Content-based Re-ranking Method for Recommendation

Xin Huang\textsuperscript{1,2}, Jiameng Bai\textsuperscript{1,2}, Jia Peng\textsuperscript{1} and Min Li\textsuperscript{1}

\textsuperscript{1}State Key Laboratory of Information Security, Institute of Information Engineering, CAS, Beijing, 100093, China
\textsuperscript{2}School of Cyber Security, University of Chinese Academy of Sciences, Beijing, 100049, China
\textsuperscript{*}Corresponding author’s e-mail: pengjia@iie.ac.cn

Abstract. The recommendation system is one of the effective tools to solve information overload. Most of the current deep learning recommendation algorithms only focus on the accuracy of the recommendation results by learning users’ preferences. However, the diversity of the recommendation results is neglected, resulting in homogeneous recommendation results, which reduces users’ satisfaction. Diversity can not only effectively avoid over-fitting, but also comprehensively consider multiple dimensions to improve the quality of recommendation results. Therefore, we propose a content-based re-ranking method (CBR) for recommendation systems. The proposed method can make full use of the data set, and effectively supplement and re-rank the recommended results produced by some deep learning algorithms based on the edge information such as tags to be recommended from different perspectives, thus can effectively improve the diversity while preserving the accuracy of recommendation results. Experimental results demonstrate the significant improvements of the proposed re-ranking method.

1. Introduction
With the development of the Internet, people are in an era of information explosion. The recommendation system is one of the most effective tools for solving information overload. It is committed to learning the potential interests of each user by establishing a binary relationship between users and items.

Recommendation algorithms based on deep learning realize the abstract expression of data and the generation of high-level semantic concepts through a multi-layer feature extraction mechanism. They achieve better results than traditional recommendation algorithms in terms of accuracy and recall rate. Jun Wang\textsuperscript{[1]} proposed a two-layer model IRGAN consisting of the generative model and discriminant model, and it performs best. However, like some other recommendation algorithms, the model only uses the scoring matrix for feature modeling, which wastes semantic-rich edge information such as tags, resulting in a low diversity of recommendations. Some algorithms use the tag information to make recommendations. Xu\textsuperscript{[2]} uses DSSM to learn tags and map users and items to the semantic space. Han Xiao\textsuperscript{[3]} and others proposed a two-layer feature generation model SAR, mapping users and items to the same semantic space, and then performing semantic similarity calculation. These two models are slightly worse in terms of accuracy, but they prove that tags can reflect the content of items.

In 2013, Castagos\textsuperscript{[4]} conducted a survey to provide users with a diverse list of recommendations, analyzing users’ satisfaction. The study found that the diversity of recommendations significantly improves users’ satisfaction with the recommendation system. Deep learning algorithms produce
accurate recommendation results, but they usually neglect the diversity of recommendations, thus have the following deficiencies\textsuperscript{[5-7]}. Firstly, they usually use only part of the data information, resulting in low data utilization. Secondly, their recommendation results are more homogeneous, affecting user satisfaction.

To solve these problems, we propose a content-based re-ranking method (CBR). CBR divides the dataset into scoring information and tag information and adds probabilistic models to content-based recommendations to effectively use the tag information. It can make full use of the dataset and make recommendations from multiple perspectives while preserving the high accuracy of given deep learning recommendation results, thus produce more diverse results. Besides, the cold start problem can be solved because there is no requirement for user ratings for content-based recommendations.

In summary, the main contributions of this paper are as follows:

- **Diversity.** We add probabilistic models to content-based recommendations to increase the likelihood of recommending different tags. This method ensures the accuracy of the given results and further improves diversity.
- **Universality.** Our proposed method is based on the existing results, so it can be applied to a variety of deep learning algorithms, which has a certain universality.
- **Usability.** We combine and re-rank deep learning results and content-based results, and set parameter adjustments to make it a realistic recommendation system that is suitable for various situations.

2. Related work

In order to increase the diversity of recommendations, the current main research can be divided into the following three based on the different implementation methods: adding semantic information, using multiple recommendation algorithms, and re-rank the result.

- **Adding semantic information.** Rao Junyang\textsuperscript{[8]} integrated semantic information into the content-based recommendation system, thereby enhancing the relevance between users and items, and calculating the correlation between users and projects to improve the quality.

- **Using multiple recommendation algorithms.** Cao Yi\textsuperscript{[9]} combines content-based filtering and collaborative filtering, and can adopt different recommendation strategies at different stages of the recommendation system. Li Zhongjun\textsuperscript{[10]} proposed a new model, the prediction matrix of content-based recommendation is the input of collaborative filtering, thus predicting the user's score on the new project.

- **Re-rank the result.** Adomavicius\textsuperscript{[11]} firstly used collaborative filtering to get the results, secondly filtered the results according to certain strategies. Finally, the results were re-ranked according to the popularity and average score of the items to balance the accuracy and overall diversity. Chen Tianyi\textsuperscript{[12]} used collaborative filtering and content filtering to remove the similar recommendation results to improve the diversity.

At present, a large amount of research has focused on one of the above aspects, and relatively few studies have incorporated it. This paper is dedicated to integrating above methods, while ensuring the accuracy of the recommendation results, and increasing the diversity of recommendations.

3. Content-based re-ranking method

The method consists of three parts, the main structure flow chart is shown in Figure 1. The data set used in this model is movielens. Movielens is split into three parts: movie tags, user rating data, user tags. Different methods are recommended for different data:

- For rating data, the relationship between the user and the movie is modeled by using deep learning models such as IRGAN to obtain the candidate set $RD$ for its’ powerful feature extraction capabilities.

- For movie tags, firstly pre-processing the tag, and then the cosine similarity is used to calculate the feature vector similarity. In this section, the candidate movie is sampled according to the
weight of the feature and a probabilistic model, so the diversity of the recommendation can be effectively improved.

- For user tags, the potential association between users’ intuitive attributes such as age, occupation, and other attributes can be found. Thus, the recommendation according to the users’ preference with similar tag characteristics can effectively improve the diversity of the recommendation and bring certain surprise to the user, and can also solve the users’ cold start problem.

3.1. content-based recommendation

The method proposed in this paper combines the content of the movie and the user attribute to recommend. We extract the features of movies and users, then calculates the similarity.

3.1.1. content-based recommendation for movie. It is generally believed that the tags of a movie represent the type of the movie and can reflect the actual content of the movie. Therefore, content-based recommendation for movies needs to be based on the tags of the movie.

**Feature construction.** In this method, n-gram is used to construct feature words. Unlike the n-gram model in traditional documents, movie tags in movielens are independently scattered and unordered, so the construction features are described by a combination of multiple tags. For example, the tags of “Say Anything (1989)” is “Comedy | Drama | Romance”, whose bi-gram are (Comedy Drama), (Comedy Romance) and (Drama Romance). In addition, since there are some movies with only one tag, in this model, if an n-gram is used to construct the feature, it actually contains 1-gram, 2-gram, ..., and n-gram together to form a feature vector and then calculate the weights. This article uses $tf-idf$ to calculate the weight of features:

$$tf - idf = \frac{tf_{w,d} \times idf_w}{N \times \log \frac{D}{d_w}} \tag{1}$$

Where $w$ represents a term, $d$ represents a document, $n$ represents the number of occurrences of the term $w$ in the document $d$, and $N$ represents the total number of terms of the document $d$. $D$ represents the number of documents in the document set, and $d_w$ represents the number of documents in the document set containing the term $w$.

**Feature similarity calculation.** Both the user and the movie can be mapped to the feature space, represented by a feature vector. A movie can be directly represented as a feature vector by the weight of the feature. The user will have behavior on the movie, and the tags of the movie that the user has scored highly are used as the feature tag of the user itself, and can also be mapped to the feature space, as shown in Figure 2.
The user can use the feature vector to represent the matrix $U(m \times k)$, a total of $m$ rows and $k$ columns, $m$ represents the number of users, $k$ represents the number of features constructed. Where $U_{ij}$ is expressed as the weight of feature $j$ for user $u_i$. The movie can be represented as a matrix $M(n \times k)$, $n$ represents the number of movies, $k$ represents the number of features constructed. After obtaining the feature vectors of the user and the movie, the relevance between the user and the movie are calculated using the cosine similarity. As shown in Figure 3, it can be intuitively seen that user $u_1$ is significantly more relevant to $m_1$ than movie $m_2$.

For any user $u_i$ and any movie $m_j$, the correlation degree weight integral $rel$ can be calculated as:

$$ rel = \log \cos(U_{i*}, M_{j*}) $$

The log is used to prevent the product from being too small when calculating the cosine similarity. **Recommend algorithm.** In this method, the main part of the recommendation result set is derived by the deep learning algorithm. The content-based recommendation is mainly to make some supplementary recommendations to $RD$ to enhance its diversity. Given $RD$, and all the features of $RD$ are regarded as a document, and the times of occurrence of the features is counted. The more the feature appears, the better this feature can represent the user's preference. A lower frequency indicating that the feature does not reflect the user's real interest. In order to improve the diversity of recommendations, it is necessary to punish the features with more occurrences, so that it is not easy to be selected in the process of film content-based recommendation, and vice versa, it should reward the features with fewer occurrences, making it easier to be recommended.

Construct the feature vector $r$ with $k$ columns correspond to $k$-dimensional features, and each column value corresponds to the frequency at which the feature appears. Then the user feature weight vector $w'$:

$$ w'_j = \frac{U_{ij}}{\log (\max (e, r_j))} $$

Assuming $\text{sum} = \sum_{i=1}^{k} w'_i$, normalization can be obtained $w = \frac{w'}{\text{sum}}$. Changing the value of $w$ to make the value of $w_i$ is equal to the value of the original $w_i$ plus the value of $w_{i-1}$. So that it maps to a range of 0-1. The calculation process like a Fibonacci sequence:

$$ \text{for } i = 1 \text{ to } k-1 : \quad w_i = w_i + w_{i-1} $$

Given a random number $\varepsilon_1$, we can select a feature according to $w$. As shown in the Figure 4, if $\varepsilon_1 = 0.3$, then we choose the feature $f$ corresponding to $w_3$ to recommend. After selecting the feature, we need to select the movie that is relevant to the feature for recommendation. The recommendation process is similar to the feature selection process described above. For feature $f$, the correlation vector with the movie is $M_{f*}$. Add the probability value of each movie and normalize it, then obtain the weight vector $t$ of the feature-movie. Take a random number between 0 and 1 to get the corresponding movie in $t$ for recommendation. Assuming the random number is $\varepsilon_2$, the recommended procedure is shown in Figure 5.
3.1.2. content-based recommendation for user. Content-based recommendation for user is to solve the problem of cold start of the user. Besides, it can make full use of the user's tag information to carry out personalized recommendation. As shown in Figure 6, this process is similar to movie content-based recommendation. But there still has two differences.

- User content-based recommendation is based on the user's tag to build features, such as gender, age and other attributes.
- The model obtains the user-tag matrix $U$, $U_r$ represents the feature vector of the user $u_i$, and can directly calculate the user similar to the target user by the cosine similarity. Then recommending the movie similar to the user, and obtaining the user-recommend result set $UD$.

3.2. Combing and re-ranking

After obtaining $RD$, $MD$ and $UD$, as Figure 7 shows, the results need to be combined by a certain strategy to form a final recommendation result set $result$.

For a recommendation system, the most important thing is to recommend the items that users like, and secondly to recommend as many categories as possible to enhance the user experience. Therefore, $RD$ with high accuracy should be placed in the top position in the original order, with a proportion of $\alpha$. After that, $MD$ is placed, and the proportion is $\beta$. Finally, the $UD$ is placed, and the proportion is $\gamma$, and $\alpha + \beta + \gamma = 1$.

For users with more rating records, their preferences are relatively easier to model, so the weight of $\alpha$ should be appropriately increased to obtain better performance. If the user scores less, it is not easy to obtain their preferences, and the weight of $\alpha$ should be appropriately reduced to solve the cold start problem. When $\beta=1$, the model is transformed into a full content-based recommendation model. When $\alpha=1$, the model is transformed into a complete deep learning recommendation model.

4. Experimental result

4.1. Experimental Settings

The experimental environment for this article is configured for Windows 10, Python 3.6.
The data set used movielens 100k containing 943 users' 100,000 ratings for 1,682 movies. In addition, it also includes the title of the movie, as well as the tags of the movie, some attributes of the user, and so on. The evaluation methods used in this paper include the accuracy rate P, the normalized discounted cumulative gain NDCG. At the same time, the number of different categories included in the recommendation results is recorded as “genres”, indicating the diversity of recommendations. In this experiment, based on the experimental results, the values of $\alpha$, $\beta$ and $\gamma$ are set to be 60%, 20% and 20%. As for n-gram to generate features, the recall rate for content-based recommendations is shown in Table 1.

| n-gram | recall@5 | recall@10 |
|--------|----------|-----------|
| 1-gram | 14.37    | 23.53     |
| 2-gram | 12.26    | 19.85     |
| 3-gram | 10.94    | 18.12     |
| 4-gram | 10.56    | 17.60     |

Based on the results, 1-gram is used to construct the feature in the actual modeling process.

4.2. Experiments
In this paper, deep learning methods IRGAN and BPR\cite{13} are selected for baselines. Both models use only the ratings data of the movielens data set. CBR method was used to supplement and re-rank the result sets produced by IRGAN and BPR. Accordingly, we name the two relevant model CBR-IRGAN and CBR-BPR. The results of the experiments are as follows.

| Model            | P@5  | P@10 | NDCG@5 | NDCG@10 | genres@5 | genres@10 |
|------------------|------|------|--------|---------|---------|-----------|
| IRGAN            | 37.17| 31.39| 40.03  | 37.18   | 2       | 3         |
| CBR-IRGAN        | 35.74| 30.94| 38.46  | 37.03   | 4       | 7         |
| BPR              | 30.03| 26.47| 32.21  | 31.89   | 3       | 4         |
| CBR-BPR          | 29.97| 26.64| 31.89  | 31.78   | 4       | 7         |

Comparing IRGAN and BPR, it can be seen that P and NDCG of IRGAN and BPR decrease obviously with the increase of the recommended number, indicating that the correlation is weakened with a lower ranking. There is still a certain gap between the recommended accuracy of BPR and IRGAN. However, the average number of Genres in BPR is more than that of IRGAN, indicating that IRGAN is more specific and accurate in modeling user preferences, but the recommended types are relatively homogeneous and not diverse.

Comparing CBR-IRGAN and CBR-BPR, the overall performance of the CBR method is closely related to the performance of the deep learning algorithm, so the diversity and accuracy of the recommendation are still inseparable.

After adding the CBR method to the IRGAN and BPR models, it can be seen from the experimental results that the accuracy of the model has a little degree of influence, but the impact gradually decreases as the number of recommendations increases. However, it can be seen that the CBR method has a very significant improvement in the number of types of recommended movies compared to the original model. In the case where the recommended accuracy is not much different, the increase in diversity will greatly enhance user satisfaction. Therefore, the CBR method has broad application prospects.

5. Conclusion and future work
Most of the recommended methods currently available are blindly considering the accuracy of the recommendation, while ignoring the diversity of recommendations. This paper proposes a CBR method that can re-rank the existing results and add content-based personalized recommendation results to enhance the diversity of recommendations. The method can fully utilize the information of
the data set, and learn the user’s interest from various dimensions, and it can also solve the problem of recommendation over-fitting effectively which has a positive significance.

However, the concept of diversity is still vague at present. Many studies on diversity are subjective, and the actual user demand for diversity is not easy to quantify. And the research on the diversity of recommendations does not form a systematic theory.

CBR can solve certain diversity problems at present, but it still needs further investigation on how to adjust the weighting parameters automatically of different recommendation result sets. At the same time, the recommended accuracy is still insufficient compared with the original recommendation method when the recommended number is small. We leave more theoretical investigation on these problems for future work.

Acknowledgments
This work is partially supported by National Key Research and Development Program of China (No.2019QY1704).

References
[1] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinhui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. (2017) IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. SIGIR. Tokyo. pp. 515-524.
[2] Xu Z, Chen C, Lukasiewicz T, et al. (2016) Tag-aware personalized recommendation using a deep-semantic similarity model with negative sampling. CIKM. Indianapolis. pp. 1921-1924.
[3] Han Xiao, Minlie Huang, Xiaoyan Zhu. (2017) SAR: A Semantic Analysis Approach for Recommendation. ICJAI. Melbourne.
[4] Castagnos, S., Brun, A. and Boyer, A. (2013) When Diversity Is Needed...But Not Expected!. In: Advances in Information Mining and Management. Lisbon. pp. 44-50.
[5] Ho, Y.-C., Chiang, Y.-T. and Hsu, Y.-J. (2014) Who Likes It More? Mining Worth-Recommendating Items from Long Tails by Modeling Relative Preference. In: ACM International Conference on Web Search and Data Mining. New York. pp. 253-262.
[6] Lee, K. (2015) Escaping Your Comfort Zone: A Graph-Based Recommender System for Finding Novel Recommendations among Relevant Items. Expert Systems with Applications, 42(10): 4851-4858.
[7] di Noia, T., Ostuni, V., Rosati, J., Tomeo, P. and di Sciascio, E. (2014) An Analysis of Users’ Propensity toward Diversity in Recommendations. In: 8th ACM Conference on Recommender Systems. Foster City. pp. 285-288.
[8] Rao Junyang, Jia Aixia, Feng Yansong. (2014) Ontology-based News Personalized Recommendation. Acta Scientiarum Naturalium Universitatis Pekinensis, 50(1): 2-7.
[9] Cao Yi. (2007) Research on a Hybrid Recommendation Model Based on Collaborative Filtering and Content Filtering. Changsha.
[10] Li Zhongjun, Zhou Qihai, Shuai Qinghong. (2009) Recommender System Model Based on Isomorphic Integrated to Content-based and Collaborative Filtering. Computer Science, 36(12): 142-145.
[11] Adomavicius, G. and Kwon, Y. (2012) Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. In: IEEE Transactions on Knowledge and Data Engineering. 896-911.
[12] Chen Tianhao, Shuai Jianmei, Zhu Ming. (2014) A Film Recommendation Method Based on Collaborative Filtering. Computer Engineering, 40(1): 55-58.
[13] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. (2009) BPR: Bayesian personalized ranking from implicit feedback. In UAI. pp. 452-461.