A Movie Recommender System with implicit feedback

Ni Yongxin (A0231559B)

Zheng Zhiwei (A0231437L)

Peng Junhao (A0231329L)

Contents

1 Business requirements

The development of recommender system has passed nearly 20 years. Recommender System in general can be considered as a system which recommend items to users, the items includes goods, songs, movies, activities and so on. With time passing, users have realized that recommender system has eroded in every part of their life, especially in the digital era. They are more eager to be accurately recommended when they pick up a phone and click into the online shopping website.

Besides purchasing goods from the internet, people's inner world are waited to be satisfy. For the Covid-19 environment globally, the time people stay at home is increasing sharply. They tend to click into some websites, like YouTube, Netflix, Douban, or so on, to watch movies and episodes based on their past taste, which means accurate movie recommendation system meets people's requirements.

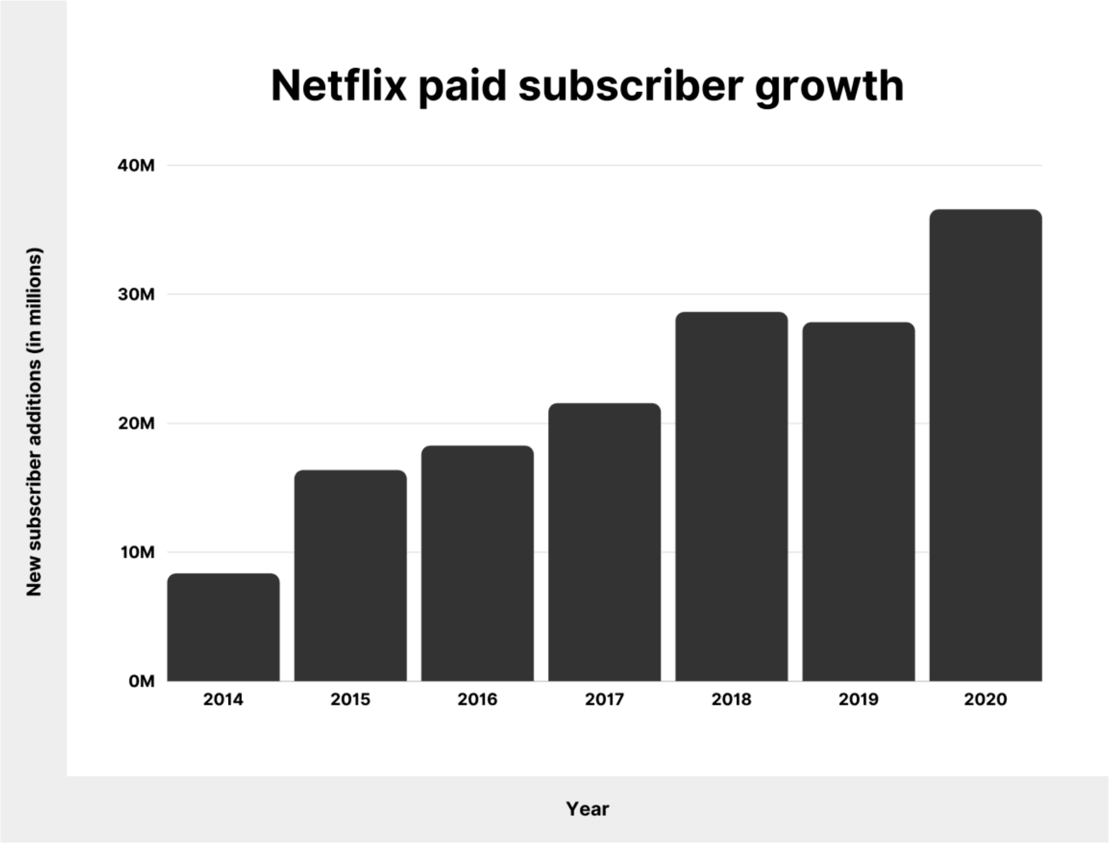


Figure 1 Netflix paid subscriber growth
(source from https://www.affde.com/zh-TW/netflix-users.html)

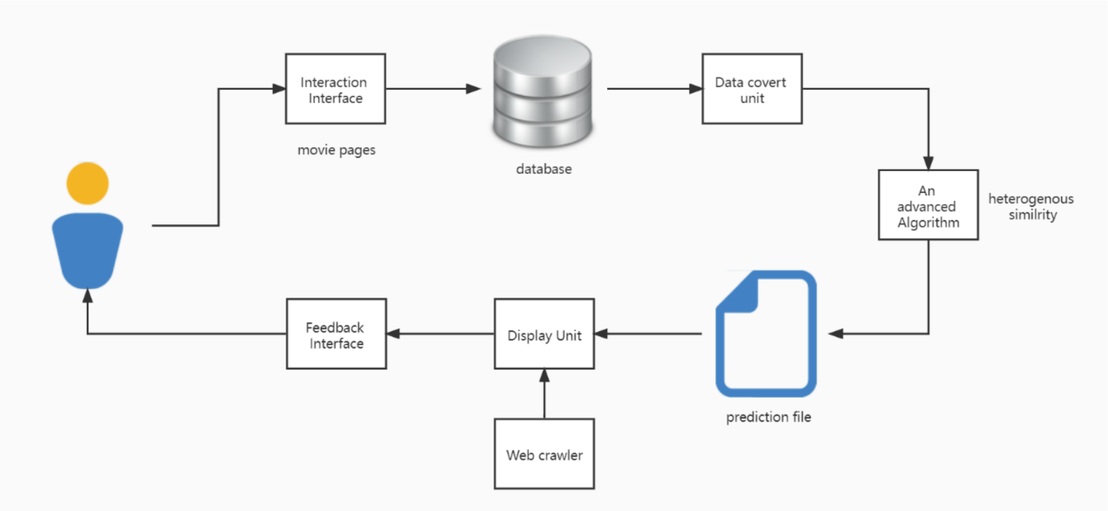
Personalized services can build a bridge between users and merchants. When users use the website more frequently, the system can recommend suitable items to users to meet their demands. A good recommender system can attract more consumers and build a long-term relationship between each other in order to keep consumers from skipping into other websites. Of the multiply personalized recommendation algorithm, collaborative filtering is widely used, which is considered as the most successful recommendation algorithm. Our project mainly focus on using the one-class collaborative filtering to achieve better performance, which will be illustrated in detail below. We will take movie recommendation as an example, in order to make the whole designment more intuitively.

2 Business value

When a person want to see a movie, but no idea what to see, what will he do? Maybe he will ask for his friends' recommendations, open up the chat app to chat with friends who frequently watch movies or directly send a new post in Instagram to ask for most netizens' recommendation, which is considered as social recommendation. Or we may directly search the favorite actor or director in Google, and try to find any movie that we have not watched before, which is considered as content-based filtering in recommendation system. However, when we do not have stable taste and just scan the movie list and may click every item to see a few minutes about this movie and then decide to quit out or not, how the recommendation system accurately work for us?

Our system can do so. Based on the records users have clicked, and the time they stay on the exact movie or episodes, our system will send these records to the database and recommend to theses users directly. Also, it can recommend some items that ranked high on the list to the users regularly. A mature and accurate recommendation system will satisfy audience's taste, which based on the innovation of the algorithm (one-class collaborative filtering).

3 System Structure



Our system architecture is shown in the figure above.

(1) For users, our system provides them with an interaction interface, which presents as a mainstream film website where users can select and watch movies online.

(2) These interaction records generated by users are then recorded in the database.

(3) Because our algorithm only accepts data in the format of (user, item) as input (records with only

the user ID and item ID are called implicit feedback. The advantage of implicit feedback-based algorithms is that they can be easily transplanted to different application fields). Since the database records a lot of attributes, we need to convert the data from the database into the form of (user ID, item ID) for learning model, this process is completed through the “data convert unit”.

(4) The applied algorithm uses a novel heterogeneous similarity to learn from the input data and generates a personalized recommendation list for each user. The prediction results are stored in a prediction file.

(5) We use "display unit" to display the prediction results and user's history of watched movies, during which a web crawler is designed to collect the posters of the target movies. Here a lightweight game engine is used to render the visual interface "feedback interface".

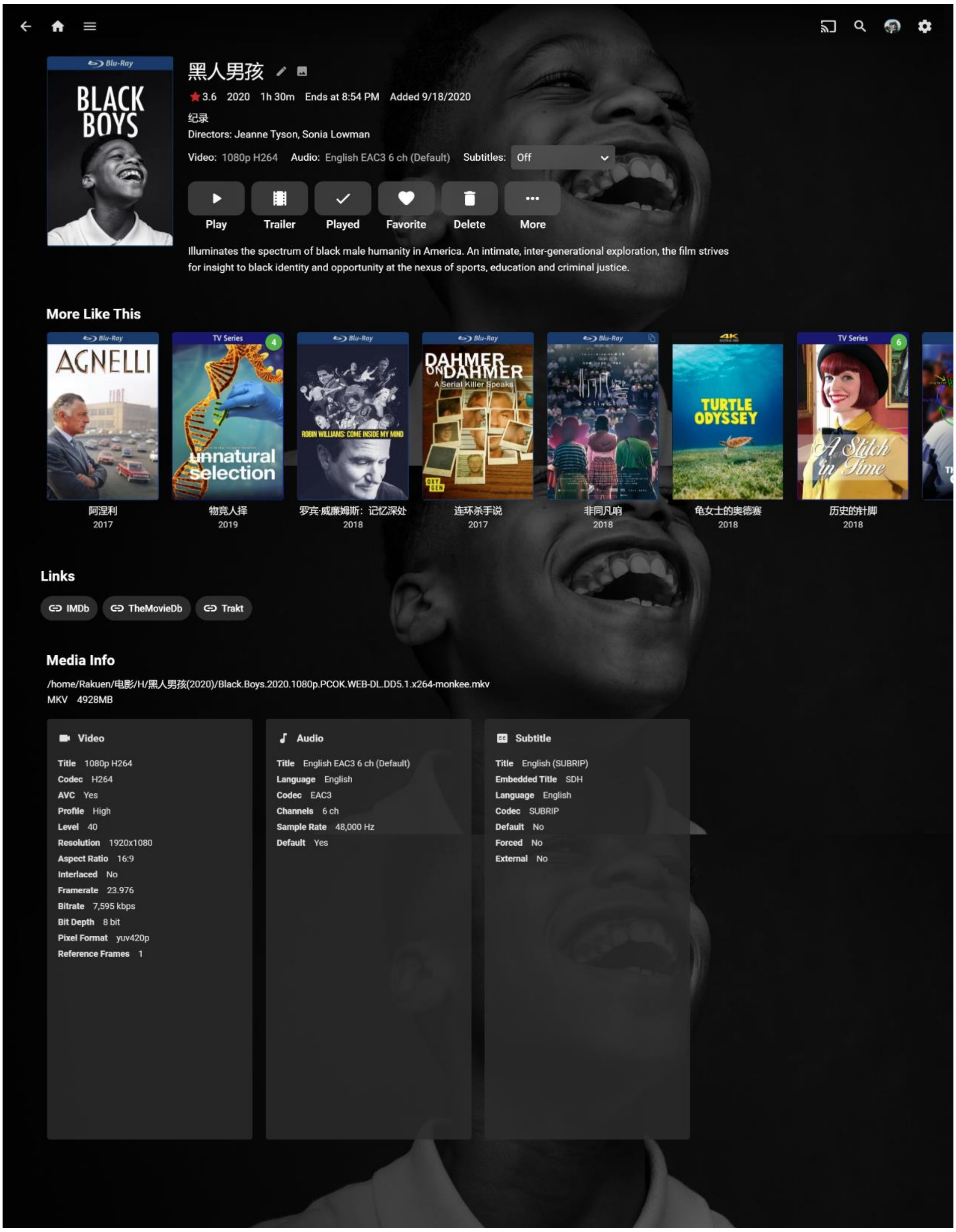
(6) The final display results will be presented to users through the "feedback interface", the additional interface can improve users' trust in our system and increase users' satisfaction.

4 Advantages of Our System

1. The system is equipped with beautiful front-end pages, which support users to not only view the movies but also watch movies online through the web pages we provide.
2. The core algorithm of our system is adopted from a paper published in the journal *neurocomputing* 2020 (https://doi.org/10.1016/j.neucom.2021.05.009, one of our team members is the first-author of that publication), this is an attempt to put a state-of-the-art algorithm into real-world applications, our system is thus more effective even compared with some cutting-edge methods in recent years, since the core of this system is a powerful model that has been recognized by the journal reviewers.
3. Not only the interaction interface as well as the associated database are established in our system, we also use a lightweight game development engine (pygame) to develop a feedback interface to show users their historical items and predicted items given by the system. The user-friendly interface of ours can increase users' satisfaction with our system.

5 Movie Database Construction and Interface Visualization

The process of building a movie database includes collecting movies from various social media sources and matching movie data such as introductions, top cast, more like this movie and related social media links about this movie based on original file name of these materials. In the process of data name matching, some fuzzy filtering and intelligent matching algorithm were applied. At the same time, it is also necessary to obtain relevant data of movie scoring websites.



黑人男孩

3.6 2020 1h 30m Ends at 8:54 PM Added 9/18/2020

纪录

Directors: Jeanne Tyson, Sonia Lowman

Video: 1080p H264 Audio: English EAC3 6 ch (Default) Subtitles: Off

Illuminates the spectrum of black male humanity in America. An intimate, inter-generational exploration, the film strives for insight to black identity and opportunity at the nexus of sports, education and criminal justice.

More Like This

阿涅利 2017
物竞人择 2019
罗宾·威廉姆斯：记忆深处 2018
连环杀手说 2017
非同凡响 2018
龟女士的奥德赛 2018
历史的针脚 2018

Links

Media Info

/home/Rakuen/电影/H/黑人男孩(2020)/Black.Boys.2020.1080p.PCOK.WEB-DL.DD5.1.x264-monkee.mkv
MKV 4928MB

Video

Title 1080p H264
Codec H264
AVC Yes
Profile High
Level 40
Resolution 1920x1080
Aspect Ratio 16:9
Interlaced No
Framerate 23.976
Bitrate 7,595 kbps
Bit Depth 8 bit
Pixel Format yuv420p
Reference Frames 1

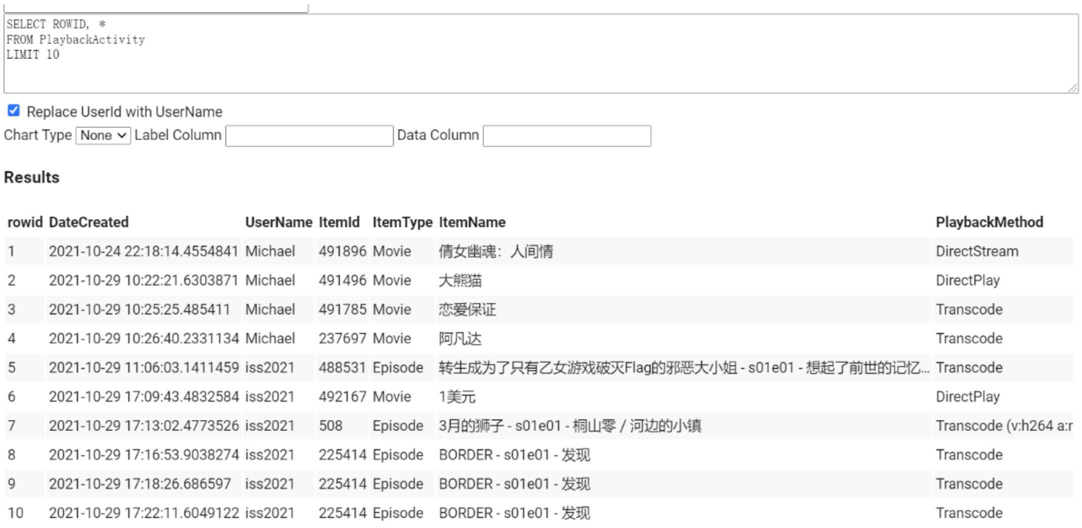
Audio

Title English EAC3 6 ch (Default)
Language English
Codec EAC3
Channels 6 ch
Sample Rate 48,000 Hz
Default Yes

Subtitle

Title English (SUBRIP)
Embedded Title SDH
Language English
Codec SUBRIP
Default No
Forced No
External No

Matching each movie data



```
SELECT ROWID, *
FROM PlaybackActivity
LIMIT 10
```

☑ Replace UserId with UserName

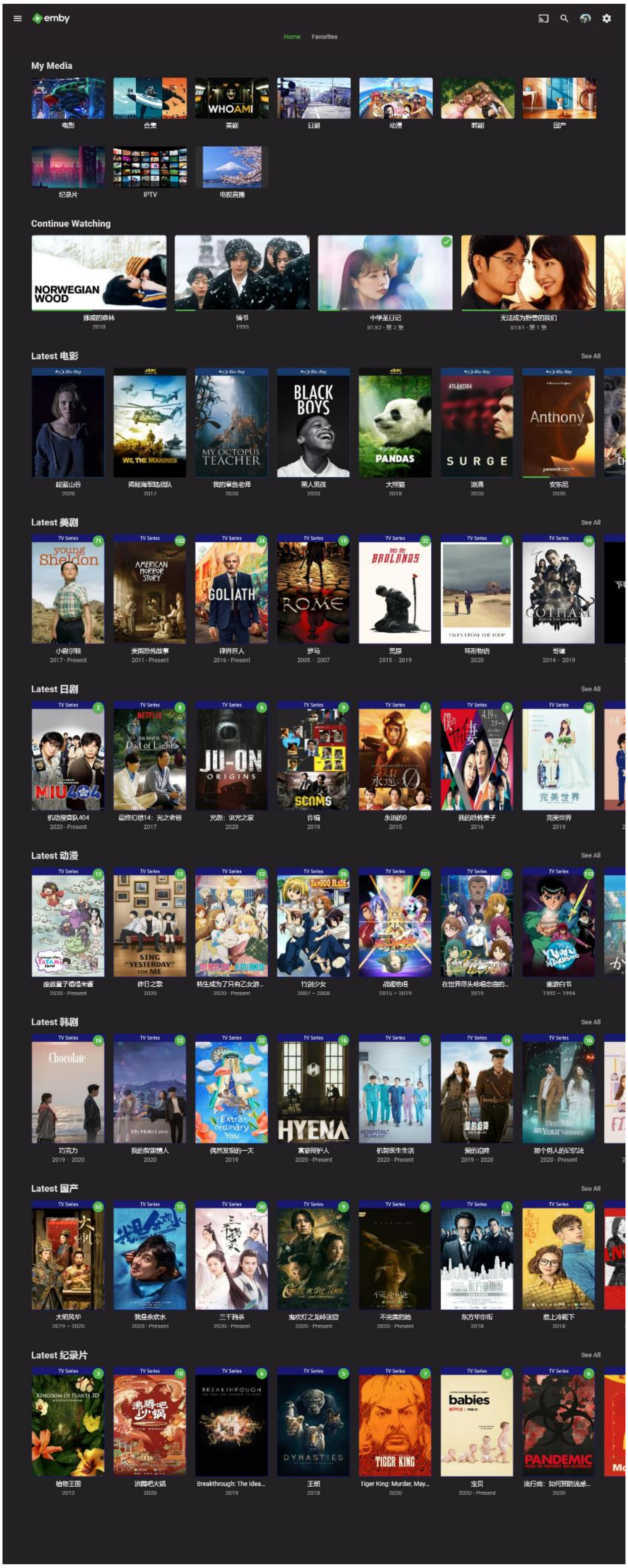
Chart Type None Label Column Data Column

Results

rowid	DateCreated	UserName	ItemId	ItemType	ItemName	PlaybackMethod
1	2021-10-24 22:18:14.4554841	Michael	491896	Movie	倩女幽魂：人间情	DirectStream
2	2021-10-29 10:22:21.6303871	Michael	491496	Movie	大熊猫	DirectPlay
3	2021-10-29 10:25:25.485411	Michael	491785	Movie	恋爱保证	Transcode
4	2021-10-29 10:26:40.2331134	Michael	237697	Movie	阿凡达	Transcode
5	2021-10-29 11:06:03.1411459	iss2021	488531	Episode	转生成为了只有乙女游戏破灭Flag的邪恶大小姐 - s01e01 - 想起了前世的记忆...	Transcode
6	2021-10-29 17:09:43.4832584	iss2021	492167	Movie	1美元	DirectPlay
7	2021-10-29 17:13:02.4773526	iss2021	508	Episode	3月的狮子 - s01e01 - 桐山零 / 河边的小镇	Transcode (v:h264 a:r
8	2021-10-29 17:16:53.9038274	iss2021	225414	Episode	BORDER - s01e01 - 发现	Transcode
9	2021-10-29 17:18:26.686597	iss2021	225414	Episode	BORDER - s01e01 - 发现	Transcode
10	2021-10-29 17:22:11.6049122	iss2021	225414	Episode	BORDER - s01e01 - 发现	Transcode

Movie Database Interface (SQL supported)

The process of Interface Visualization includes deploying Linux server with Linux shell language, build a database, and establish a front-end display interface (https://junhao.io:8096 Username: iss2021 Password: Aa123456!), the front-end display interface is powered by an open-source software named Emby. We store all our movie data in G Suites(An enterprise version of Google Drive), then mount it on our Linode's Linux server. Both the database construction process and data matching process were completed while debugging through the SSH commands.



My Media

电影
合集
美剧
日剧
动漫
韩剧
国产
纪录片
IPTV
电视直播

Continue Watching

挪威的森林
2010

情书
1995

中学圣日记
S1:E2 - 第 2 集

无法成为野兽的我们
S1:E1 - 第 1 集

Latest 电影

See All

靛蓝山谷
2020

揭秘海军陆战队
2017

我的章鱼老师
2020

黑人男孩
2020

大熊猫
2018

浪涌
2020

安东尼
2020

Latest 美剧

See All

小谢尔顿
2017 - Present

美国恐怖故事
2011 - Present

律界巨人
2016 - Present

罗马
2005 – 2007

荒原
2015 – 2019

环形物语
2020

哥谭
2014 – 2019

Latest 日剧

See All

机动搜查队404
2020 - Present

最终幻想14：光之老爸
2017

咒怨：诅咒之家
2020

诈骗
2019

永远的0
2015

我的恐怖妻子
2016

完美世界
2019

Latest 动漫

See All

座敷童子榻榻米酱
2020 - Present

昨日之歌
2020

转生成为了只有乙女游...
2020 - Present

竹剑少女
2007 – 2008

战姬绝唱
2015 – 2019

在世界尽头咏唱恋曲的...
2019

幽游白书
1992 – 1994

Latest 韩剧

See All

巧克力
2019 – 2020

我的智能情人
2020

偶然发现的一天
2019

富豪辩护人
2020 - Present

机智医生生活
2020 - Present

爱的迫降
2019 – 2020

那个男人的记忆法
2020 - Present

Latest 国产

See All

大明风华
2019 – 2020

我是余欢水
2020 - Present

三千鸦杀
2020 - Present

鬼吹灯之龙岭迷窟
2020 - Present

不完美的她
2020 - Present

东方华尔街
2018

惹上冷殿下
2018

Latest 纪录片

See All

植物王国
2012

沸腾吧火锅
2020

Breakthrough: The Idea...
2019

王朝
2018

Tiger King: Murder, May...
2020

宝贝
2020 - Present

流行病：如何预防流感...
2020

6 Core Algorithm of Our System

6.1 Problem definition

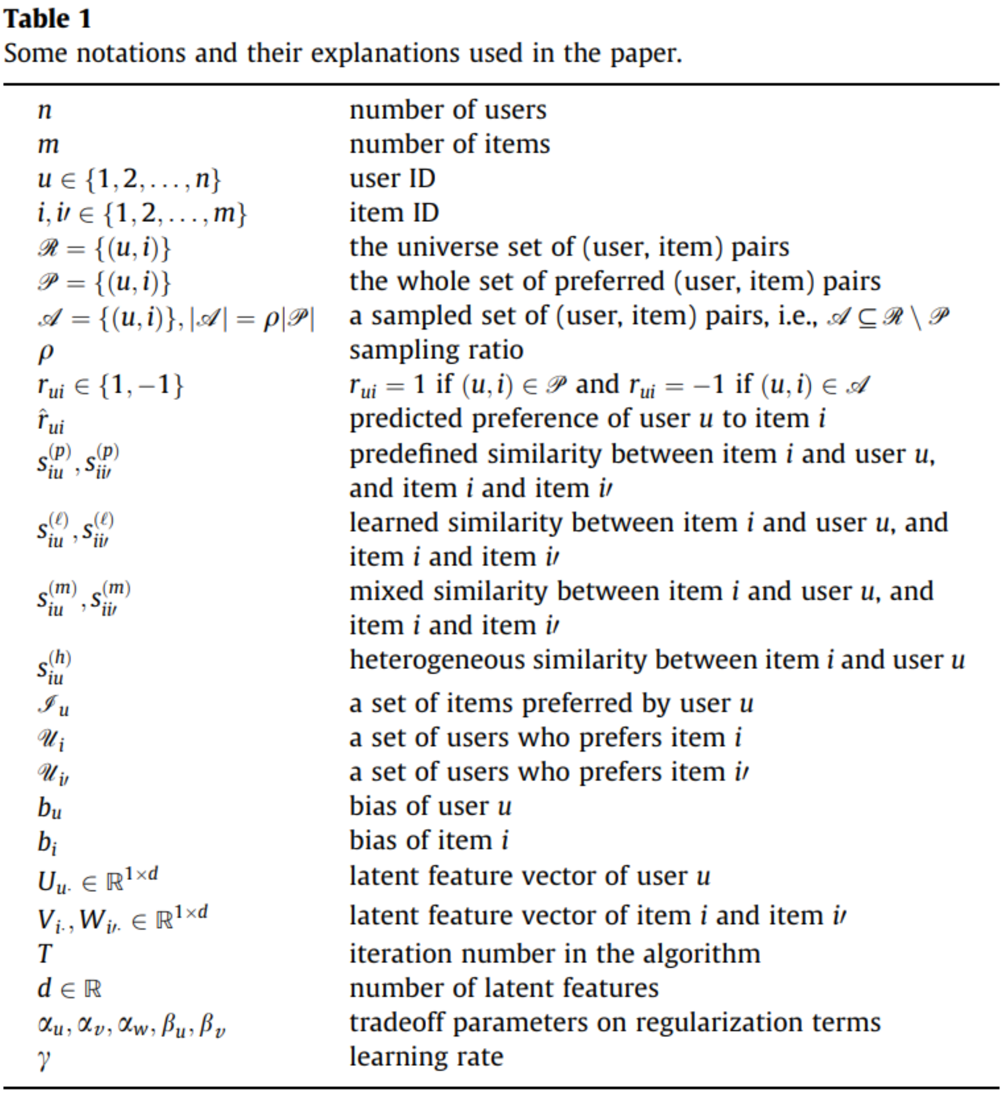


Table 1
Some notations and their explanations used in the paper.

n	number of users
m	number of items
$u \in \{1,2,\ldots,n\}$	user ID
$i, i' \in \{1,2,\ldots,m\}$	item ID
$\mathscr{R} = \{(u,i)\}$	the universe set of (user, item) pairs
$\mathscr{P} = \{(u,i)\}$	the whole set of preferred (user, item) pairs
$\mathscr{A} = \{(u,i)\}, \lvert\mathscr{A}\rvert = \rho\lvert\mathscr{P}\rvert$	a sampled set of (user, item) pairs, i.e., $\mathscr{A} \subseteq \mathscr{R} \setminus \mathscr{P}$
ρ	sampling ratio
$r_{ui} \in \{1,-1\}$	$r_{ui} = 1$ if $(u,i) \in \mathscr{P}$ and $r_{ui} = -1$ if $(u,i) \in \mathscr{A}$
$\hat{r}_{ui}$	predicted preference of user u to item i
$s_{iu}^{(p)}, s_{ii'}^{(p)}$	predefined similarity between item i and user u, and item i and item i'
$s_{iu}^{(\ell)}, s_{ii'}^{(\ell)}$	learned similarity between item i and user u, and item i and item i'
$s_{iu}^{(m)}, s_{ii'}^{(m)}$	mixed similarity between item i and user u, and item i and item i'
$s_{iu}^{(h)}$	heterogeneous similarity between item i and user u
$\mathscr{I}_u$	a set of items preferred by user u
$\mathscr{U}_i$	a set of users who prefers item i
$\mathscr{U}_{i'}$	a set of users who prefers item i'
b_u	bias of user u
b_i	bias of item i
$U_{u\cdot} \in \mathbb{R}^{1\times d}$	latent feature vector of user u
$V_{i\cdot}, W_{i'\cdot} \in \mathbb{R}^{1\times d}$	latent feature vector of item i and item i'
T	iteration number in the algorithm
$d \in \mathbb{R}$	number of latent features
$\alpha_u, \alpha_v, \alpha_w, \beta_u, \beta_v$	tradeoff parameters on regularization terms
γ	learning rate

In a typical recommendation problem with implicit feedback, we have n users, m items and their associated implicit feedback in the form of (user, item) pairs recorded in a set {u,i}. Our goal is to learn users' preferences and then recommend a personalized ranked list of items for each user u. We illustrate the studied problem, i.e., the input and the output, in the left part and right part of Fig. 2, respectively. Notice that the left part is an example of the interaction records in a data and the right part shows the recommendation generated by our trained model. We put some notations in Table 1.

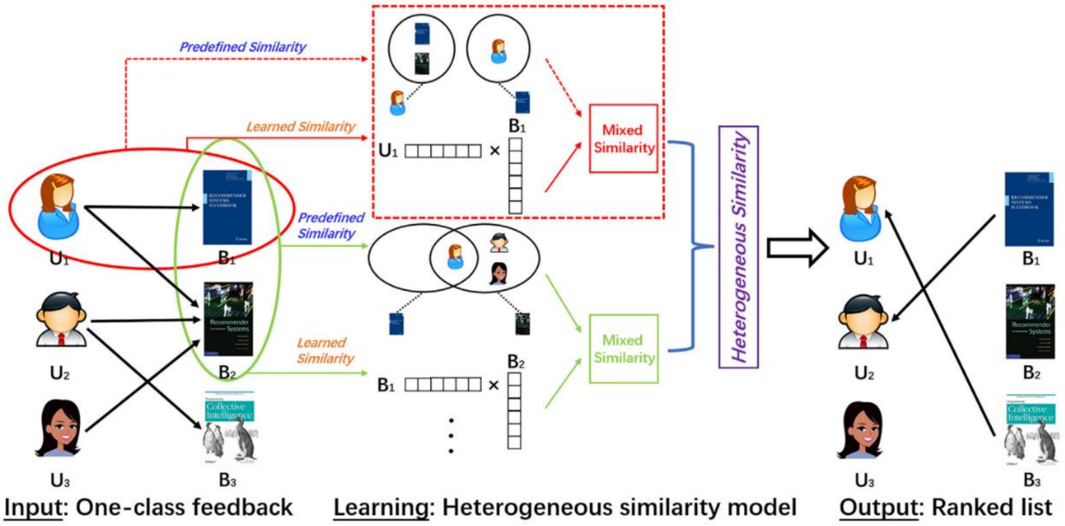


Figure 2 Problem definition diagram

6.2 Model Basis: Predefined Similarity

For modeling implicit feedback in recommender systems, most neighborhood-based algorithms leverage a certain predefined similarity measurement, e.g., Jaccard index or cosine similarity, for the purpose of constructing the neighboring sets of users or items. For instance, the typical cosine similarity is commonly used in item-based collaborative filtering, which measures the similarity between item i and item i' as follows:

$$s_{ii'}^{(p)} = \frac{|\mathcal{U}_i \cap \mathcal{U}_{i'}|}{\sqrt{|\mathcal{U}_i|}\sqrt{|\mathcal{U}_{i'}|}}, \qquad (1)$$

where |Ui ∩Ui'| denotes the number of users who prefer both item I and item i', and |Ui| and |Ui'| denote the number of users who prefer item i and item i', respectively.

6.3 Model Basis: Learned Similarity

Learned similarity based on latent feature vectors is able to capture the global relations between users (or items) so as to alleviate the cold-start problem because of the good transitivity. For instance, an algorithm called FISM is a well-known similarity learning method that can learn the similarity between the preferred items and the candidate items via some latent feature vectors. The learned similarity between item i and item i' is as follows.

$$s_{ii'}^{(\ell)} = V_{i\cdot}W_{i'\cdot}^{T}, \qquad (2)$$

where the learned similarity is factored into the inner product of two items' feature vectors Vi, $W_{i'}^{T}$.

It is usually more accurate than the predefined similarity mentioned above for its good transitivity.

6.4 Model Basis: Mixed Similarity

To inherit the merit of the predefined similarity and the learned similarity, mixed similarity is proposed, which is able to recommend items more accurately. Generally, relations existing among items are known to be two kinds, i.e., local relations and global relations, which are denoted by the predefined similarity $s_{ii'}^{(p)}$ and the learned similarity $s_{ii'}^{(l)}$ respectively. To excavate both the local and global relations and calculate the similarity between neighboring items, mixed similarity took $s_{ii'}^{(l)}$ as the core of their mixed similarity model because previous works have shown that $s_{ii'}^{(l)}$ is superior to the other one in most cases. In particular, mixed similarity made $s_{ii'}^{(p)}$ an enhancement factor of $s_{ii'}^{(l)}$ via (1-λ)+ λ$s_{ii'}^{(p)}$ as a contribution parameter of $s_{ii'}^{(l)}$. The new mixed similarity model usually has better performance in empirical studies, which can be represented as follows.

$$s_{ii'}^{(m)} = ((1-\lambda) + \lambda s_{ii'}^{(p)})s_{ii'}^{(\ell)} = (1-\lambda)s_{ii'}^{(\ell)} + \lambda s_{ii'}^{(p)}s_{ii'}^{(\ell)} \qquad (3)$$

where λ is a tradeoff parameter. Notice that the similarity above suffers from the shortcoming of the simple linear combination of two types of similarity measurements. Moreover, it is time consuming to tune the tradeoff between the two types of similarities. To sum up, a method that merely relies on one single type of similarity, i.e., predefined similarity or learned similarity, is inferior to the one uses the mixed similarity for the limitations mentioned above.

6.5 Our Solution: Heterogenous Similarity

In order to capture rich relations, we propose to make use of two types of mixed similarities in our heterogeneous similarity, including a mixed similarity between an item i and a user u, i.e. $s_{iu'}^{(m)}$, and a mixed similarity between an item i and an item i', i.e., $s_{ii'}^{(m)}$. Specifically, edified by Eq, we use a broad concept of similarity (or affinity) in finding some likely-to-preferred items for users. Moreover, with such a broad concept, we reach a unified form of mixed similarity, including the item-user similarity and item-item similarity, thus we have $s_{iu}^{(m)} = s_{iu}^{(l)} + s_{iu}^{(p)}s_{iu}^{(l)}$ and $s_{ii'}^{(m)} = s_{ii'}^{(l)} + s_{ii'}^{(p)}s_{ii'}^{(l)}$ for the two types of mixed similarities. Notice that the predefined similarity s

$s_{iu}^{(p)} = \frac{|\mathscr{U}_i \cap \mathscr{I}_u|}{\sqrt{|\mathscr{U}_i|}\sqrt{|\mathscr{I}_u|}} = 0,$, and thus we have a reduced form of the mixed similarity between an item i and a user u, i.e., $s_{iu}^{(m)} = s_{iu}^{(l)}$, which is commonly used in matrix factorization-based methods. As another notice, our mixed similarity between two items does not include a tradeoff parameter, which makes it easy for configuration in empirical studies and real deployment, in comparison with that in.

For each (user, item) pair, our task is now to estimate the heterogeneous similarity $s_{iu}^{(h)}$ between item i and user u. There is only one mixed similarity $s_{iu}^{(m)}$ between item i and user u, but there may be a large number of mixed similarity $s_{ii\prime}^{(m)}$ between item i and its related items i's, and $s_{iu}^{(h)}$ is accumulated by all these similarities. The number of $s_{ii\prime}^{(m)}$ depends on how many items are interacted with by user u (the item i itself is excluded), reflected as the size of $\mathscr{I}_u \setminus \{i\}$.

Finally, we reach our heterogeneous similarity between item i and user u,

$$\begin{aligned} s_{iu}^{(h)} &= s_{iu}^{(m)} + \sum_{i' \in \mathscr{I}_u \setminus \{i\}} s_{ii'}^{(m)} \\ &= s_{iu}^{(\ell)} + \sum_{i' \in \mathscr{I}_u \setminus \{i\}} \left(s_{ii'}^{(\ell)} + s_{ii'}^{(p)} s_{ii'}^{(\ell)}\right). \end{aligned} \tag{4}$$

We can see that the proposed heterogeneous similarity $s_{iu}^{(h)}$ is different from the mixed similarity in. In particular, we define the similarity between an item i and a user u instead of between two items only, which is a common treatment in statistical recommendation methods.

As we can see, every time before we construct $s_{iu}^{(h)}$, we obtain a target user u, a target item i and its related items i's. For item i and item i', we can calculate their predefined similarity and learned similarity simultaneously. Combining these two similarities appropriately will not only make up for the shortcomings of only capturing the local relations, but also in turn enhance the effect of the learned similarity. The experimental results of P-FMSM also show that constructing more than one type of similarity and effectively combining them could lead to the improvement of the recommendation accuracy. Based on this, it is reasonable to further construct a kind of user-item similarity since it will include more connections established from the implicit feedback.

6.6 Our Solution: Preference Learning

We adopt the flexible pointwise preference learning paradigm instead of the pairwise preference learning one. The former focuses on modeling each user's preference on a single item rather than a pair of items, which is more flexible in sampling the negative items with a proper configuration. We will also study this issue in our empirical studies. Hence, we call our method pointwise factored heterogeneous similarity model (Poi-FHSM). In pointwise preference learning, we aim to maximize the overall log-likelihood of the predicted preference shown in Eq., and have the following optimization problem,

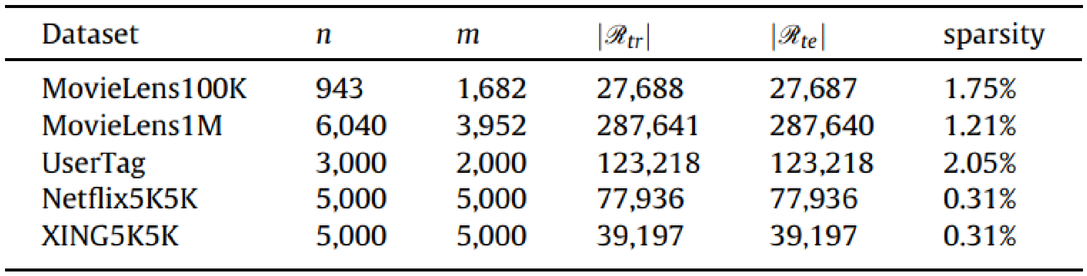
$$\min_{\Theta} \sum_{(u,i)\in\mathcal{P}\cup\mathcal{A}} f_{ui}, \qquad (7)$$

where $f_{ui} = \log(1+\exp(-r_{ui}\hat{r}_{ui})) + \frac{\alpha_u}{2}||U_{u\cdot}||_F^2 + \frac{\alpha_v}{2}||V_{i\cdot}||_F^2 + \frac{\alpha_w}{2}\sum_{i'\in\mathcal{I}_u\setminus\{i\}}||W_{i'\cdot}||_F^2 + \frac{\beta_u}{2}b_u^2 + \frac{\beta_v}{2}b_i^2$ is the objective function for each (user, item) pair (u, i), and $\Theta = \{U_{u\cdot}, V_{i\cdot}, W_{i'\cdot}, b_u, b_i | u = 1,\ldots,n; i = 1,\ldots,m\}$ denotes the set of model parameters to be learned. Notice that P is the whole set of observed (user, item) pairs, a sampled set of (user, item) pairs, and $r_{ui} = 1$ if $(u,i)\in\mathcal{P}$ and $r_{ui} = -1$ if $(u,i)\in\mathcal{A}$.

7 Evaluation

Five public datasets have been adopted to validate the core algorithm of our system, including MovieLens 100K (ML100K) and MovieLens 1M (ML1M) from the famous GroupLens research group,1 UserTag about users' tagging behaviors, Netflix5K5K about users' rating behaviors from a subset of the Netflix competition dataset,2 and XING5K5K about users' behaviors from a job recommendation site.3 Each of the five datasets is randomly split into a training data and a test data equally. To be specific, the statistics of the processed datasets are as follows: (i) ML100K consists of 943 users, 1,682 items, 27,688 training records and 27,687 test records; (ii) ML1M consists of 6,040 users, 3,952 items, 287,641 training records and 287,640 test records; (iii) UserTag consists of 3,000 users, 2,000 items, 123,218 training records and 123,218 test records; (iv) Netflix5K5K consists of 5,000 users, 5,000 items, 77,936 training records and 77,936 test records; and (v) XING5K5K consists of 5,000 users, 5,000 items, 39,197 training records and 39,197 test records. Each dataset contains three copies of training records and test records in the form of (user, item) pairs. For parameter tuning, we use the first copy of each dataset to construct a validation data by sampling n (i.e., the number of users) records from the corresponding training records. The statistics of each dataset is shown in Table 2.

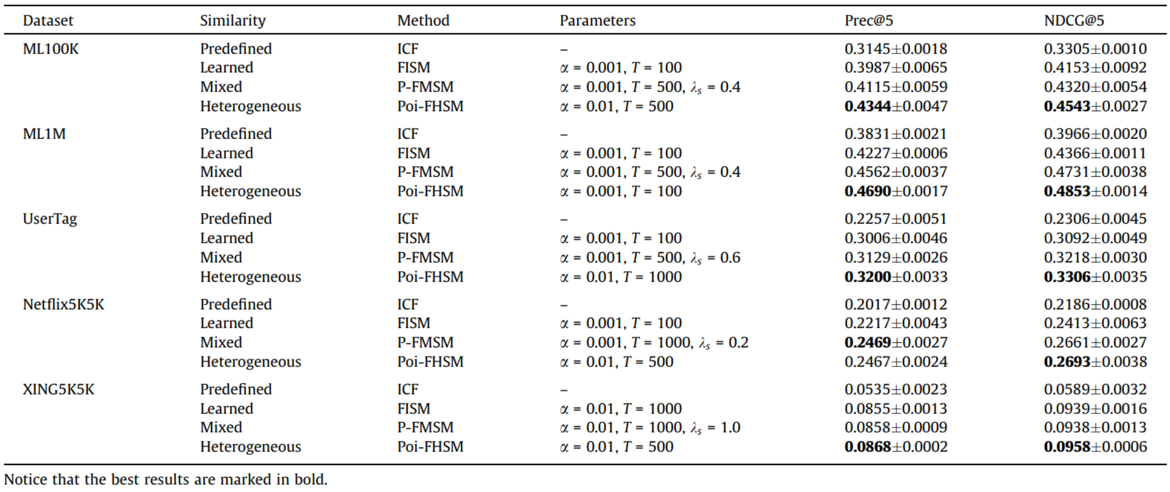
Table 2



Dataset	n	m	$\lvert\mathcal{R}_{tr}\rvert$	$\lvert\mathcal{R}_{te}\rvert$	sparsity
MovieLens100K	943	1,682	27,688	27,687	1.75%
MovieLens1M	6,040	3,952	287,641	287,640	1.21%
UserTag	3,000	2,000	123,218	123,218	2.05%
Netflix5K5K	5,000	5,000	77,936	77,936	0.31%
XING5K5K	5,000	5,000	39,197	39,197	0.31%

In order to study the effectiveness of our model (named as Poi-FHSM) in a direct comparison with the closely related works, we include some representative methods for modeling implicit feedback in recommender systems. In particular, we include two series of methods, including the similarity-based methods and the deep learning-based methods.

Table 3



Dataset	Similarity	Method	Parameters	Prec@5	NDCG@5
ML100K	Predefined	ICF	–	0.3145±0.0018	0.3305±0.0010
	Learned	FISM	α = 0.001, T = 100	0.3987±0.0065	0.4153±0.0092
	Mixed	P-FMSM	α = 0.001, T = 500, λ_s = 0.4	0.4115±0.0059	0.4320±0.0054
	Heterogeneous	Poi-FHSM	α = 0.01, T = 500	**0.4344**±0.0047	**0.4543**±0.0027
ML1M	Predefined	ICF	–	0.3831±0.0021	0.3966±0.0020
	Learned	FISM	α = 0.001, T = 100	0.4227±0.0006	0.4366±0.0011
	Mixed	P-FMSM	α = 0.001, T = 500, λ_s = 0.4	0.4562±0.0037	0.4731±0.0038
	Heterogeneous	Poi-FHSM	α = 0.001, T = 100	**0.4690**±0.0017	**0.4853**±0.0014
UserTag	Predefined	ICF	–	0.2257±0.0051	0.2306±0.0045
	Learned	FISM	α = 0.001, T = 100	0.3006±0.0046	0.3092±0.0049
	Mixed	P-FMSM	α = 0.001, T = 500, λ_s = 0.6	0.3129±0.0026	0.3218±0.0030
	Heterogeneous	Poi-FHSM	α = 0.01, T = 1000	**0.3200**±0.0033	**0.3306**±0.0035
Netflix5K5K	Predefined	ICF	–	0.2017±0.0012	0.2186±0.0008
	Learned	FISM	α = 0.001, T = 100	0.2217±0.0043	0.2413±0.0063
	Mixed	P-FMSM	α = 0.001, T = 1000, λ_s = 0.2	**0.2469**±0.0027	0.2661±0.0027
	Heterogeneous	Poi-FHSM	α = 0.01, T = 500	0.2467±0.0024	**0.2693**±0.0038
XING5K5K	Predefined	ICF	–	0.0535±0.0023	0.0589±0.0032
	Learned	FISM	α = 0.01, T = 1000	0.0855±0.0013	0.0939±0.0016
	Mixed	P-FMSM	α = 0.01, T = 1000, λ_s = 1.0	0.0858±0.0009	0.0938±0.0013
	Heterogeneous	Poi-FHSM	α = 0.01, T = 500	**0.0868**±0.0002	**0.0958**±0.0006

Notice that the best results are marked in bold.

8 User-friendly interface designment

After we acquired the user and item data from the database, and successfully go through the one-class collaborative filtering algorithm, although we can see the direct chart that shows the accuracy. It illustrates the efficiency that every model achieve, the audience still cannot understand, they need to see the intuitive and exact recommendations to confirm that our system has achieved better performance than others. Therefore, we design a user-friendly interface, which will help users experiences the strengths of our system more directly.

When we run the algorithm which designed before, we can see exact movies which are recommended to specific user. The train picture is shown below, which illustrates one specific user's taste and his favorite top 15 movies in records.

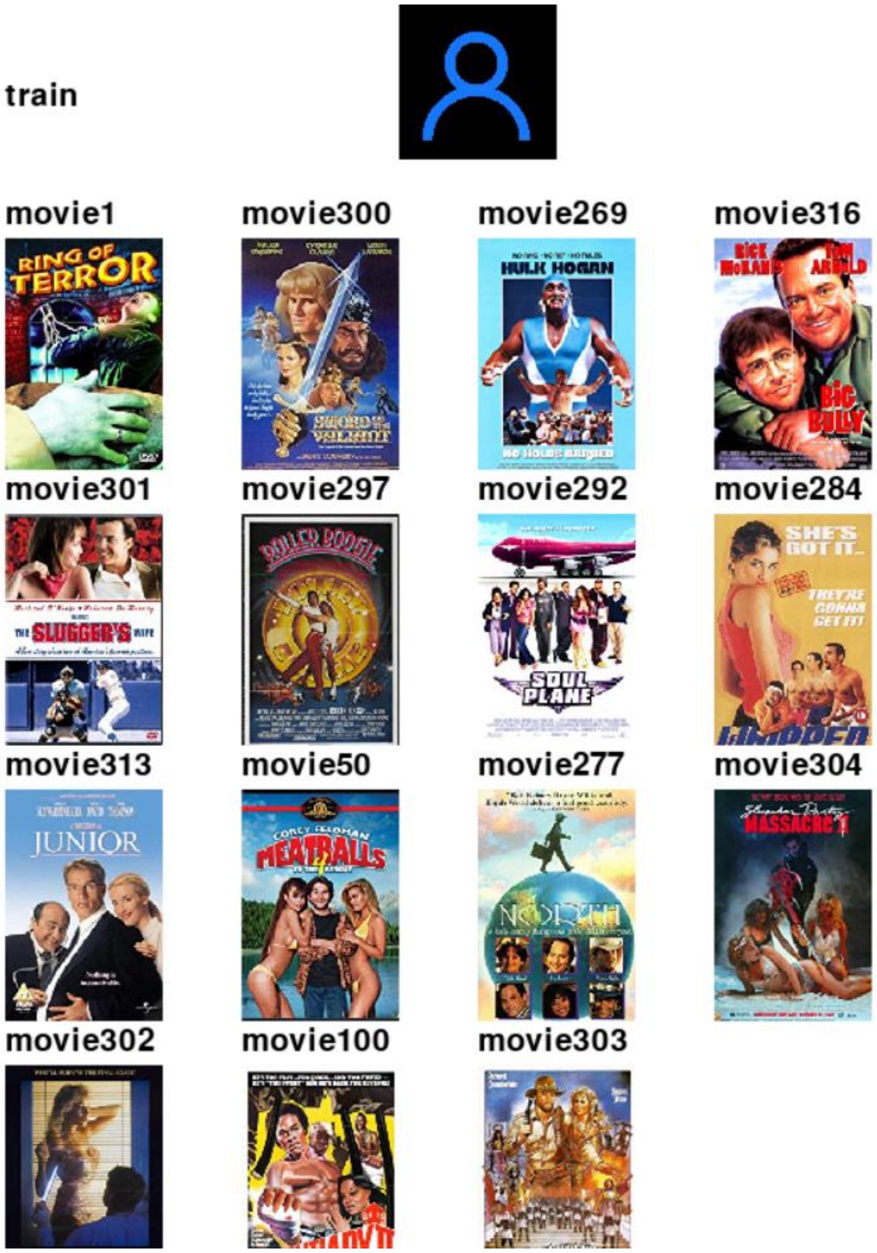
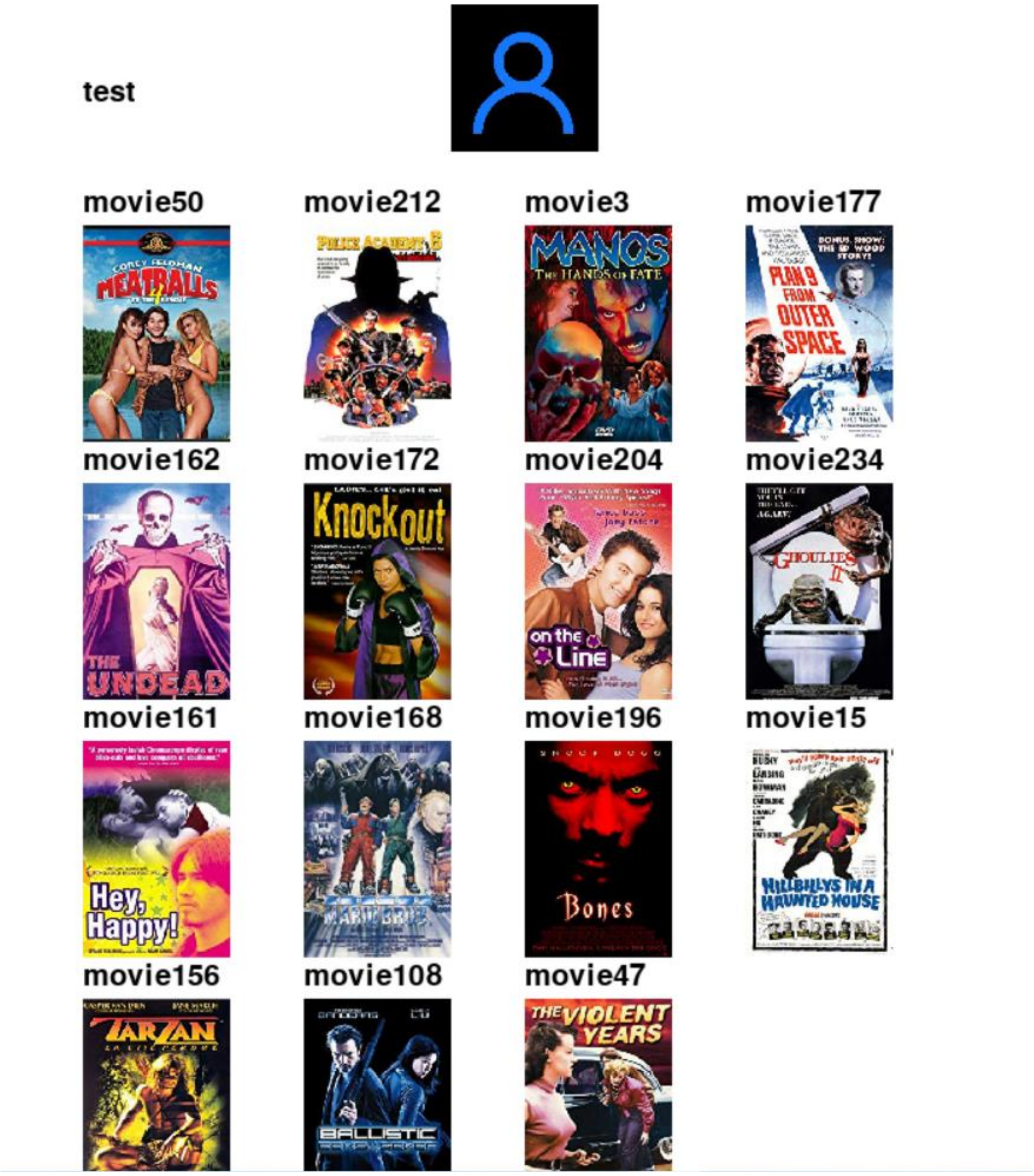


Figure 3 One specific user's top15 movie records

Limited to the size of the picture, we can only show top 15 movies that this specific user used to watch. Also, the name or the ID of the user is not allowed to be shown when designing this user-friendly interface. The movie ID will make the whole picture more directly instead of the long movie name. And several posters will also be demonstrated when making the specific report to the specific user.

Based on the train and test records, we use our one-class collaborative filtering to predict the movies that users would like, and then compare them to the test records that the user exactly watched before, so as to see the performance of our system directly.



test

movie50 movie212 movie3 movie177

movie162 movie172 movie204 movie234

movie161 movie168 movie196 movie15

movie156 movie108 movie47

Figure 4 Top 15 test records

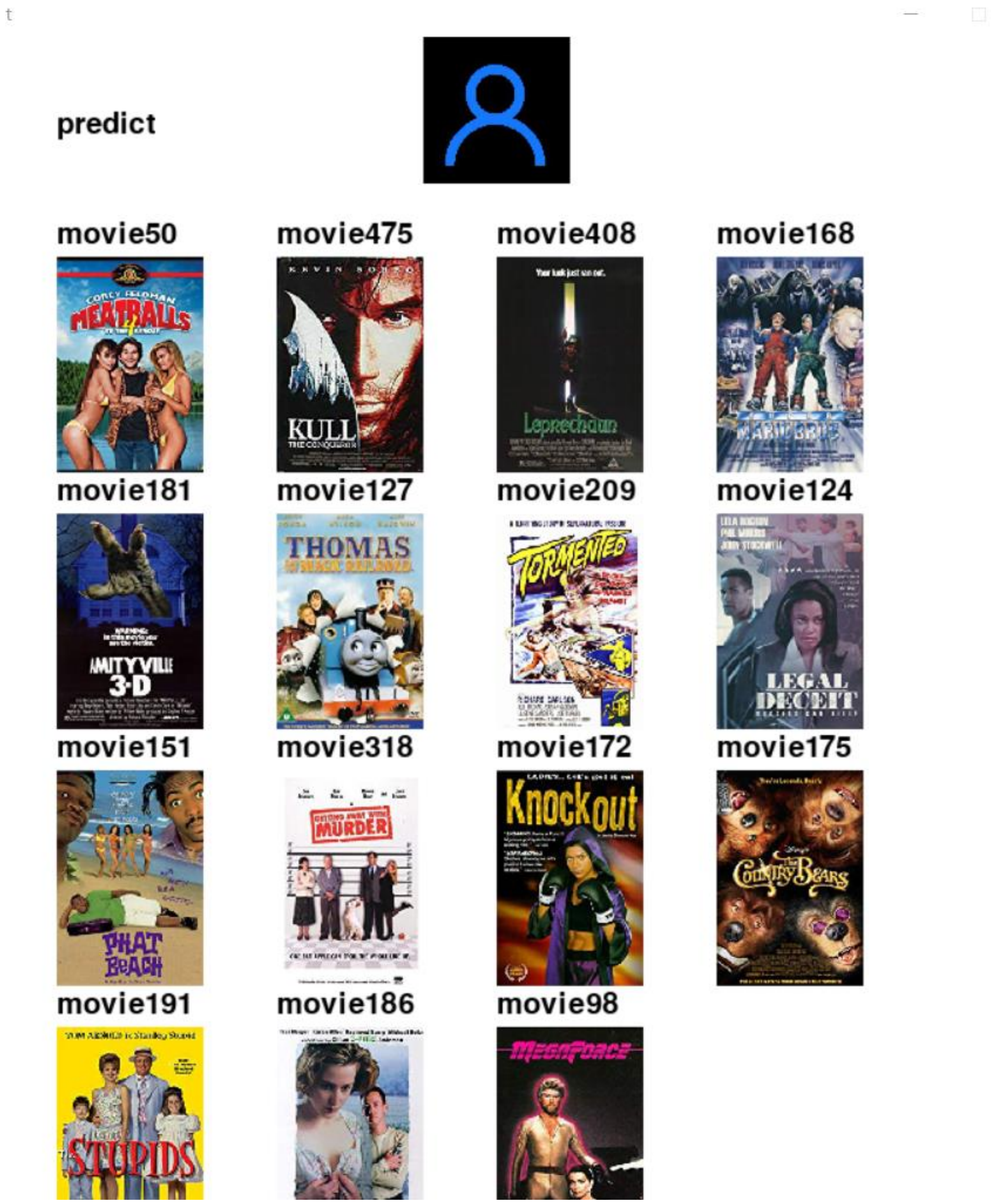


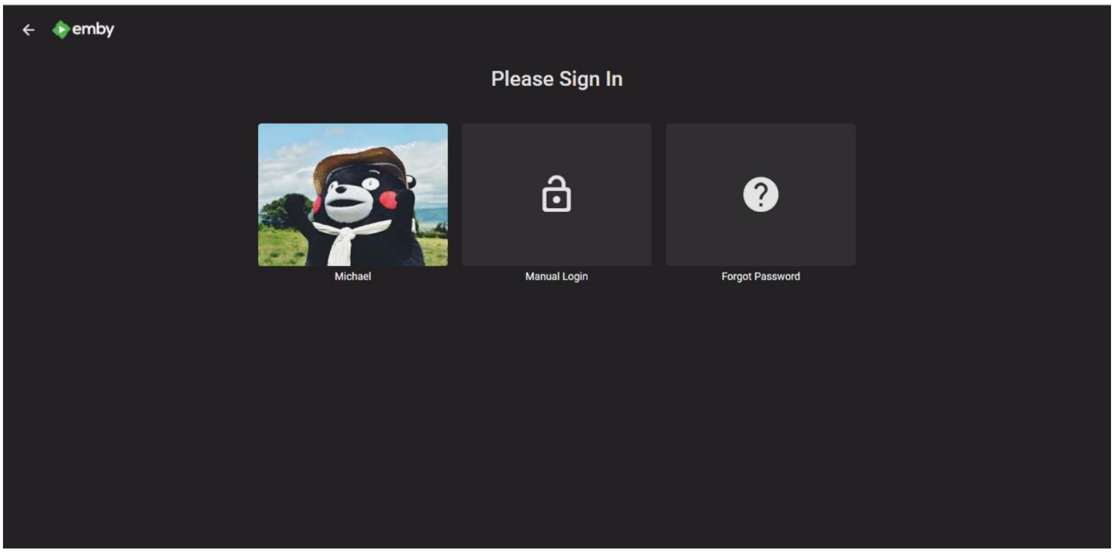
Figure 5 Top 15 Predicted records

Through the comparison of the test records and predicted records, we can find that the movie 15, 172 and 168 are the same for this specific user. Actually the performance result is far more better than the pictures shown, which will be illustrated in the exact figure of the algorithm briefing part. Some movies which are predicted successfully are on the below part of the test records. For example, movie 98 is successfully predicted, however, it is not the top 15 records of this specific user, so we cannot see the similar result. And the movie sequence is depend on the users' watching order, which may be random.

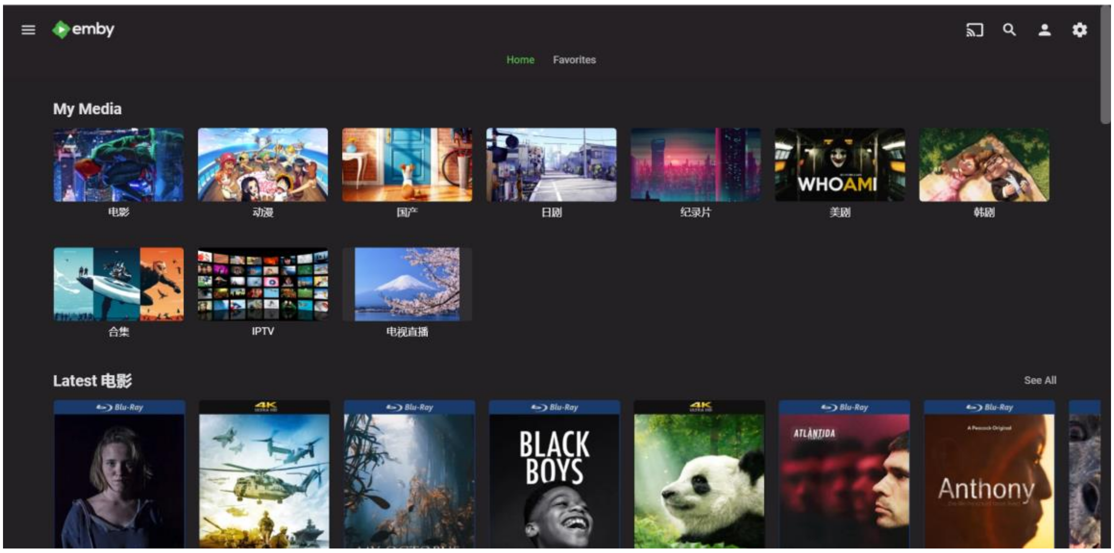
The user-friendly design still own many drawbacks, including part-shown of the records. We will do the improvement for this project. The scope is that we can show a long list of predicted result including posters and movie names, which is more helpful for the users to do the comparison and see the performance of our recommendation system themselves.

Appendix A User Guide

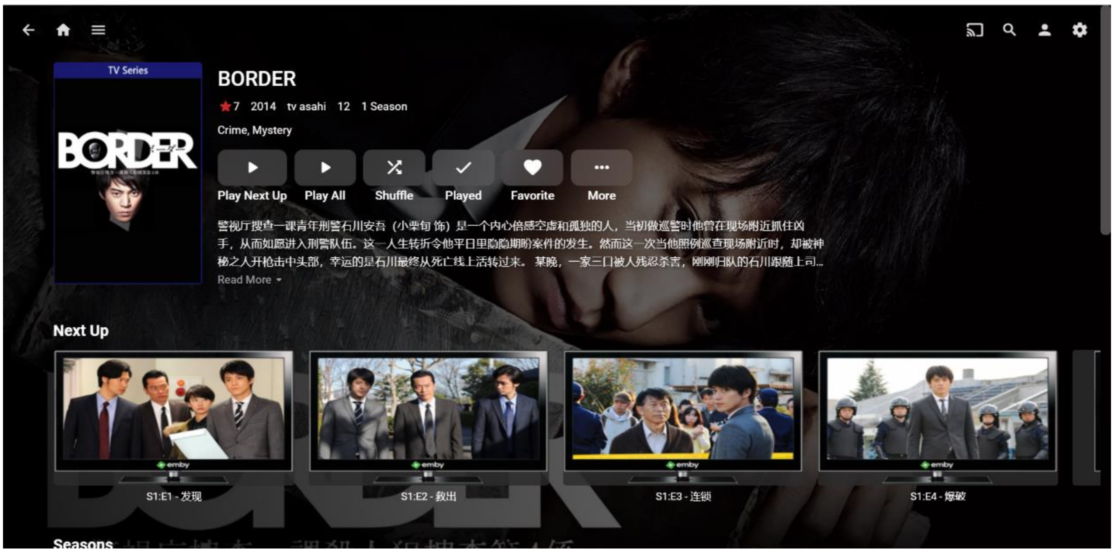
Type the URL http://junhao.io:8096/ and we can see the front interface shown like this.



Then log in with the id and password that we have registered already. Like the id ‘iss2021’ and password ‘Aa123456!’ we used in the video and then we can see the interface below.

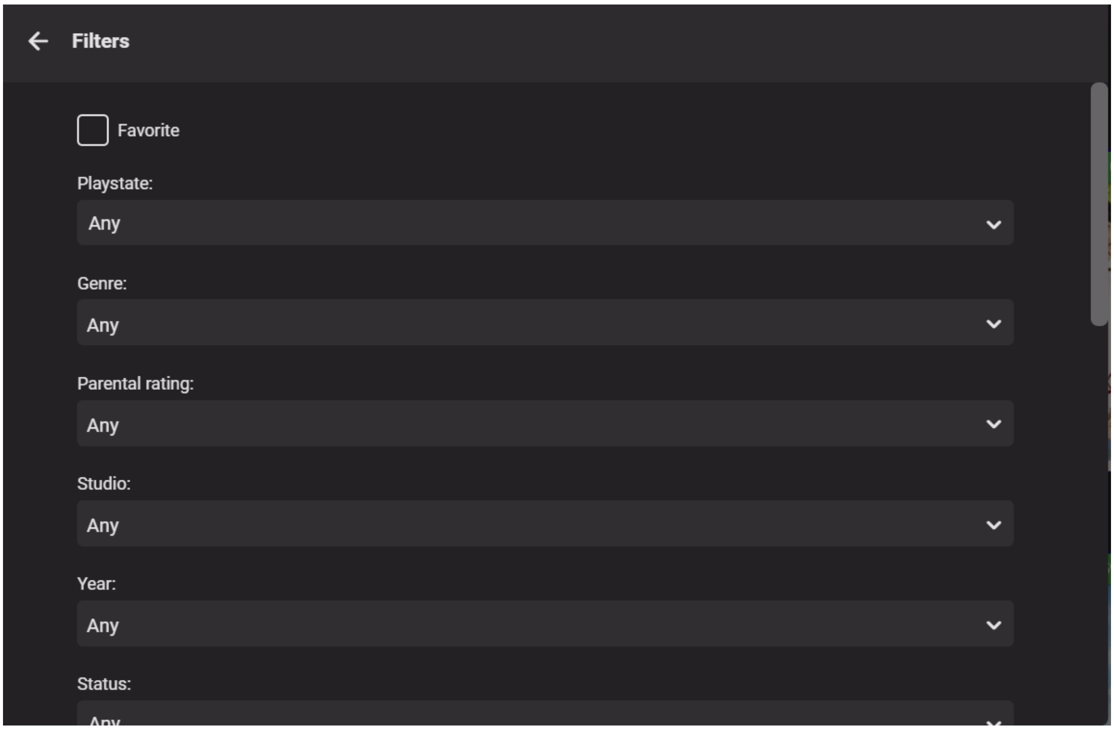


We can click in whatever movies or episodes we like and start to play.



警视厅搜查一课青年刑警石川安吾（小栗旬 饰）是一个内心倍感空虚和孤独的人，当初做巡警时他曾在现场附近抓住凶手，从而如愿进入刑警队伍。这一人生转折令他平日里隐隐期盼案件的发生。然而这一次当他照例巡查现场附近时，却被神秘之人开枪击中头部，幸运的是石川最终从死亡线上活转过来。 某晚，一家三口被人残忍杀害，刚刚归队的石川跟随上司...

We can also use the filter to choose the genres of specific movie or episodes we like.



□ Favorite

Playstate: Any

Genre: Any

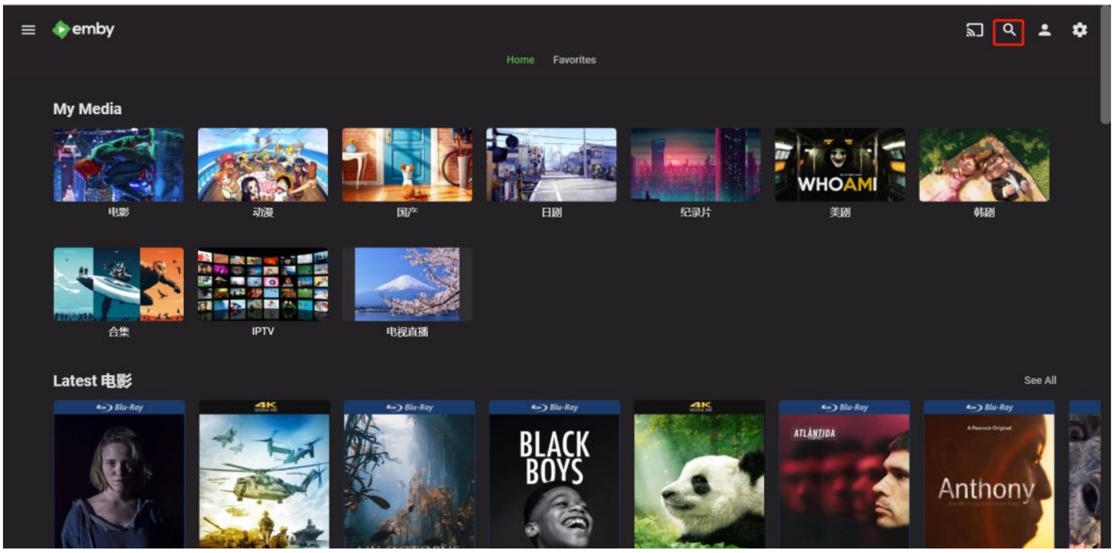
Parental rating: Any

Studio: Any

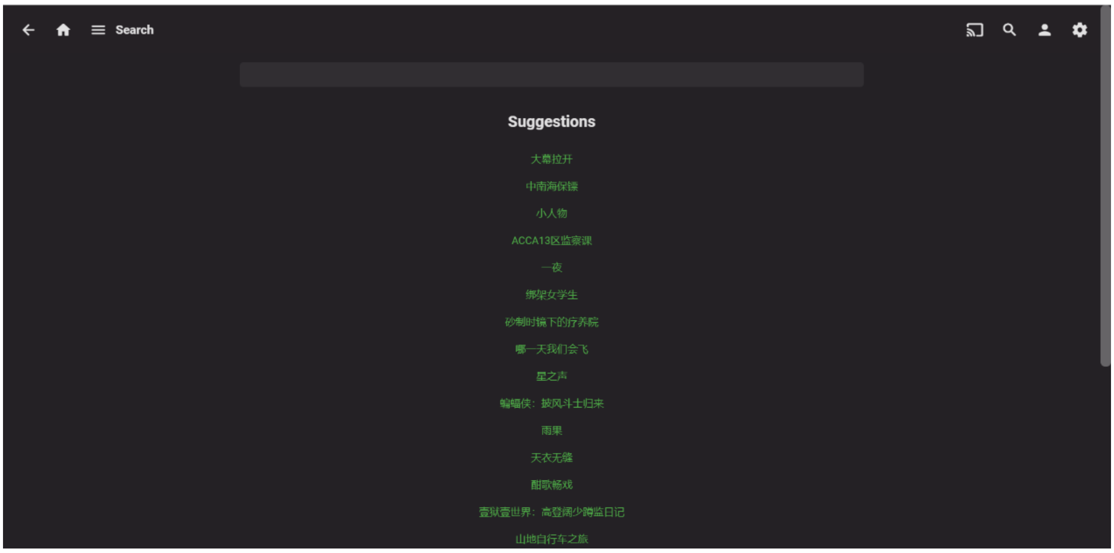
Year: Any

Status: Any

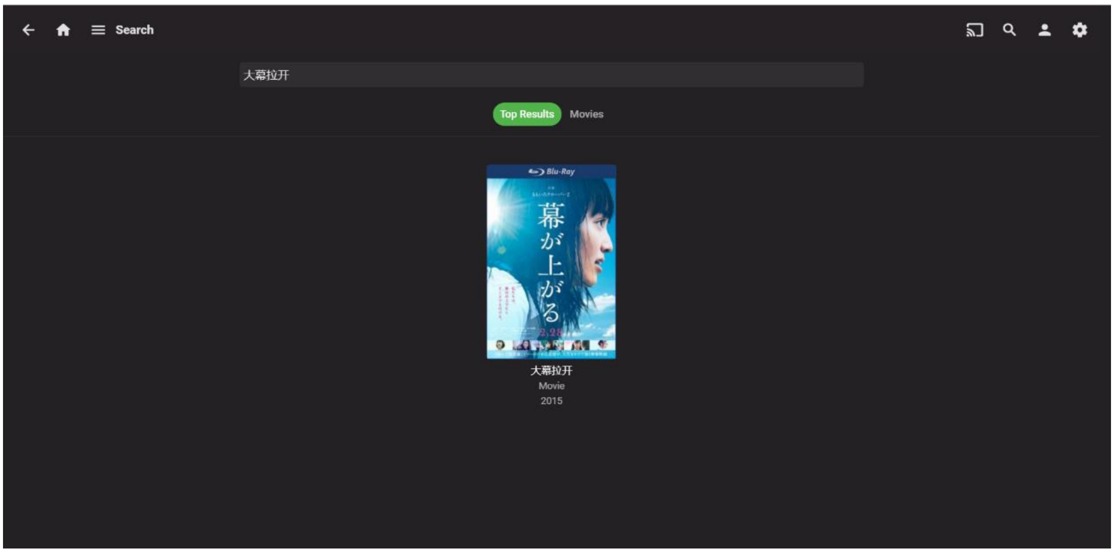
If we have clear purposed movie, we can directly click on the ‘search’ button on the top right side of the whole interface, which is marked by a red rectangle.



And we will see the interface with several suggestions, which are recommended according to several users' taste. It means these movies or episodes are quite famous now.



After searching, we can see the exact movie we want to watch.



### interaction interface ###
To use movie database and interaction interface, please refer to Appendix A User Guide;
User should log in to access all the movie database and query the database with sql language, account username and password are provided in the appendix;
You may use Report and Statistics function to access more related data such as favorite movie of each user and their watching records, or just use api key to further exploration;
For personal privacy considerations, the api key and the code of accessing user's data is currently not provided in our github.
The "data_convert.py" is used to convert the interaction records into the form that matches the input of the algorithm.

### run the algorithm ###
Click the "bat" file, the algorithm would run and record the predict results in "result/metrics.txt" and "result/predict";
"metrics.txt" reports the overall performance of the algorithm on the given dataset;
"predict" records the predicted (user, item) pairs.

### display feedback to users through the feedback interface ###
The "web_crawler.py" is used to collect and storage the movie posters from the web;
The "view.py" is used to display (i) user's historical items in train data, (ii) predicted items and (iii) items in test data,
which are collect from the train file, the generated prediction file and the test file respectively. It's supported by a light
game engine and is designed for display the feedback interface for users.

Appendix B Mapped System Functionalities

- Basically, in the process of designing the system, we actually used the "analogical reasoning" in the "Reasoning" part of the MR course. By using the goal-driven thoughts, we crawl the data which are suitable for our design and consideration.

- The techniques we used are belong to the supervised learning task due to our data with exact labels. And we built models to predict exact recommended movies or episodes that users may prefer.

- Through the whole process, we adopt the ‘knowledge graph’ method to generalize outputs. We divide the final goal into several sub-goals and searching whatever solutions exist already. If there are no exact solutions, we will try to solve the problem by splitting into sub-problem again and deal with. Or we will choose another method which can reach the effect we want to try.

- One of the most obvious characteristic that is quite fit to our RS course is the one-class collaborative filtering which we repeatedly mentioned above.

Appendix C Individual reports

Zheng Zhiwei (A0231437L)

Personal contribution

- Help to contribute in putting a state-of-the-art algorithm into real-world application
- Design the user-friendly interface to increase the user satisfaction.
- Download the movie data set in the database and crawling other data sets on the network
- Through python programming, one-to-one correspondence between users and items and movie posters, visual interface production through ‘pygame'
- Output prediction movie results and compare with the model effect, and then check the model code to a certain extent.
- Evaluated and considered the commercialization of the movie recommendation system from a theoretical and practical point of view.
- Optimized our movie recommendation system by digging into deep-seated user needs and proposed the next improvement direction.
- Report writing and organization.

Learning outcome

Through this project, I have deepened my understanding of recommendation systems and collaborative filtering algorithms. In the past, the classification of recommendations simply stayed on the basic concepts of user-based, item-based, content-based, and collaborative filtering. At the same time, I also deeply realized the importance of a user-friendly interface, which allows users to see the effect of the implementation of the entire system more intuitively, making it more valuable in business. Although this interface is not perfect, I will further study and study this aspect in the follow-up to make a more complete user-friendly interface for our recommendation system.

Knowledge and Skill Application

Recommendation systems exist in all aspects of our lives. Although this time we take a movie as an example to show the effect of our one-class collaborative filtering algorithm. But in fact, this algorithm is applicable to other aspects of recommendation, such as recommendations on e-commerce platforms, Taobao, Amazon, etc. It will have better results than traditional recommendation algorithms. In the future, I may think about how to dig into this algorithm on the basis of deep learning to make it more in line with the trend of the times, and to serve people more easily and conveniently in terms of recommendation.

Ni Yongxin (A0231559B)

Personal contribution

- Overall system design
- Control the task division
- Implement the recommendation algorithm as well as perform evaluation comparing to other baselines
- Implement the code of two units connected to the interaction interface and the feedback interface
- Adjust and optimize the system code, manage to dock with modules that constructed by other members at the integration stage
- Assisted in the implementation of web crawler and interface visualization.
- Report writing

Learning outcome

Through the practice during completing this project, I strengthened my understanding of the "system". In the past, I only focused on how to propose a better recommendation algorithm, now my understanding of "recommendation" extends to a higher system level to a certain extent. There are more factors to be considered at the system level, i.e., data collection and processing, better presentation of prediction results, as well as the impact of the designed interface on user satisfaction.

This project provides me with the opportunity to deploy my proposed algorithm to practical application, although the system may have many places to improve, this is my first attempt that meaningful to me. I have now gained the basic knowledge on what should be considered while putting an algorithm into real-world application.

I have noticed that many places in the whole system can be more automated and deploy other types of intelligent algorithms. And as far as I know, recommendation systems put into use in large companies are often supported by a variety of recommendation algorithms at the same time. I will study further to accumulate more knowledge on building a more robust system, which may involve deeper knowledge derived from the taught content and experience from practical attempt.

Knowledge and Skill Application

As emphasized earlier, our algorithm is based on implicit feedback. The advantage is that it can be extended to any recommendation scenario that can provide similar feedback. For example, we can apply to music websites, e-commerce platforms, short video applications and other fields. In addition, based on the knowledge and skills we learned, we can further use the auxiliary information provided by other platforms to make the recommendation more accurate. For example, we can use category information to enhance the recommendation and time information to make sequential recommendation.

Peng Junhao (A0231329L)

Personal contribution

- Movie collecting and movie data scraping, and matching based on artificial intelligent algorithm.
- Construction of movie database based on Google Suites and Linux Server
- The debugging and optimization of Linux movie server
- Use open-source code to mount and connect to the transfer server and Google Drive that stores movies
- Download movies in batches via python algorithm
- Use the open-source software to build a movie server and index database and applied interface visualization
- Use SQL code as well as pandas to complete the sorting of the key fields of the movie database
- Coordinate the work progress of each member and connect

Learning outcome

By completing this project, I applied python artificial intelligent algorithm to collect movies from various social media sources. Based on original file name of these materials, movie data were matched such as introductions, top cast, more like this movie and related social media links about this movie. We chose this project because we thought it would help myself better understand machine learning algorithm and improve comprehensive programming skills, also, collecting a movie database enable us to better train our recommendation systems and add some fun to our daily lives. Finding an effective way to download thousands of movies and match all the data accurately was not easy, because there are always many problems that seem trivial but must be solved, for example, a movie may have many different names, even if a movie has only one name, there will be different naming methods, therefore, how to avoid duplication of data as much as possible, and at the same time match other information according to the movie file name as accurately as possible, became a huge challenge for me, I consulted netizens and friends, read, and watched various tutorials, and pored over several approaches to find the one that works best. Then I discussed the ideas with my teammates every week and optimized each step in the database constructing process. I carried a notebook wherever we started discussing how to build and optimize our system, took down every good idea that came to me, tested each of them, and then summarized and reflected on them. Through this experience, I became more confident in my ability to navigate challenges, At the same time, I have become more proficient in machine learning algorithms, linux shell scripts, python data analysis techniques, front-end construction and other programming skills, and better integrate these scattered and isolated knowledge to solve the real problems we face. More importantly, I learned that perseverance and hard work are the essential ingredients for success.

Knowledge and Skill Application

I believe that the artificial intelligence algorithms, server deployment, database construction and python data mining techniques used in this project are common in future studies and work, so the knowledge points learned and used in the process of completing this project Skills can greatly help me complete many more complex projects in the future, such as data mining systems based on machine learning, information acquisition and feedback systems, and user behavior prediction systems. Thanks to this project, I integrated a lot of fragmentary knowledge learned in class and use them to solve problems in real life. Problems that seemed to be isolated from each other in the past are integrated through the intersection of knowledge points. I think this this comprehensive ability is indispensable in the process of solving complex problems.