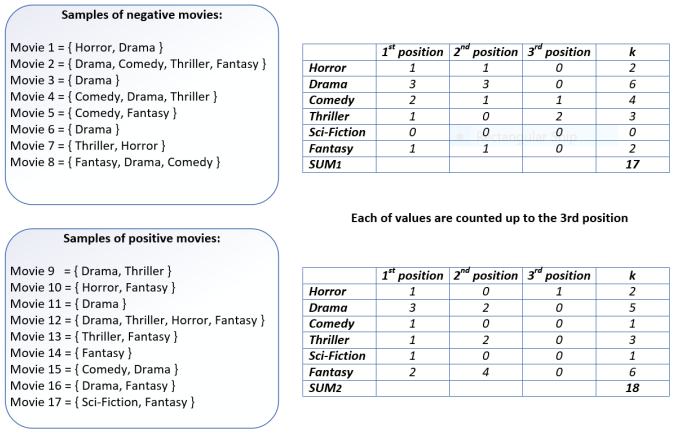

Fig. 2. Counting distinct genres from negative and positive ranked movies

2) Calculating weights for individual genres occurring in positive and negative rated movies, see Fig. 3.
3) Generating the input value to the neuro-fuzzy system based on the calculated weights and nominal values of a movie, see Fig. 4.

C. Generation of fuzzy rules

After data preprocessing, there are several stages to build neuro-fuzzy rules and to learn the system. For each linguistic variable, there are prepared fuzzy sets with a location determined by the occurrence of variables. The domain intervals are bounded by minimum and maximum of linguistic variables values. If we consider the example of movies taking into account movie release year and genres (see Fig.5), and a decision to recommend it or not, there are several steps of building fuzzy sets and rules:

1) The fuzzy sets building process is illustrated in the example of movie release year and genres, as it is shown in Fig. 6. In this example, the minimum year of movie release is 1910, and the maximum is 2010. In the case of genres values, the range is from 0 to 1, as it is explained in point B, see also Figs. 3 and 4.

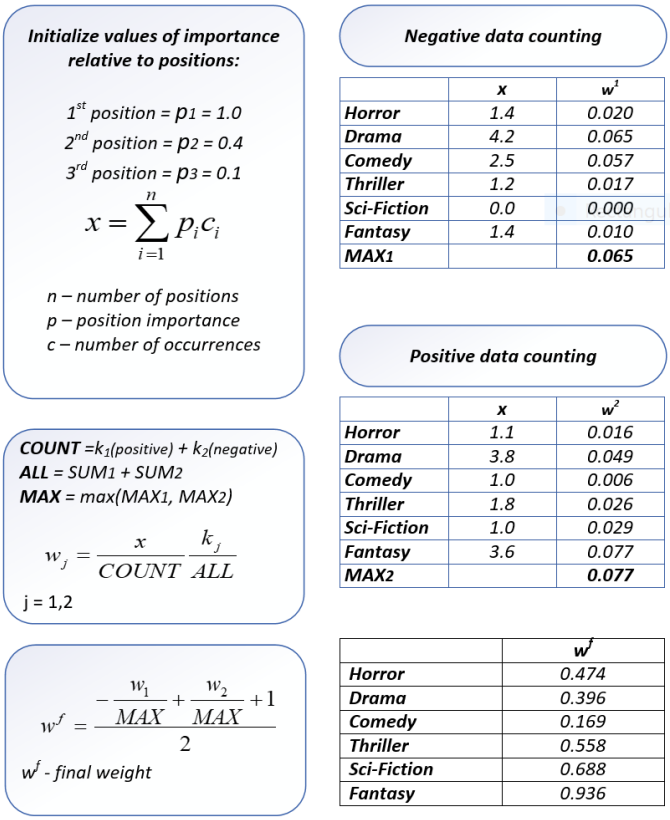

Fig. 3. Calculating weights

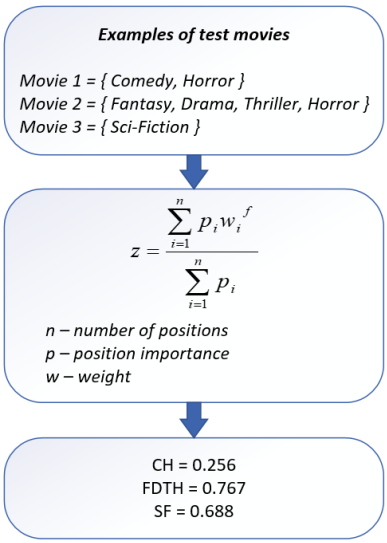

Fig. 4. Generating the input value to the neuro-fuzzy

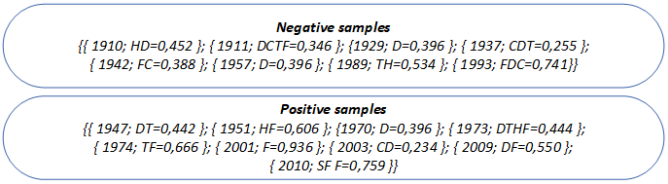

Fig. 5. Exemplary samples for movie recommender

This example presents only negative recommendations to assure readability of the figure. To build fuzzy sets for this linguistic values, there is a need to set the center of the first (1910) fuzzy set in minimal value and the second (2010) in maximal value. The distance between minimal and maximal linguistic value is a basis for preparing the locations of centers of other fuzzy sets by dividing it, e.g. into 11 fuzzy sets. Each of the fuzzy sets is tested by linguistic variable values and fuzzy sets for which the membership function value of a given variable is lower than the threshold (near 0) are excluded. This process is illustrated in Fig. 6 assuming Gaussian membership functions.

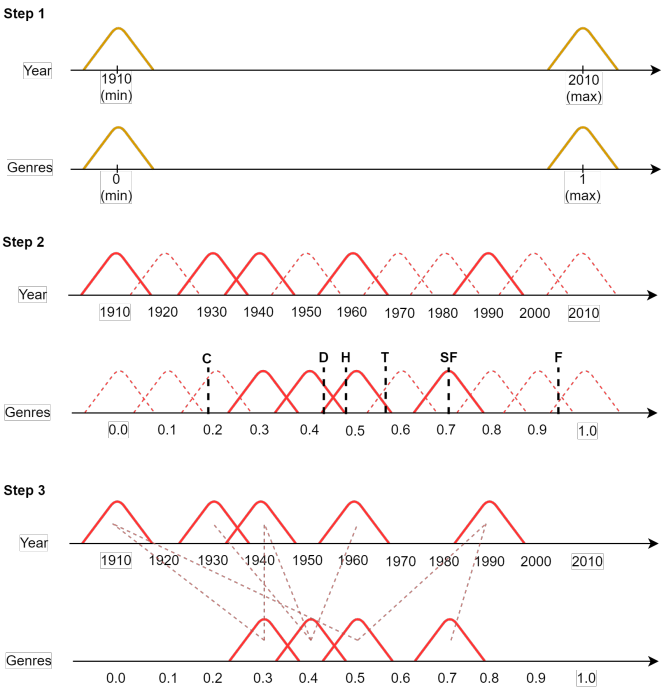

Fig. 6. Visualization of linguistic variable fuzzy sets building process. In tests environment, each fuzzy set is represented by Gaussian curves

2) The previous step should be executed for positive recommendations. In the final result, negative and positive recommendations for movie release year represented by appropriate fuzzy sets should be presented on the same linguistic value axis.
For both positive and negative recommendations, each of the fuzzy sets are prepared for the same linguistic value. Their colors are visual representations of positive and negative cases, see Fig. 7. This figure presents the final stage of building fuzzy sets and rules showing two linguistic variables and singletons describing the recommendation. The procedure described above is a slight modification of the Mendel-Wang algorithm [20], originally developed for function approximation problems.

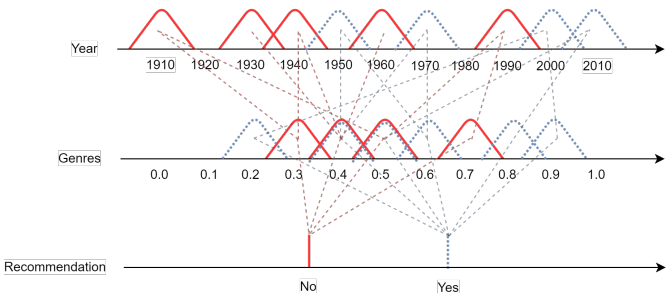

Fig. 7. Example of fuzzy-system with linguistic variables, rules between them and final result in form of decision to recommend movie or not

III. Experimental results

A. Our AI framework for recommender systems

To perform tests, we developed our own AI framework. The framework was developed in C++ programming language using the NVidia technology called CUDA (Compute Unified Device Architecture). CUDA allows us to develop much faster algorithms by using GPU units comparing to traditional C++ with CPU-only calculations. The technologies which are used in this part of the test environment are:

- CUDA Toolkit v9.0.176
- CUDNN v7
- BOOST v1.65.1 (vectors and logging features)

All of the data required to test are stored in the MS SQL 2016 database in the form of groups of features and features. All items which could be recommended are described by attributes and their values.

Our test environment is based on the components described above. To perform the tests the following conditions are required:

- User account
- This user should perform onboarding in the form of a specified amount of assessment of items prepared for him
- User can set his profile preferences – which groups of features and features he likes or dislikes

After this process, our AI framework is able to start simulations. Basing on the user's onboarding ranked items and preferences, the AI framework learns his rankings prediction. The AI framework module then creates the simulated rank for each of possible items. In the next step, a list of top recommendations is prepared for the selected user. The list is sorted in descending order by simulated rank value.

B. Data sets

Because of the need to compare our experimental results with other methods, we decide to perform tests based on the commonly known MovieLens database. It gives us the possibility to benchmark and test our software and algorithms on a significant amount of data. By using this data set, we can test the accuracy and performance of our algorithms. This data set is MovieLens 20M. The dataset is based on information

from MovieLens website, which is a non-commercial, personalized movie recommendations service. This dataset describes movies by features like genres, year, tags and 5-star rating. Additionally, there are external services movies IDs available, e.g. (IMDB, TMDB). External movie IDs could be used to connect with external services to obtain additional information about movies, e.g., casts, keywords, directors, writers and main stars [23].

The dataset contains 27278 movies, 20000263 ratings and 465564 tag applications and the first release of the database was in 1998. The database was created by 138493 randomly selected users, each of them rated from several dozens to several hundred movies. Users are represented only by ID without any sensitive information [23].

In our work, MovieLens 20M is the main data source for movies; however, we decided to obtain much more information from the TMDB API service. As a result, the MovieLens dataset is extended by data from the TMDB API service, and it allows obtaining much more efficient and much more accurate results in the field of movies recommendation. As additional data, apart from the genre of the movie, we take into account such data as actors, contributor, description, keywords, and spoken language. This allows us to build a better user model based on his movie ratings. It should be noted that in the vast majority of previous approaches, the authors used the MovieLens 100K Dataset. In our simulations, the sizeable MovieLens 20M was used to increase trustworthiness and thus, our ability to draw conclusions. In the next two subsections, we present a simulation result which can be a reference point for researchers performing experiments with the MovieLens 20M dataset.

C. Neuro-fuzzy based recommender system

In this subsection, we present simulations performed with the neuro-fuzzy structure described in the previous subsections. As the evaluation criterion, we use the Mean Absolute Error (MAE) given by the following equation

$$\frac{1}{n}\sum_{i=1}^{n}|y_i - z_i| \tag{1}$$

where n denotes a number of samples, and y_i and z_i stand for predicted by our system and expected output, respectively. This criterion was used to determine the effectiveness of 10 sample users.

At the testing stage, a recommendation score greater than the threshold (in this simulation equal to 3,5 in the scale 1-5 of possible movie ratings) indicates a positive recommendation, while a score below the threshold, a position that should not be recommended. The results of the testing phase are shown in Table I.

Figure 8 presents the functioning of our system for two linguistic variables (genres and actors). In Fig. 8a, fuzzy sets, and consequently fuzzy rules, are built on the basis of the method illustrated in Fig. 6, with initial fuzzy singletons at the points 1,0 and 2,0. In Fig. 8b, we depicted fuzzy sets

TABLE I
NEURO-FUZZY MODULE EFFICIENCY FOR 10 USERS, MEASURED BY VERIFYING IF ITEM WAS RECOMMENDED PROPERLY (YES OR NO VALUES)

MovieLens User ID	Recognized (%)
100005	100%
100010	100%
100013	96,46%
100028	97,97%
100034	95,22%
100071	100%
100086	100%
100130	100%
100153	100%
100168	100%
TOTAL	**98,97%**

obtained after a training using the back propagation algorithm. Fig. 8c shows the results of the fuzzy inference for the selected user, for whom, using the method illustrated in Figs. 3 and 4, the inputs of the recommender are equal to 0,498 (genres) and 0,453 (actors). On this figure, we indicated two values of fuzzy singletons at the points 0,97 and 2,04, leading to the final defuzzyfied (crisp) output equal to 1,16. As the threshold for a positive recommendation is 1,5, this specific item is not recommended for the user. The MAE for the neuro-fuzzy recommender is presented in Fig. 9.

D. Deep learning-based recommender system

The Convolutional Neural Network (CNN), see e.g. [24], is one of the most well-known structures in deep learning. This type of AI architecture is used especially in the field of image processing [25]. The CNN can also be used in natural language processing [26] and in many other applications. In this subsection, our implementation of the CNN in recommender system is presented. Our software based on deep learning structure is prepared for gaining knowledge about the way how to rank movies by the each user. These ranks are collected and grouped for different users, hence training data is prepared. The system based on Deep Neural Network mechanisms, analyses such movie attributes as: genres, casts, directors, production companies and keywords. Before data can be inputted to the neural network, one-hot encoding preprocessing has to be done, see e.g. [27]. In this way, we collect every ranked movie for each user. Next, we encode every attribute from these samples and return vectors with distinct values. As a result, training samples are determined. After this phase, coding the testing database was performed. It is done by comparing the attributes vectors used for training with attributes in each group. In the first stage – propagating signal forward in the neural network – the features extraction is performed. It is done by using the CNN, which structure is presented in Table II. The main goal to use CNN is to extract features on different levels from encoded form.

The Multi-Layer Perceptron (MLP), connected to the CNN structure, processes attributes vectors and calculates values of the regression function as the output. Hence, the ReLU acti-

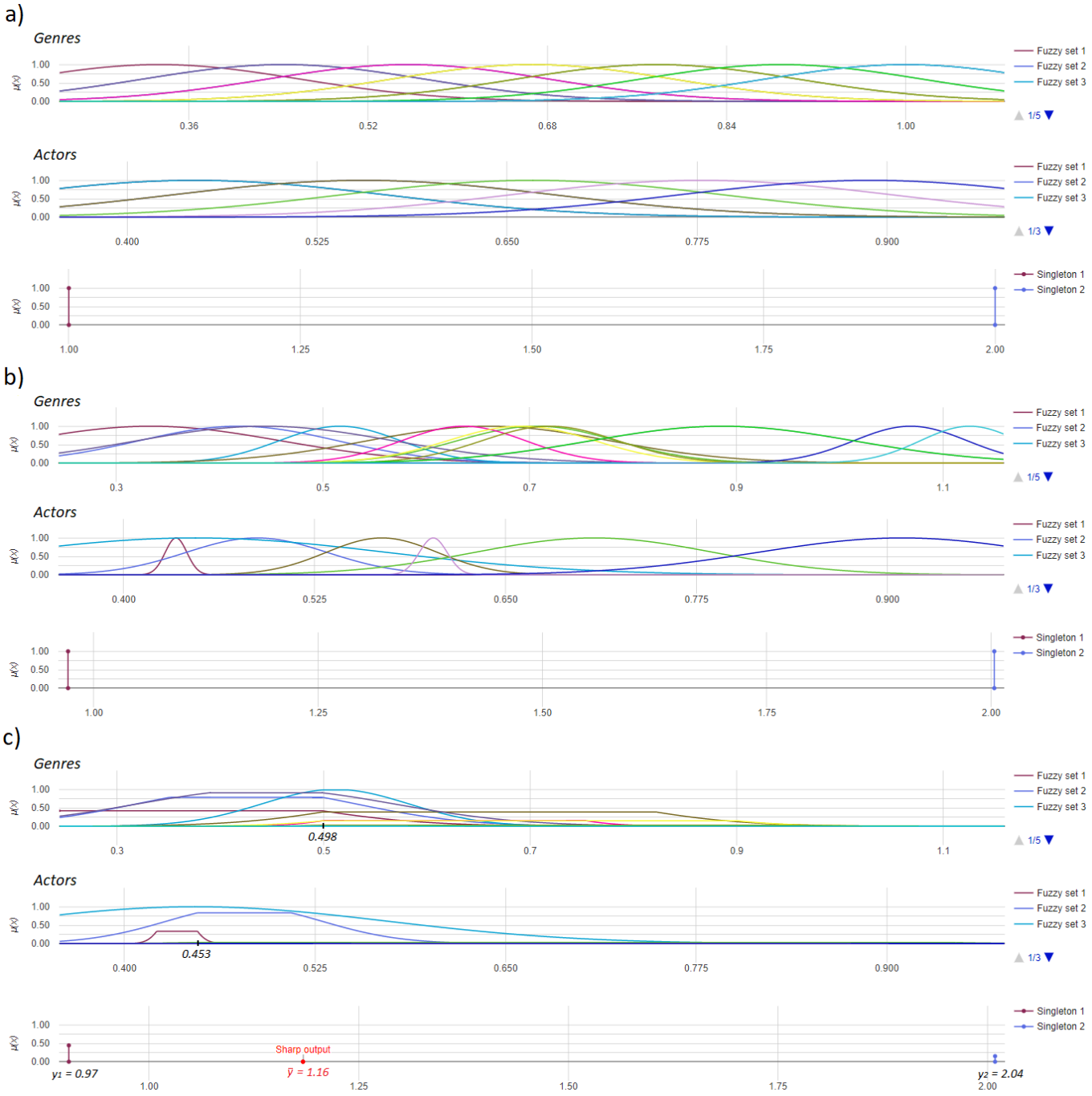

Fig. 8. Fuzzy inference in the recommender system

vation function is used. Architecture of the MLP is presented in the Table III.

The entire CNN + MLP architecture was trained with learning rate starting from 0.001. Learning rate was decreased during training phase by dividing by ten after the 50th epoch. The process ends after reaching of the 110th epoch. We used the same formula as previously, see equation (1), for measuring the effectiveness of the recommender. In this case, if the rank value (a number in the interval 1-5) is predicted with the error $|y_i - z_i| < 0.1$ then the recommendation is treated as positive. The decrease of the MAE is presented in Fig. 10.

The DNN module was designed to determine the rating for recommended items. Finally, this solution is used to build a ranking list. Table IV presents the efficiency of DNN module.

To measure the efficiency of our approach, we select ranks for 10 random users. Then we apply the cross-validation method and get the results, which are shown in Table IV.

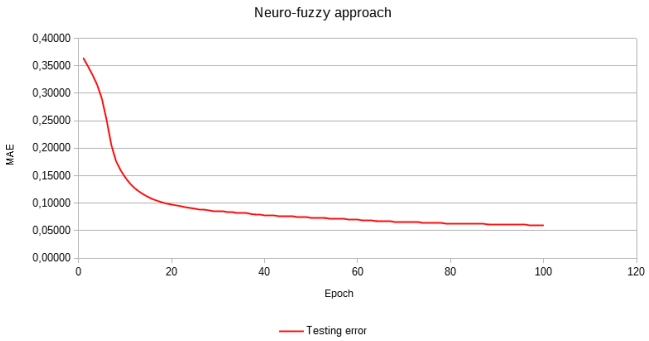

Fig. 9. MAE for the neuro-fuzzy system

TABLE II
STRUCTURE OF THE CNN USED FOR FEATURE EXTRACTION

	Layer type	Parameters
1	Convolutional	*channels: 32, kernel: 1x7, stride: 1x5, z-padding*
2	Convolutional	*channels: 64, kernel: 1x5, stride: 1x2, z-padding*
3	Pooling	*window: 1x3, stride: 1x1, type: avg with padding*
4	Activation	*type: ReLU*
5	Convolutional	*channels: 32, kernel: 1x3, stride: 1x1*
6	Convolutional	*channels: 32, kernel: 1x5, stride: 1x1*
7	Pooling	*window: 1x5, stride: 1x2, type: max*
8	Activation	*type: ReLU*

TABLE III
STRUCTURE OF THE MLP USED FOR REGRESSION

	Layer type	Parameters
1	Fully connected	*neurons: 50*
2	Activation	*type: ReLU*
3	Fully connected	*neurons: 10*
4	Activation	*type: ReLU*
5	Fully connected	*neurons: 1*
6	Activation	*type: ReLU*

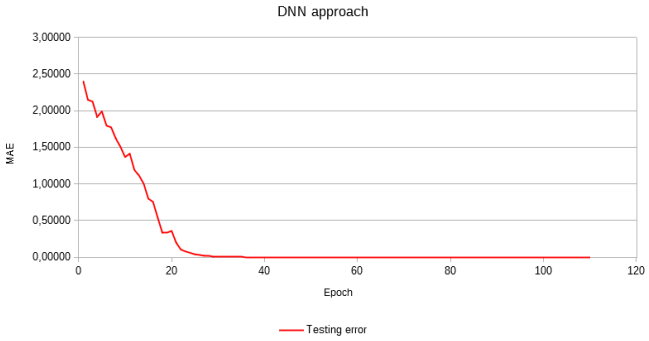

Fig. 10. MAE for DNN approach

TABLE IV
DNN MODULE EFFICIENCY FOR 10 USERS, MEASURED BY VERIFYING CORRECTNESS OF RANK PREDICTION (VALUES IN RANGE 1-5)

MovieLens User ID	Recognized (%)
100005	100%
100010	100%
100013	100%
100028	100%
100034	100%
100071	100%
100086	100%
100130	96%
100153	100%
100168	100%
TOTAL	**99,6%**

E. Comparison of the neuro-fuzzy and deep learning approaches

The graph in Fig. 11 shows the decrease of the MAE during learning the neuro-fuzzy system and the deep neural network for 100 epochs. The rules in the neuro-fuzzy system were built on the basis of training samples, which makes the error much smaller in the initial phase of learning. For the DNN, the initial error is very high due to the random selection of weights, but after learning it is slightly smaller than in the case of neuro-fuzzy approach. This is due to the multilayer structure of deep neural networks, which allows for a better learning of the model. All tests have been verified by the cross-validation method in relation to the ratings of movies provided by individual users. Each user has learned his own model using the above-presented methods, which describes his preferences extracted based on his ratings. In order to generate a recommendation for a particular person, we first need to load its model and then to evaluate the candidate samples. The model can also be trained for a certain group of users who are characterized by similar preferences, then the trained model will be common to them. The MAE and the effectiveness of methods presented in Table V was generated for a different number of users. Slight fluctuations result from the number of ranks provided by the users. Our approach for recommendation is independent of the number of profiles.

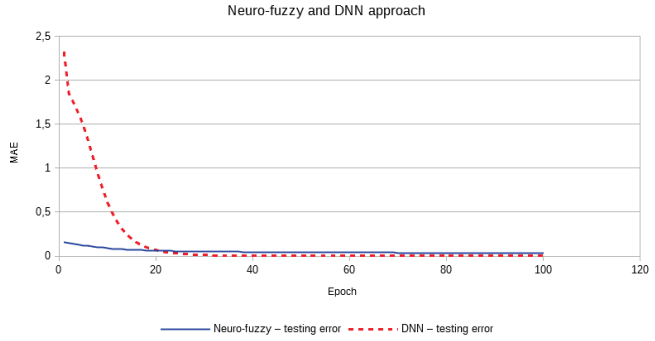

Fig. 11. Error rate decrease

TABLE V
SUMMARY OF MAE AND EFFECTIVENESS OF NEURO-FUZZY (NF) AND DNN IN RELATION TO THE NUMBER OF USERS

	Effectiveness			MAE	
Users	**NF pre-train**	**NF post-train**	**DNN**	**NF**	**DNN**
500	96,79%	98,69%	99,76%	0,03738	0,00332
1000	96,92%	98,80%	99,86%	0,03479	0,00197
1500	97,06%	98,82%	99,90%	0,03455	0,00145
2000	97,17%	98,84%	99,92%	0,03477	0,00121
2500	97,20%	98,85%	99,80%	0,03456	0,00504
3000	97,28%	98,87%	99,83%	0,03426	0,00427
3500	97,26%	98,88%	99,82%	0,03402	0,00412
4000	97,26%	98,87%	99,83%	0,03408	0,00366
4500	97,23%	98,86%	99,85%	0,03437	0,00332
5000	97,26%	98,87%	99,82%	0,03431	0,00369
5500	97,31%	98,88%	99,83%	0,03405	0,00339
6000	97,30%	98,87%	99,84%	0,03418	0,00313
6500	97,30%	98,87%	99,83%	0,03401	0,00325
7000	97,32%	98,88%	99,83%	0,03396	0,00329
7500	97,31%	98,88%	99,81%	0,03389	0,00357
8000	97,32%	98,88%	99,82%	0,03399	0,00336
8500	97,30%	98,87%	99,81%	0,03399	0,00334
9000	97,30%	98,88%	99,82%	0,03402	0,00318
9500	97,30%	98,87%	99,82%	0,03417	0,00322
10000	97,29%	98,87%	99,82%	0,03417	0,00325

IV. CONCLUSIONS

In this paper, we presented two approaches, based on the neuro-fuzzy structure and the deep learning structures applied to build recommender system. Performed simulations demonstrated the very high accuracy of the recommender based on deep neural networks. The recommender based on neuro-fuzzy approach performs slightly worse; however, it generates interpretable fuzzy rules. It should be emphasized that, as it is shown in Table V, it has very good effectiveness, in most cases above 97%, even before the backpropagation training (fine-tuning) is applied. The recommenders based on the neuro-fuzzy structure and DNN have their advantages and disadvantages. A combination of both methods would lead to high-quality recommenders with an option to explain to users the reasoning behind the specific recommendation. This will be the subject of our future research.

REFERENCES

[1] Z. Gantner, S. Rendle, and L. Schmidt-Thieme, "Factorization models for context-/time-aware movie recommendations," in *Proceedings of the Workshop on Context-Aware Movie Recommendation*, ser. CAMRa '10. New York, NY, USA: ACM, 2010, pp. 14–19. [Online]. Available: http://doi.acm.org/10.1145/1869652.1869654

[2] N. Aizenberg, Y. Koren, and O. Somekh, "Build your own music recommender by modeling internet radio streams," in *Proceedings of the 21st International Conference on World Wide Web*, ser. WWW '12. New York, NY, USA: ACM, 2012, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/2187836.2187838

[3] S. Bourke, K. McCarthy, and B. Smyth, "The social camera: A case-study in contextual image recommendation," in *Proceedings of the 16th International Conference on Intelligent User Interfaces*, ser. IUI '11. New York, NY, USA: ACM, 2011, pp. 13–22. [Online]. Available: http://doi.acm.org/10.1145/1943403.1943408

[4] A. Chin, B. Xu, and H. Wang, "Who should i add as a "friend"?: A study of friend recommendations using proximity and homophily," in *Proceedings of the 4th International Workshop on Modeling Social Media*, ser. MSM '13. New York, NY, USA: ACM, 2013, pp. 7:1–7:7. [Online]. Available: http://doi.acm.org/10.1145/2463656.2463663

[5] J. He, J.-Y. Nie, Y. Lu, and W. X. Zhao, "Position-aligned translation model for citation recommendation," in *Proceedings of the 19th International Conference on String Processing and Information Retrieval*, ser. SPIRE'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 251–263.

[6] Q. He, D. Kifer, J. Pei, P. Mitra, and C. L. Giles, "Citation recommendation without author supervision," in *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, ser. WSDM '11. New York, NY, USA: ACM, 2011, pp. 755–764. [Online]. Available: http://doi.acm.org/10.1145/1935826.1935926

[7] F. Isinkaye, Y. Folajimi, and B. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261 – 273, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1110866515000341

[8] K. Meehan, T. Lunney, K. Curran, and A. McCaughey, "Context-aware intelligent recommendation system for tourism," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on*. IEEE, 2013, pp. 328–331.

[9] D. Adeniyi, Z. Wei, and Y. Yongquan, "Automated web usage data mining and recommendation system using k-nearest neighbor (knn) classification method," *Applied Computing and Informatics*, vol. 12, no. 1, pp. 90–108, 2016.

[10] R. Conforti, M. de Leoni, M. La Rosa, W. M. van der Aalst, and A. H. ter Hofstede, "A recommendation system for predicting risks across multiple business process instances," *Decision Support Systems*, vol. 69, pp. 1–19, 2015.

[11] M. D. Ekstrand, J. T. Riedl, J. A. Konstan *et al.*, "Collaborative filtering recommender systems," *Foundations and Trends® in Human–Computer Interaction*, vol. 4, no. 2, pp. 81–173, 2011.

[12] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*. Springer, 2007, pp. 325–341.

[13] R. Yera Toledo and L. Martinez, "Fuzzy tools in recommender systems: A survey," vol. 10, pp. 776 – 803, 03 2017.

[14] R. Burke, "Hybrid recommender systems: Survey and experiments," *User modeling and user-adapted interaction*, vol. 12, no. 4, pp. 331–370, 2002.

[15] Z. Kubicek, "Movie manager," 2014.

[16] M. Färber, C. Menne, and A. Harth, "A linked data wrapper for crunchbase," *Semantic Web*, no. Preprint, pp. 1–11, 2016.

[17] Z. Ivezic, A. J. Connelly, J. T. VanderPlas, and A. Gray, "Statistics, data mining, and machine learningin astronomy," *Statistics, Data Mining, and Machine Learningin Astronomy, by Z. Ivencic et al. Princeton, NJ: Princeton University Press, 2014*, 2014.

[18] K.-j. Kim and H. Ahn, "A recommender system using ga k-means clustering in an online shopping market," *Expert systems with applications*, vol. 34, no. 2, pp. 1200–1209, 2008.

[19] M. R. Chmielewski and J. W. Grzymala-Busse, "Global discretization of continuous attributes as preprocessing for machine learning," *International journal of approximate reasoning*, vol. 15, no. 4, pp. 319–331, 1996.

[20] L.-X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on systems, man, and cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.

[21] L.-X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1994.

[22] L. Rutkowski, *Computational intelligence: methods and techniques*. Springer Science & Business Media, 2008.

[23] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 19:1–19:19, Dec. 2015. [Online]. Available: http://doi.acm.org/10.1145/2827872

[24] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[25] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[26] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv preprint arXiv:1502.01710*, 2015.

[27] M. Cassel and F. Lima, "Evaluating one-hot encoding finite state machines for seu reliability in sram-based fpgas," in *12th IEEE International On-Line Testing Symposium (IOLTS'06)*, 2006, pp. 139–144.