DS 504 Project 2 Final Report

Uploader Recommendation System for Bilibili Users

Group 3

Wei Wang, Dongyu Zhang, Yuchen Shen

ABSTRACT

Bilibili is one of the most popular video sharing website themed around animation, comic, and game (ACG) based in China. In this paper, we use bilibili API to collect uploaders and users information and develop the recommendation system using collaborative filtering. The recommendation system is generated from three methods: User-Uploader Interest, we measure the similarity between the interest of a user and the focusing region of an uploader, and recommend similar focusing uploaders to the user; User-User, we measure the similarity between users and recommend uploaders having similar focusing region followed by similar users to the target user; Matrix factorization, we use stochastic gradient descent to minimize the loss and update two factor matrices. The basic model of our recommendation model is Collaborative Filtering. Using cosine similarity, we evaluate the performance of the three models, The result shows that matrix factorization outperforms the other two methods.

Keyword

Uploader, Recommendation System, Bilibili, Matrix factorization, Collaborative Filtering

1. INTRODUCTION

1.1 Bilibili Website

Bilibili (Chinese: 哔哩哔哩, nicknamed as B站, literally "the B site") is a video sharing website themed around animation, comic, and game (ACG) based in China, where users can submit, view, and add commentary subtitles on videos.

The videos are mostly user-generated content. The items are described through content data and metadata. Condata like length and signal quality are inherent in the video stream. Meta-data like title and description is included by the uploader. The recommendations are displayed based on the user's personal activity (watched, being favorited and liked videos) and by the activity of others. The main benefit for the users is the visibility of videos that they would not find otherwise, as well as the possibility of easily finding previously watched videos.

Data about items:

- Content data (length, quality of video etc) is inherent in the video stream. For each item, there is also data about the total number of views, rating, number of comments and shares, upload time etc.
- Video meta-data(title, description etc) is included by the uploaders but can be incomplete or erroneous.

Data about users:

- Explicit user activity data is generated when the user does specific activities on the site (rating and liking videos, subscribing to an uploader)
- Implicit user activity data (starting to watch a video, watching a large part of it) is generated asynchronously while the user is on the site. This data is partly based on predefined categories as watching a large part of a video corresponds to a predefined length.

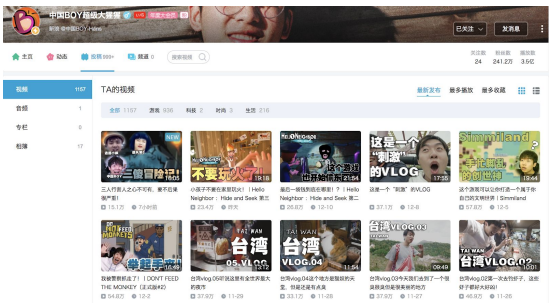

Figure 1. Information page of Bilibili uploader.

Figure 2. Following information page of Bilibili users.

1.2 Challenges

Recommending Bilibili videos is extremely challenging from three major perspectives:

- Scale: Many existing recommendation algorithms proven to work well on small problems fail to operate on a large scale.
- Freshness: The recommendation system should be responsive enough to model newly uploaded content as well as the latest actions taken by the user.
- Noise: Historical user behavior on Bilibili is inherently difficult to predict due to sparsity and a variety of unobservable external factors. We rarely obtain the ground truth of user satisfaction and instead model noisy implicit feedback signals.

1.3 Motivation and Benefits

Bilibili is one of the internet's biggest video sharing platform. Since it has to deal with incredible amounts of video content it triest omake viewer stay at the platform by recommending videos, channels, and users they might like. Obtaining recommendations is a critical component of the process of human decision making. With burgeoning consumerism buoyed by the emergence of the web, users are being presented with a range of choices while the website is being faced with the challenge of personalizing content. In parallel, it has become common for enterprises to collect large volumes of transactional data that allows for deeper analysis. Recommender system can fulfill the requirements of users and website by automating the generation of recommendations based on data analysis.

1.4 Previous Work

Recommender systems typically produce a list of recommendations in one of two ways – through collaborative filtering or through content-based filtering (also known as the personality-based approach). Collaborative filtering approaches build a model from a

user's past behavior (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users. This model is then used to predict items (or ratings for items) that the user may have an interest in. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties. These approaches are often combined

2. Methodology

In this section, we will introduce the API used to collect uploaders and users information, and Recommendation System using collaborative filtering.

2.1 API

Since we are aiming to create a video uploader recommendation system, we are going to use the relationship between users and the relationship between uploaders. Relations here means the similarity between each user or uploaders. The similarity will be calculated based on a presentation vector for each uploader and user. For a target user, when we get the knowledge of his/her similar users, we can recommend uploaders their following and similar uploaders to our target user.

To generate the representation vector, we need to collect the number of followers, focusing regions and number of videos for each uploader; and also collect following uploaders for each user. Using APIs provided from Bilibili website, we can gather all features we are going to use.

2.2 Sampling and Dataset

Since the user's id is assigned in order sequence, we randomly sample a dataset with 5.5 million users and filter these users who follow at least 5 uploaders. The final dataset includes 0.45 million users with their following uploaders, we build the recommendation system based on this dataset.

After getting users' information, we generate the candidate set for uploader from all users following. Potentially, without filtering, some uploaders with only little videos or specialized in other interest areas can be recommended, so we collect uploader's following list and uploader's video list. For uploaders the system should recommend, these uploaders should be popular and highly activate, we filter the uploaders with at least 20 videos uploaded and 3000 followers.

Table 1. User with his/her following uploader

userid	upid_0	upid_1	upid_2	upid_3
3164	27070210	197608993	2082556	324844204
5269	326282657	20645285	2929048	1678535
5615	2174185	1606879	810149	569864
6269	12434430	28534515	8521296	12646318
8078	15129631	4899781	4401694	3917479

Finally, the dataset used for the recommendation system consist of two part, users' information and uploaders information. In the user dataset, we have the ID with their following uploaders; in the uploaders dataset, we have the ID with the number of videos uploaded and the focusing region.

Table 2. Uploader with followers, count, and focusing region

upid	follower	count	1	11	119	12	129	13	155	160
97067230	7396	136	0	0	0	0	0	0	0	136
97078324	3345	198	0	0	0	0	0	0	0	198
97101165	71769	265	0	0	0	0	0	0	0	265
97102435	19729	29	0	0	0	0	0	0	0	22
97142412	144891	300	0	0	0	0	0	0	0	298

2.3 Recommendation System

The Recommendation System is generated from three methods. User-Uploader Interest, we measure the similarity between the interest of a user and the focusing region of an uploader, and recommend similar focusing uploaders to the user; User-User, we measure the similarity between users and recommend uploaders having similar focusing region followed by similar users to the target user; Matrix Factorization, generate two latent matrix, and calculate the possibility of a user following an uploader, recommending those high possibility uploaders.

2.3.1 Collaborative Filtering

Collaborative Filtering is the basic model of our recommendation model. It has two different types and our two model use both of them.

User-based Collaborative Filtering uses recommend items by finding similar users to the active user. The most important key here is to choose good similarity measurement for users. Instead of focusing on users, Item-based Collaborative Filtering focus on what items from all the options are more similar to what we know the users enjoy. It can be divided into 2 tasks; Calculate the similarity between items, and the calculation of prediction.

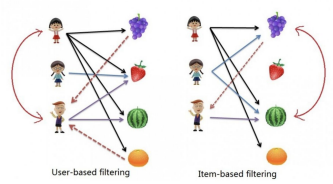

Figure 3. concept of UB-CF and IB-CF

User-Uploader Interest is Item-based Collaborative Filtering, and User-User is User-based Collaborative Filtering.

Table 3. Notation and Terminology.

U_i	The character vector for user *i*
UP_i	The character vector for uploader *i*
v_i	Vector of videos counting in each region for uploader *i*
$Follow_i$	Set of all uploader followed by user *i*
$weight_{ij}$	Importance of uploader *j* followed by user *i* based on followed time

2.3.2 User-Uploader Interest Recommendation

$$UP_i = Norm(v_i)$$

$$U_i = Norm\left(\sum_j weight_{ij} * UP_j\right),\ j \in Follow_i$$

UP_i represent the weight of each region an uploader focus on; the vector will be a good representation of the interest of the uploader. Knowing the uploaders of a user followed, we can generate a vector that represents the interest of that user in each region in bilibili website U_i. Since the uploader recently followed will show more of the current interest of a user, we weight the uploader's importance based on the following time order.

We will recommend the uploader to the user that have similar weight in each region. The similarity will be measure using cosine similarity as:

$$Similarity\left(U_i, UP_j\right) = \frac{U_i \bullet UP_j}{\|U_i\| \times \|UP_j\|}$$

For each target user, we will recommend the uploaders which have the top similarity.

2.3.3 User-User Recommendation

The similarity between users is similar as:

$$Similarity\left(U_i, U_j\right) = \frac{U_i \bullet U_j}{\|U_i\| \times \|U_j\|}$$

For a target user, we will generate top users based on the similarity and recommend not been followed uploaders that followed by the similar users recently. Specifically, we will generate top 10 similar users, and for each one, we will generate top 10 recent followed uploaders, then in this 100 uploaders, we will recommend top 10 uploaders with biggest fan size or with highest video counts, which have not been followed, to the target users. This method is based on the same interest region between different users.

2.3.4 Matrix Factorization Based Recommendation

Matrix factorization algorithms work by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices. The idea behind matrix factorization is to represent users and items in a lower dimensional latent space.

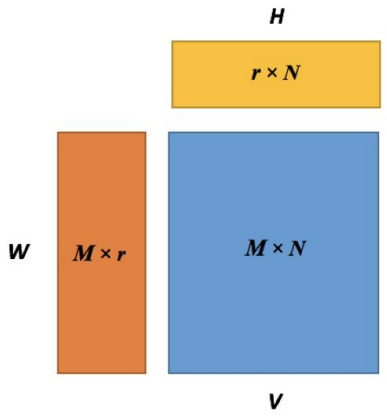

Figure 4. Structure of Matrix Factorization.

As shown in the figure, for the user-item matrix *V*(M*N), each user can be described by *r* attributes or features. For example, some features might be a number that says how much each user likes anime.

Each uploader can be described by an analogous set of *r* attributes or features. To correspond to the above example, some features for the uploader might be a number that says how many percentages of anime upload for the uploader.

If we multiply each feature of the user by the corresponding feature of the uploader and add everything together, this will be a good approximation for the rating the user would give that uploader. For here, it is the possibility of users following the uploader.

Therefore, the goal for matrix factorization is to find low rank matrix *W* and *H,* that minimize the loss between *W*H* and *V*.
The loss we used here is a non-zero squared loss:

$$L_{ij} = l(\mathbf{V}_{ij}, \mathbf{W}_{i*}, \mathbf{H}_{*j}) = (\mathbf{V}_{ij} - \mathbf{W}_{i*}\mathbf{H}_{*j})^2$$

$$L_{NZSL} = \sum_{(i,j)\in\mathbf{Z}} L_{ij}$$

It is the sum of squared loss in each non-zero data point in the matrix.

-Stochastic Gradient Descent Based MF

To realize the matrix factorization, we use stochastic gradient descent to minimize the loss in each iteration and update the factor matrices.

Similar to normal gradient descent, the goal of SGD is to minimize some losses in each iteration; The difference is, SGD randomly choose a single data point in each iteration to calculate derivative of loss, and update the parameter. Correspondingly, in our matrix factorization, for each iteration, we select a single data point (*i, j*) and update the corresponding row *i* of factor matrix *W* and column *j* of factor matrix *H* in the direction of the negative gradient. Repeat the process, until match the termination metric. The update function shows below:

$$\mathbf{W}'_{i*} \leftarrow \mathbf{W}_{i*} - \epsilon_n \frac{\partial}{\partial \mathbf{W}_{i*}} l(\mathbf{V}_{ij}, \mathbf{W}_{i*}, \mathbf{H}_{*j})$$
$$\mathbf{H}'_{*j} \leftarrow \mathbf{H}_{*j} - \epsilon_n \frac{\partial}{\partial \mathbf{H}_{*j}} l(\mathbf{V}_{ij}, \mathbf{W}_{i*}, \mathbf{H}_{*j})$$

In our algorithm, we will form a matrix with the user and uploader representing the following relationship. If a user follows an uploader, then the element will be 1, otherwise will be empty. We will perform SGD base matrix factorization on this matrix. When it converges, we will have the two-factor matrix and we can calculate the possibility of a user will follow an uploader.

2.4 Evaluation Metric

Since the uploaders in Bilibili has an enormous amount, it difficult to evaluate the result using accuracy. Therefore, we will use the cosine similarity to measure the recommendation uploaders with actual uploaders followed in the future to evaluate the performance. We will generate the fusion character vector for the recommended uploaders and actually followed uploaders, and measure the similarity between them. The closer the similarity to 1, the better the recommendation result.

3. Experiments and Result

In this section, we will perform our algorithm on the bilibili crawled data set, and show

the comparison of different recommendation system result.

3.1 Statistical Information

We crawled 2 datasets from the bilibili website; User information dataset contains followed uploaders for each user. Some statistical information is shown in the *table*:

Table 3. Statistical information about users and uploaders.

	Average	Median	Max
User followed UP	10.64	8.32	388
UP follower	42109.44	14381.5	4427386
Videos uploaded	157.63	51	24379

We can find that most users have followed at least 8 uploaders, median lower than average means there is a small part of users follow lots of uploaders. Average videos uploaded is 160, but the median is only 51, means there are some extremely active uploaders.

Uploader information dataset contains followers and focusing region for each uploader. Some statistical information is shown in the *table*:

Table 4. Statistical information about uploaders focusing region.

Region	Anime	Dancing	Life	···	Movie	Music	Game
Average	6.54	2.67	36.34		0.11	10.44	46.71
Max	3367	4800	9895		1126	9481	23656

We can find Life and Game are the most popular focusing regions for uploaders, lots of uploaders upload videos in these two areas. Dancing is more specialized for skillful people to upload and the movie is limited by the authority.

3.2 User-Uploader Recommendation

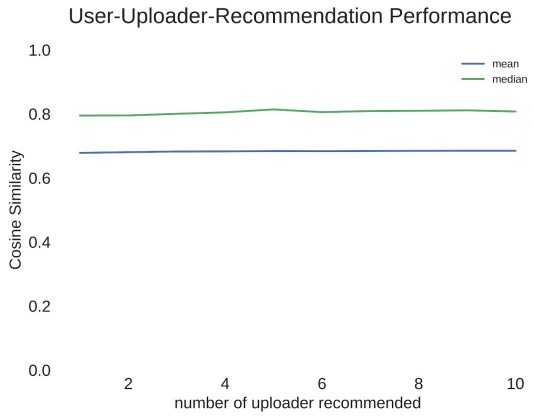

Figure 5. Similarity between recommended and actually followed uploaders.

In figure 5, we can find the similarity is not influenced by the number of uploaders recommended, and have a high result about 0.7 average and 0.8 medians.

3.3 User-User Recommendation

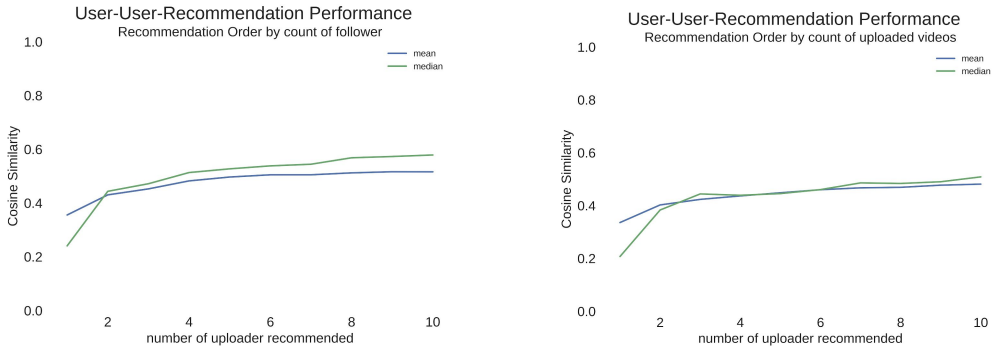

Figure 6. Similarity between recommended and actually followed uploaders.

As the result shown in figure 6, with the number of uploaders recommended increasing, the similarity also increase. The left figure shows recommendation uploaders with a large number of followers, and the right one shows recommendation uploaders with a large number of uploaded videos. As a result, more followers which mean high-quality videos is more important than more videos which mean active uploaders. Users are more interested in following famous uploaders, rather than highly activated but low-quality uploaders.

3.4 Matrix Factorization Recommendation

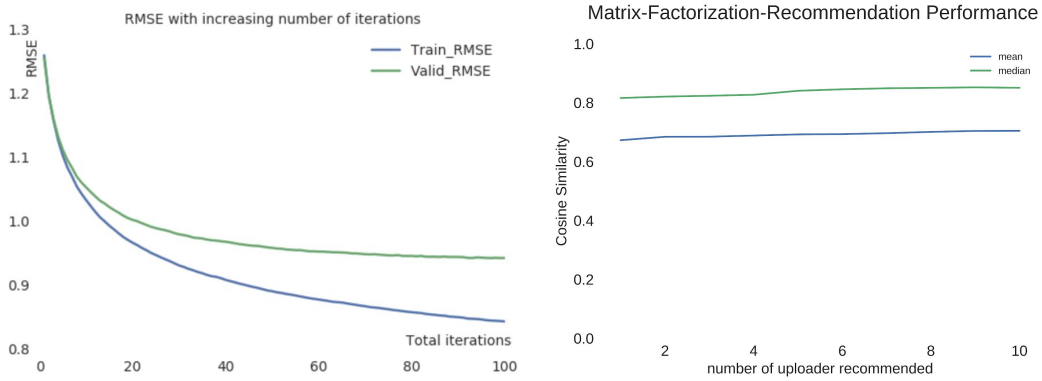

Figure 7. Training and Validation RMSE; Similarity performance.

For each iteration, we will update the factor matrices using non-zero squared loss. The figure shows both the training and validation loss is decreasing as the iteration goes, the training loss is still decreasing, but validation is converged at 40 iterations. The similarity performance is better than all other algorithms which is stable at about 0.83 median and 0.7 average.

3.5 Performance

Selecting the best results from each algorithm, we will get the average similarity 0.5159 for the user-user recommendation, 0.6850 for the user-uploader recommendation and 0.7045 for matrix factorization based recommendation. The matrix factorization method gives the best result as we supposed because today's recommendation system is using this method widely.

4. Conclusion

In our experiment, we recommend bilibili uploaders to users who will probably interested in following them. We first crawl all users' following information and uploaders' focusing region information from bilibili website using APIs provided, basically we drop those uploaders with little videos or little followers. Creating character vector for each user and uploader, we will have a general knowledge about the focusing or interest region for them. Then, we use three collaborative filtering recommendation algorithms, uploader-based, user-based and matrix factorization. For each algorithm, we try to recommend a different number of uploaders to them from 1 to 10 and use the average and median cosine similarity between recommended uploaders and the actually followed uploaders in the test set to measure the performance of all three algorithms.

As the result shows, providing more recommended uploader to users will potentially cover more actual followed in the test set, and improve the performance. Comparing recommendation using the number of the follower with the number of videos uploaded, we find users focus more on the follower number, which means high-quality videos and reputation, rather than a large amount of normal quality videos.

Above all, the matrix factorization has the best performance with similarity about 0.70 and not influenced largely by the recommended number. It means the uploaders actually followed by users in the future will be quite similar to our recommendation. It is important to recommend those uploaders since this will provide a new way for users to know new uploaders and for uploaders to be more active in uploading videos.

5. Future Research

Since we have limited access to the bilibili dataset, we do not have much users' and uploaders' information, e.g. the view history, most recent upload video time. For future research, we may request more data from bilibili then we could use more information to build a better model.

Also, for real-world application, the time for searching and computation should be taken into consideration. We may build a simulated environment to test the consuming time of each model. The best model may no longer be the matrix factorization because it takes a long time to train the model. A trade-off between model performance and consuming time should be made. Some other factors also should be taken consideration, e.g. the number of users used to build the model, the selection of a feature to build the model.

6. References

[1] https://api.bilibili.com/x/relation/followings?vmid=userID

[2] https://api.bilibili.com/x/relation/followers?vmid=uploaderID

[3] https://space.bilibili.com/ajax/member/getSubmitVideos?mid=ID

[4] Aberger, C.R., 2016. Recommender: An analysis of collaborative filtering techniques.

[5] https://blog.insightdatascience.com/explicit-matrix-factorization-als-sgd-and-all-that-jazz-b00e4d9b21ea

[6] https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collab orative-filtering-5d5f375a127f