An Ensemble Method for Job Recommender Systems

Chenrui Zhang
CAS Key Lab of Network Data Science and
Technology, Institute of Computing Technology,
Chinese Academy of Sciences
Beijing, China
zhangchenrui@software.ict.ac.cn

Xueqi Cheng
CAS Key Lab of Network Data Science and
Technology, Institute of Computing Technology,
Chinese Academy of Sciences
Beijing, China
cxq@ict.ac.cn

ABSTRACT

In this paper, we present an ensemble method for job recommendation to ACM RecSys Challenge 2016. Given a user, the goal of a job recommendation system is to predict those job postings that are likely to be relevant to the user[1].

Firstly, we analyze the train dataset and find several interesting patterns. Secondly, we describe our solution, which is an ensemble of two filters, combining the merits of traditional collaborative filtering and content-based filtering. Our approach finally achieved a score of 1632828.82, ranked at the 10th place on the public leaderboard.

Categories and Subject Descriptors

H.3.3 [**Information systems**]: Recommender systems

Keywords

top-N Recommendation, RecSys Challenge 2016, LSI, word2vec, Ensemble

1. INTRODUCTION

The recommender system technology plays an important role in various e-commerce applications by helping individuals find right items in a large option space, which match their interests. The mainstream approaches to recommender systems are classified into four categories: Collaborative Fitering(CF), Content-Based Filtering(CBF), knowledge-based and hybrid approaches [6]. Besides, utility-based and demographic approaches were also discussed in some papers.

All recommendation approaches have their specific strengthes and weaknesses. The main advantage of CF approaches is that they can find the patterns among user ratings data and work well for complex objects. On the other hand, there are some challenges suffered by CF, such as, cold-start and ramp-up problems [1].

CBF try to recommend items based on similarity in content and hence is also referred as item-to-item correlation method [4]. Item description and a profile of the user's orientation play an important role in CBF [5]. CBF does not suffer from the new-item problem, but still the new-user problem.

The hybrid filtering is a combination of more than one filtering approach. The hybrid filtering approach is introduced to overcome some common problem that are associated with above filtering approaches such as cold start problem, overspecialization problem and sparsity problem. Another motive behind the implementation of hybrid filtering is to improve the accuracy and efficiency of recommendation process [5].

During this challenge, All teams are allowed to upload five submissions a day. The score calculated on a fixed 1/3 of the ground truth data is shown on the public leaderboard, and the entire ground truth data(private leaderboard) is used to determine the final standings.

RecSys Challenge 2016 is a competition for predicting job postings that are likely to be relevant to target users. It is a personalized recommendation problem. There are some challenges:

- *matching* - a best fit between job and candidates may depend on many aspects, such as locations, career levels, job roles and so on.
- *ranking* - How to provide a way to determine the order in which to present items when the recommender finds too many matching items.

2. PROBLEM STATEMENT

The central task of this challenge is to predict those job postings (items) that the user will interact with in the next week. In this section, we give more details on the data typically and share some data mining on them. In addition, we will use items to denote job postings in the following sections.

2.1 Data Description

The training data provided by RecSys Challenge 2016 [1] comprises of interactions performed by a set of users from week 34 to week 45 of year 2015 on a business-oriented social networks, along with users and items profile, part of impression data. In addition, the impression data is generated by XING's job recommender. Testing data is made up of target users and a list of active items. The task of the challenge is to compute 30 recommendations (or less) for each of the 150,000 target users. Each interaction is described by:

- *user_id* — the unique identifier of a user;

[1] http://2016.recsyschallenge.com/

RecSys Challenge '16, September 15 2016, Boston, MA, USA
© 2016 ACM. ISBN 978-1-4503-4801-0/16/09…$15.00
DOI: http://dx.doi.org/10.1145/2987538.2987545

- *item_id* — the unique identifier of a job posting;
- *interaction_type* — the type of action that was performed by a user, which includes four types, clicking, bookmarking, replying to and deleting a job posting;
- *created_at* — a unix time stamp representing the time when the interaction got created.

There are a total of 7.2M clicks, 206K bookmarks, 422K replies and 1M deletes in the training data. The time span of those interactions is 3 months. We have a large number of clicks and deletes available, in contrast, rather few replies or bookmarks. The interaction data generated by users is unbalenced because clicks and deletes cause less effort compared to bookmarks and replys.

The total number of item IDs and user IDs is 1,358,098 and 1,367,507 correspondingly. The attributes in users profiles and items profiles are similar, both contains *career_level*, *discipline_id*, *industry_id*, *country*, *region*, *employment*. The *jobroles* field in users corresponds to is to the *title* and *tags* in items. The career level of majority users and items is 3 , which corresponds ”professional/Experienced”.

2.2 Evaluation Measure

According to the target of top-k recommender systems, the evaluation of the contest is taking into consideration both precision and recall measure. The evaluation metric is computed by the following equation:

$$\begin{aligned} Score(sl) \quad = \quad & \sum_{i=1}^{n} 20 * (U_{i,p@2} + U_{i,p@4} + U_{i,r} + U_{i,us}) + \\ & 10 * (U_{i,p@6} + U_{i,p@20}) \end{aligned}$$

where the symbols are:

- $U_{i,p@k}$ - precision within the first top k items.
- $U_{i,r}$ - recall within top 30 items.
- $U_{i,us}$ - at least one relevant item recommended for user i.

2.3 Challenges and Analysis

To solve the challenge, we start with an analysis of the training data. We observe that users’ implicit feedbacks are not always consistent with their profiles. The discrepancy means that the expectations users expressed in words are not accurately match with items’ titles and tags. The overlapping percentages of title-jobroles and tags-jobroles are computed from positive interactions by week(See Table 1). It shows that only ten percentage of postive interactions can be easily explained to word matching in content.

Table 1: Overlapping Challenge

Week	Active Users	Title-jobroles	Tags-jobroles
45	130128	11.3%	7.4%
44	137676	11.2%	7.4%
43	143738	11.7%	7.5%
42	140111	11.4%	7.4%

Going deep into implict feedback information, we observe that both users and items follow the heavy-tailed distribution(see Figure 1). The horizontal axis represents the number of users(left) and items(right), and the vertical axis sums the probability mass from positive interactions for each user or each item. It also shows that positive interactions are sparse, and majority users and items appear several times in the training interactions dataset.

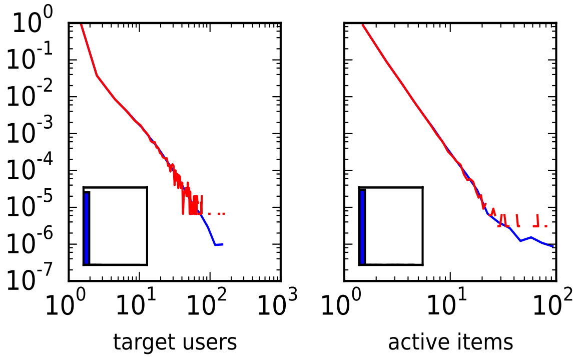

Figure 1: An analysis of bookmarking behaviors.It shows that both users and items fit the pow-law distribution well.

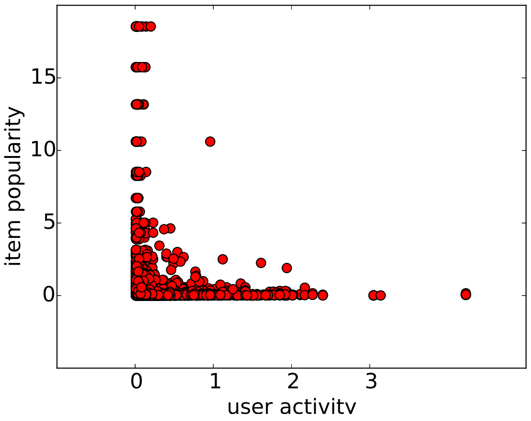

Figure 2: The relation of user activity and item popularity

We capture the relationship between user activity and item popularity by a statistical analysis of interactions data. There is a pattern which can be illustrated in Figure 2. The horizontal axis is user activity which represents frequency of positive interactions by each user in week 44. The vertical axis is item popularity. When a user is a new starter, he perfers some popular items in most cases. So the recommender systems can select some popular items to display when there are no history records about a given user in the system database.

Table 2: Target users

	Total	Not-cold	Cold
target users	150,000	110,245	39,755
target items	327,003	194,217	132,786

Overlapping exists in the interactions for a given user between every 2 weeks. The percentage is about 30%. Some users may bookmark or reply to several job postings in one day, and continue to interact with them next several days. So the simple and effective way is to select some recently interacted items by users to make a personalized recommendation.

3. SOLUTION METHOD

In this section, we describe the details of our approach for the given recommended task in this challenge. Firstly, we introduce some preprocessing on the implicit feedback data. Secondly, we make content analysis on Latent Semantic modeling. Then, we analyze item-to-item relations and describe a method named Item2Vec for item-based CF that produces embedding for items in a latent space. Lastly, we ensemble two models to make a top-30 recommendation list for each target user.

3.1 Ratings for Implicit feedback

There are totally four different user actions extracted from implicit feedbacks that are provided by this challenge, including *clicking*, *bookmarking*, *replying to* and *deleting*. All of them provide different levels of evidence for interest. If a user bookmarks or replies to a job posting, it implies that the user is interested in it to a great extent. Conversely, clicks can signal intersets, curiosity or be due to accident [2].

A better recommendation system needs an expressive representation, which is able to capture users' interests and preferences. Considering the factor of action type and frequency, we transform implicit feedback to ratings with a scale of 0-5 based on artificial rules(See Table 3).

Table 3: rating transformation

Type of actions	Frequency	rating
clicking on the items	1 or 2	1
clicking on the items	2,3,4,5	3
clicking on the items	over 5	5
bookmarking	1+	5
clicking on the reply button	1+	5

3.2 CF Approach

Recommender systems usually use tradition CF methods based on ratings to find users with similar taste. In this challenge, our approach takes Latent Semantic model(LSI) to capture similar users in interactions. LSI is a method that use singular value decomposition(SVD) to identify relationships between a set of documents and the terms they contain. The assumption in LSI is that the new dimensions are a better representation of documents and queries. Here is a brief overview of the user-based CF method.

Suppose A is a $m * n$ matrix, the SVD operation of A is:

$$A_{m*n} \approx U_{m*k} \Sigma_{k*k} V_{k*n}^T$$

In this challenge, we use SVD to judge the similarity between a pair of users in interactions. Each user is treated as a row, items are treated as the columns, the elements in matrix is the score tranformed from implicit feedbacks. After building a SVD model, we can make a recommendation by the interactions of the given users' neiborhoods.

The detail steps are described in the following paragraph:

- Selecting the interactions in last week, we construct a sparse matrix, where each row represents a user, and a column represents a item. The size of the corpus matrix is *users_size*items_size*. The value in the matrix cell is the implicit feedback score.

- Use the corpus to build the LSI model, the number of topics we selected is 50.

- Based on the LSI model, each user as a document is represented by 50-dimensions vector.

- For a given user, selecting top items as a candidate set based on the neighbors list.

Given a user i, the CF probability that i will interact with item j is computed:

$$P_{ui} = \sum_{j \in N(u) \bigcap (i,K)} w_{ij} r_{uj}$$

3.3 CBF Approach

In most recommendation systems, an accurate profile of the user's interests plays an important role. There are two main ways to use the history of user interactions to build a profile. Firstly, the recommender can simply display some recently bookmarked or replied to items to facilitate the user focusing on them. Secondly, the system can use the history information to train a user model to predict users' interest in content. In this challenge, our approach takes word2vec model to capture similar items in content. Here is a brief overview of the word2vec method.

Word2vec uses continuous bag-of-words (CBOW) or continuous skip-gram to produce a distributed representation of words[**?**]. Each word is represented as a several hundred dimensions vector such that words that share common contexts in the corpus are located in close proximity to one another in the space [3]. Given a sequence of words $(w_i)_{i=1}^K$ from a finite vocabulary $W = \{w_i\}_{i=1}^W$, the skip-gram objective aims at maximizing the following term:

$$\frac{1}{K} \sum_{i=1}^{K} \sum_{-c \leqslant j \leqslant c, j \neq 0} \log p(w_{i+j} | w_i)$$

where c is the context window size, and $p(w_j|w_i)$ is the softmax function:

$$p(w_j|w_i) = \frac{\exp u_i^T v_k}{\sum_{k \in I_w} \exp u_i^T v_k}$$

where u_i and v_i are latent vectors that correspond to the target and context representations for the word w_i.

In this challenge, we use word2vec to compute the similarity between tag words, and build a doc2vec model to judge the content similarity between a pair of items. Each job posting is treated as a document, title and tags attributes as the terms[2]. After building the model, we can also recommend some new job postings judged by users' history. It solve the new-item cold start problem.

Firstly, we construct a train corpus that each item as a document comprise of a set of tags. Then training a word2vec model and each item is represented by a vector,

[2] In this paper, we don't take other fields of items into account.

we use cosine similarity to compute the affinity between two items.

Note that different number of topics only impact little on the similarity of items. so we select 100 topics defaultly. After building the model, we can reduce the number of candidate items for each target user.

Our approach generalizes to unseen job postings by providing a representation of the items in terms of latent space vectors discovered from the corpus. The another advantage is that CBF is interpretable and makes meaningful recommendations on job postings before anyone interacts with them.

3.4 Ensemble methods

Ensemble methods which combine multiple models usually produce more accurate results than a single model would. In recommender systems, all approaches have their specific strengthes and weaknesses. So we make an ensemble of CF and CBF methods finally.

In order to improve leaderboard perfermance, We adopt a two-stage ensemble of CF and CBF methods. Firstly, we use CBF method to select 100 relevant items for each user, and secondly use the CF method to re-rank the list and select a top-n list. In the collaborative filtering approach, the recommender systems identify rating patterns and recommend items that users' neighbors liked. In the content-based filtering approach, the algorithms make an recommendation by similarity in content. For each user, our approach find both popular job postings that are important to other similar users and new job postings whose content matches the users's specific interests.

The detail codes are published in my github [3]

4. RESULTS

In this section, we show the detail result of our approach. We use two kinds of data - the users' history and the content of job postings - to form the final recommendation in this challenge.

The simple way to generate a recommendation list for a given user is to use the history interactions, it achieves a score of 300257.63 on the public leaderboard. The result shows that some users prefer focusing on several items for some time. This is an important feature in job recommendation systems,

The impression data is generated by XING's job recommender, we select week 42,43,44,45 to make recommendation on the next week. The score is better than approach 1.

Table 4: Results

	Details	Leaderboard
Approach 1	simple history interactions	300257.63
Approach 2	past four weeks impressions	381415.34
Approach 3	Approach 1,2 + CF	526704.96
Approach 4	Approach 1,2 + CBF	537848.67
Approach 5	ensemble	542213.51

Then, we combine approach 1,2 and CF algorithm to make a new solution. The CF approach which is based on the LSI model discovers the users patterns in interactions. The score is 526704.96.

The CBF approach which is based on the Word2vec model computes the similarity of items in content. The greatest advantage of CBF is to solve new-item cold start problem. We can take some unseen items into consideration in the recommender systems. The approach 4 which combines 1,2 and CBF achieves a score of 537848.67.

Finally, we ensemble CF and CBF methods to generate recommendation lists for 150,000 target users. The result of the ensemble approach is 542213.51 on the leaderboard. It takes advantage of CF and CBF and makes a diversity recommendation system.

5. CONCLUSIONS

In this paper, we describe our method to solve the RecSys Challenge 2016, which is an ensemble approach that combines both collaborative filtering and content-based filtering. The approach we propose is not the best but rather a simple and easy solution. With in-depth understanding of recruitment and requirement of job searchers, the job recommender systems leave a lot to be improved.

To improve the quality of job recommender systems, future work will concentrate on drawing a more detail users profile by history data.

6. ACKNOWLEDGMENTS

This work was funded by the National Basic Research Program of China (973 Program) under grant numbers 2012CB316303, the National High-tech R&D Program of China (863 Program) under grant number 2014AA015204, and the National Natural Science Foundation of China under grant numbers 61425016, 61472401, 61572473, 61572467.

7. REFERENCES

[1] S. T. Al-Otaibi. A survey of job recommender systems. *International Journal of Physical Sciences*, 7(29):5127–5142, 2012.

[2] B. Kille and F. Abel. Engaging the Crowd for Better Job Recommendations. In *Proceedings of Workshop on Crowdsourcing and Human Computation for Recommender Systems (CrowdRec 2015)*, Sept. 2015.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[4] J. B. Schafer, J. Konstan, and J. Riedi. Recommender systems in e-commerce. In *Electronic Commerce*, 1999.

[5] P. B. Thorat, R. Goudar, and S. Barve. Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications*, 110(4), 2015.

[6] K. Wei, J. Huang, and S. Fu. A survey of e-commerce recommender systems. In *2007 International Conference on Service Systems and Service Management*, pages 1–5, June 2007.

[3] https://github.com/Cherishzhang/recsys-challenge-2016