Implement a search engine with domain-combination search function

Yang Bao, 2020/4/07

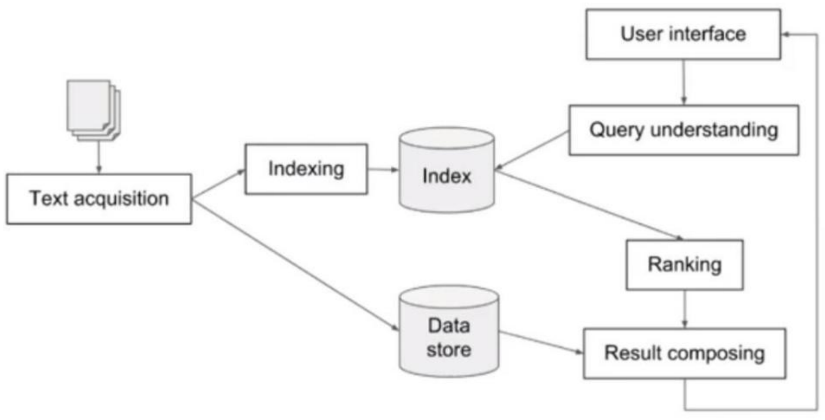
Github link:
https://github.com/baoyangisapig/Final-Project-for-information-retrieval

Abstraction

MetaCritic is a movie review site(https://www.metacritic.com/). For each movie, the site would provide the averaged scores by film critics and movie viewers. Also, we can review all the details about a movie including the comments and the summary. As to the search engine of the site, the functionality is basic. This is to say, we can only search for the name of a film. I want to add more functionality to the site to improve the search engine. The improvement is called a combination search.

The information retrieved from the website has several domains: the name of the film, director, movie poster image, comments, a summary of the movie plot, the user score and the professional score. In my opinion, These three domains have more useful information and can be used to locate a movie: name, comments and the summary of a specific movie. So I would implement two kinds of search models: the first kind is searching in one of the three domains and find the most related films, and the second one is to do a search in multiple domains. O f course, the second model is based on the first one. In the following article, I will show details on how to implement the functionality in a search engine.

Here is the basic structure of the information retrieval system:



Preparation before designing the system

1. Write a web crawler and clean origin data.

Before we implement the search system, we need to implement a web crawler to get the data we need. I use Selenium to implement the crawler.

Selenium is an open-source tool that automates web browsers. It provides a single interface that lets you write test scripts in programming languages like Ruby, Java, NodeJS, PHP, Perl, Python, and C#, among others.

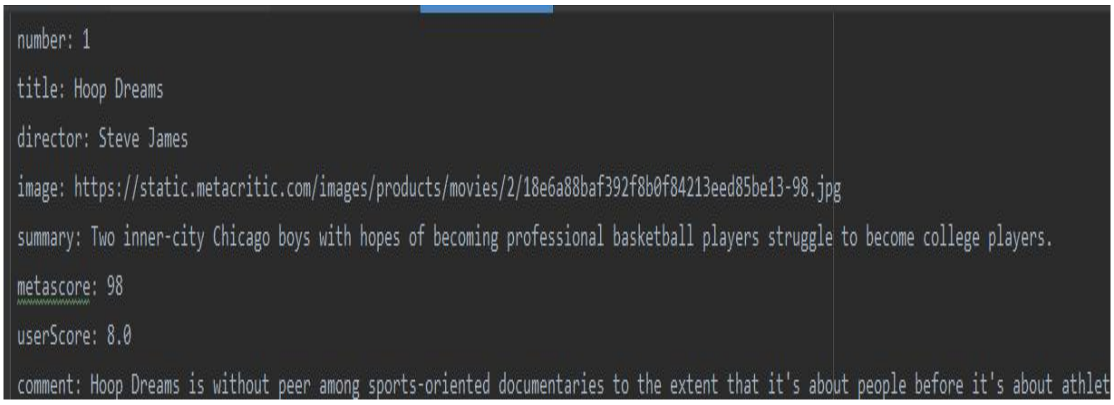
A browser-driver then executes these scripts on a browser-instance on your device (more on this in a moment).

This is how web crawler works. Firstly, we write a spider script and execute it with web driver. Then we can get the crude data from web. After cleaning the data, we can get organized data. The final step is to write it down in files and store it.

2. Parse XML file.

I store these data in XML files. So I need to read and parse the files to get organized data.

This is the structure of the XML files.I read the data line by line and get the value of each domain with BufferedWriter.



```
number: 1
title: Hoop Dreams
director: Steve James
image: https://static.metacritic.com/images/products/movies/2/18e6a88baf392f8b0f84213eed85be13-98.jpg
summary: Two inner-city Chicago boys with hopes of becoming professional basketball players struggle to become college players.
metascore: 98
userScore: 8.0
comment: Hoop Dreams is without peer among sports-oriented documentaries to the extent that it's about people before it's about athlet
```

Explanation of each domain in the XML file:

①Number: the index of a specific film.

②Title: the name of the film.

③Director: director of the movie.

④Image: the URL of the movie poster image.

⑤Summary: the summary of the movie plot.

⑥metaScore:the professional score of the movie

⑦userScore: user socre of a specific movie

⑧comment: movie comments

Steps and explanation on how to implement the information retrieval algorithm

1. construct the inverted index.

Steps to build an inverted index:

①Fetch the Document
Removing of Stop Words: Stop words are most occuring and useless words in document like “I”, “the”, “we”, “is”, “an”.
②Stemming of Root Word
Whenever I want to search for “cat”, I want to see a document that has information about it. But the word present in the document is called “cats” or “catty” instead of “cat”. To relate the both words, I’ll chop some part of each and every word I read so that I could get the “root word”. There are standard tools for performing this like “Porter’s Stemmer”.
③Record Document IDs
If word is already present add reference of document to index else create new entry. Add additional information like frequency of word, location of word etc.
④Repeat for all documents and sort the words.

The algorithm is based on multiple domains, so we need to construct an inverted index for each domain. So, after following the steps above, we would have three inverted indexes: name, comment and summary.

2. implement BM25 algorithm to get the basic score of each document.

The ranking function is based on BM25 algorithm but I will adjust the weight of each variable and add some new variables.

This part requests us to add a ranking function to the existed system and evaluate the performance of the whole search engine.

First,we have already built the inverted indexs . Based on the given data. The next thing we should focus on is that how to SCORE a document according to the input query. The higher score one document gets, the higher possible it is required by the user.

Second, we need to deal with the SCORE. Based on the textbook and notes on the class, I choose BM25 algorithm as a "SCORE" function to evaluate the degree of a document fits for the query. The reason I choose BM25 is easy to implement and it performs well on many existed models. Besides, there are a few parameters for programmers to adjust due to a specific document.

Third, we need to implement the BM25 algorithm. BM25 comes from the idea of TF-IDF. It solves the problem in the traditional TF-IDF that TERM frequency will influence the performance when it grows big enough.

```
TF-IDF TF Score = sqrt(tf)
BM2M TF Score = ((k + 1) * tf) / (k + tf)
```

with the help of the parameter k, it helps us to limit the influence from the size of tf. Now, Let's dive deep into it. Below is the method to calculate the BM25 score(here we just omit a large sum of mathematical justification).

$$R(q_i, d) = \frac{f_i \cdot (k_1 + 1)}{f_i + K} \cdot \frac{qf_i \cdot (k_2 + 1)}{qf_i + k_2}$$

$$K = k_1 \cdot (1 - b + b \cdot \frac{dl}{avgdl})$$

R is the result we want, qi means each word in the Query, d means document, fi means the frequency of qi occurs in the document d. K is a constant number and we need to calculate the length of the current document dl and the average document length of the whole database avgdl. K means that the longer a document is, the smaller the document's score will be. And we use

$$Score(Q, d) = \sum_{i}^{n} IDF(q_i) \cdot \frac{f_i \cdot (k_1 + 1)}{f_i + k_1 \cdot (1 - b + b \cdot \frac{dl}{avgdl})}$$

to calculate the final results for each document.

Finally, we have already known the power SCORE function, bm25, and then we could (1)score every documents (2)use a priorityqueue to get the top-k documents as the result of best fit document (3)return document ids to the user. Another thing is to separate the input query into a bag of words. This procedure can be ascribed as the following steps: (1)change them into a word array (2)find all stop words (3)there may be phrase in the query, so pick them out and deal with them separately (4)some words may not exist in the documents so find a proper way to deal with them (5)rearrange the score list and return the result to the user.

3. implement single-domain search

This part is very easy. We have implemented the BM25 algorithm to calculate the score of each film. Then we use a min Heap to sort these film by their score. So we can provide a Restful API to select the top K films given s specific movie id.

So we can get the top K movies in a single domain. For example, the query is “teenager love”, we can search the keywords in comments and get the top K results based on the algorithm above.

4 implement combination-domain search

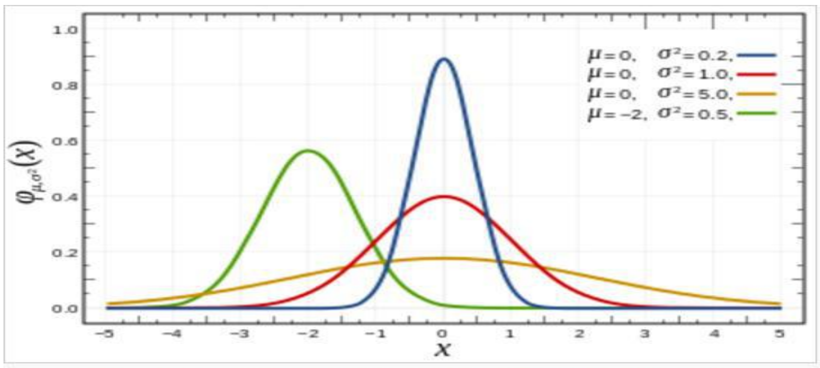
This is the core part of the information retrieval system. I would use three domains to implement the search: name, comment, and summary. When we use the single-domain search, we can know the score of each film. The higher the score, the more related the movie is to the query. So based on the three inverted indexes, we can get three kinds of a score, the scoring base on the film name S_1, the score based on film summaries S_2 and the score based on film comments S_3.

As we know, it is a combination search, the user can search for any kind of information related to the target film, so we need to analyze the intention of users. My assumption is that the closer the length of query keywords is to the average length of movie names, the more likely the user is to query the film by its name. So when the length of query keywords is close to the length of the averaged film name, the weight of S_1 increases in the final score in this query.

Suppose that the average length of film names in the data set is l_1, then the weight increases when the length of the query approaches l_1. I use the Gaussian function to simulate the process.

This is the gaussian function. σ is the standard deviation and μ is the mathematical expectation.

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}.$$



Normalized Gaussian curves with expected value μ and variance σ^2. The corresponding parameters are $a = \frac{1}{\sigma\sqrt{2\pi}}$, $b = \mu$, and $c = \sigma$.

From the graph above, we can know that when x= μ ,the value of function reaches the highest point. Also, the highest point is smaller than 1. I use the value of gaussian function to simulate the weight of S_1 in the final score.

We let σ =1, μ =l_1, so when the length of query is equal to l_1, the weight of S_1 is the highest, which is the value of g(x) . So we can get the calculation formula of final Score.

Based on BM25, the score of a film in single domain is:

$$Score\,(Q,d) = \sum_{i}^{n} IDF\,(q_i) \cdot \frac{f_i \cdot (k_1+1)}{f_i + k_1 \cdot (1 - b + b \cdot \frac{dl}{avgdl})}$$

Define the final score as S_f

$S_f=k_1*S_1+K_2*S_2+K_3*S_3$

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}.$$

k_1=g(x), let σ =1, μ =l_1. $K_2=(1-k_1)*\varepsilon_1$,$K_3=(1-k_1)*\varepsilon_2$,let $\varepsilon_1=0.4$,$\varepsilon_2=0.6$.

Based on that, we can get the calculation formula of final score

$S_f=g(x)*S_1+(1-g(x))*\ \varepsilon_1*S_2+(1-g(x))*\ \varepsilon_2*S_3.$

Now,we still need to do some improvement on S_f, because the value of S_1,,S_2 and S_3 are not in the same range. So we need to do Normalization on the original Score.

$$S_{i.j} = S_{i.j} / \sum_{k=1}^{N} S_{i,k}$$

So the value of $S_{i,j}$ is also in [0,1]. Also, we multiply the result by N to offset the effect of normalization on the result value, in which N is the number of films in the data set.

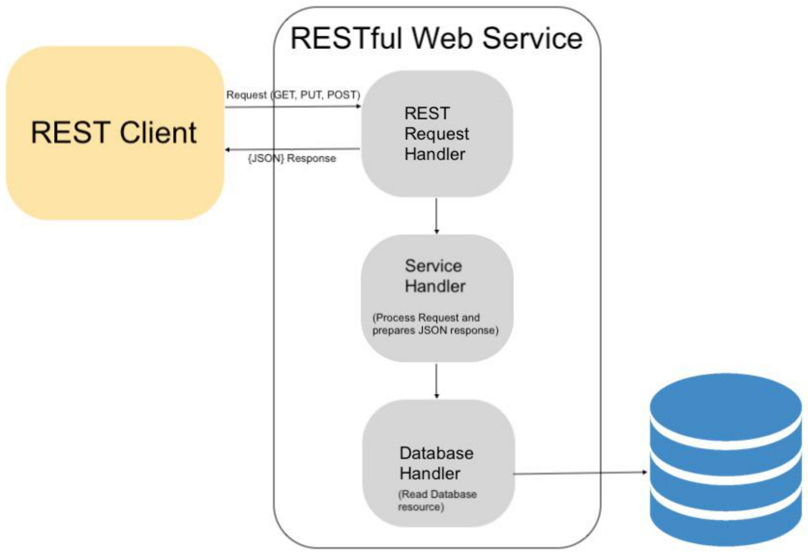
$$S_f=[\, g(x)*S_1/\sum_{k=1}^{N} S_{1,k} +(1-g(x))*\ \varepsilon_1*S_2/\sum_{k=1}^{N} S_{2,k} +(1-g(x))*\ \varepsilon_2*S_3/\sum_{k=1}^{N} S_{3,k}\ ]*N.$$

Base on the calculation formula, we can calculate the final score of each film. With a min Heap, we can sort all these films and get the top K films based on the user's query. This is the result of the combination-domain search.

Introduction on the System Design

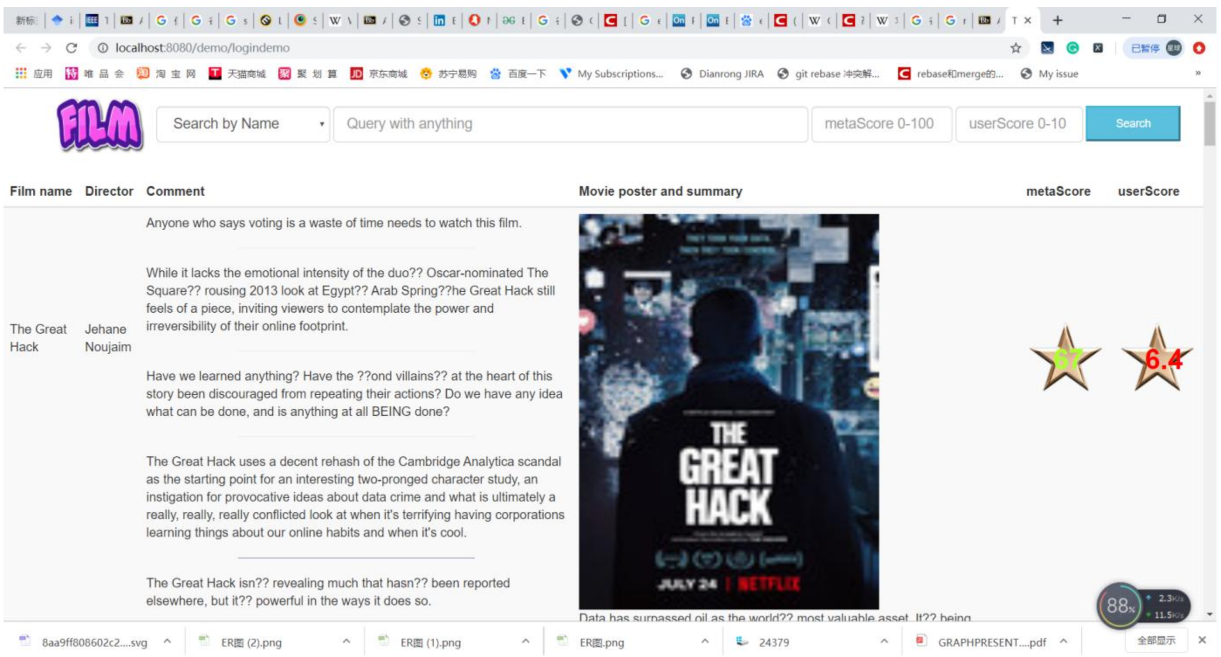
The search engine is based on a web user interface. I would implement the front-end with Bootstrap and implement the back-end part with SpringBoot.

When we search on the website, the request will be redirected to a Restful API by a corresponding controller in the MVC structure. Based on the information retrieval algorithm and ranking function, we can get the results of a query. We can convert the result to JSON formation and present it in the front-end.



Instruction on how to use the web interface.

UI of the web interface:



Film name | Director | Comment | Movie poster and summary | metaScore | userScore

The Great Hack | Jehane Noujaim

Anyone who says voting is a waste of time needs to watch this film.

While it lacks the emotional intensity of the duo?? Oscar-nominated The Square?? rousing 2013 look at Egypt?? Arab Spring??he Great Hack still feels of a piece, inviting viewers to contemplate the power and irreversibility of their online footprint.

Have we learned anything? Have the ??ond villains?? at the heart of this story been discouraged from repeating their actions? Do we have any idea what can be done, and is anything at all BEING done?

The Great Hack uses a decent rehash of the Cambridge Analytica scandal as the starting point for an interesting two-pronged character study, an instigation for provocative ideas about data crime and what is ultimately a really, really, really conflicted look at when it's terrifying having corporations learning things about our online habits and when it's cool.

The Great Hack isn?? revealing much that hasn?? been reported elsewhere, but it?? powerful in the ways it does so.

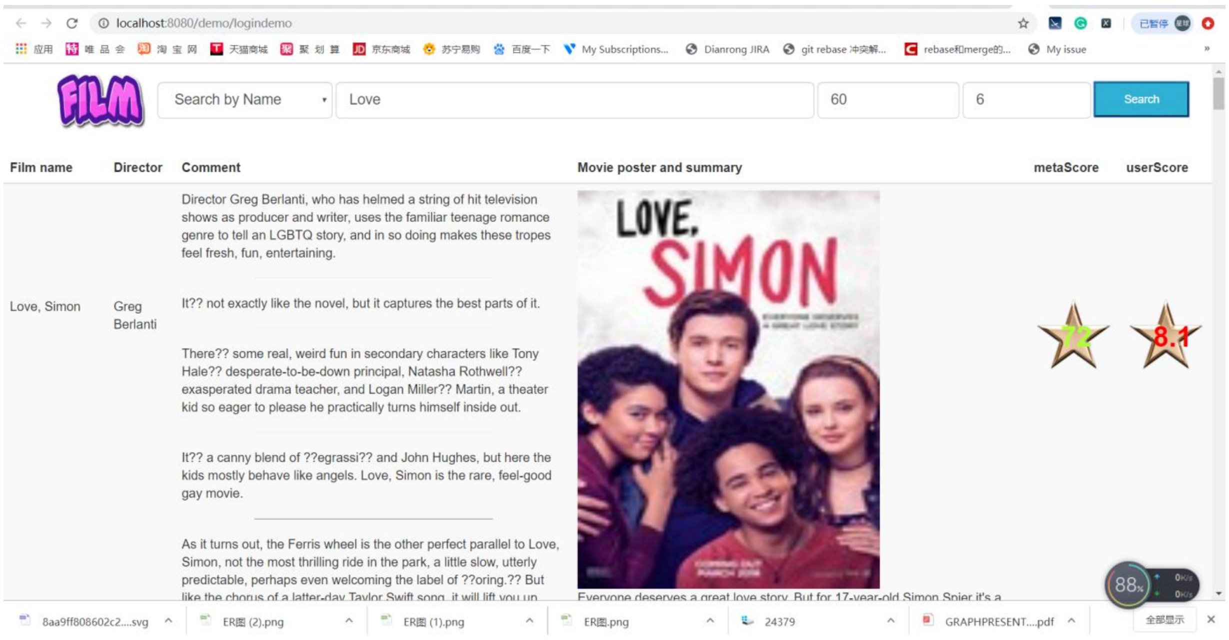
Data has surpassed oil as the world?? most valuable asset. It?? being

67 | 6.4

You can choose search in one domain or in multiple domains. Also, you can set the limit value of userScore and metaScore.

There are four function of the web interface:

① Search by Name:



Film name | Director | Comment | Movie poster and summary | metaScore | userScore

Love, Simon | Greg Berlanti

Director Greg Berlanti, who has helmed a string of hit television shows as producer and writer, uses the familiar teenage romance genre to tell an LGBTQ story, and in so doing makes these tropes feel fresh, fun, entertaining.

It?? not exactly like the novel, but it captures the best parts of it.

There?? some real, weird fun in secondary characters like Tony Hale?? desperate-to-be-down principal, Natasha Rothwell?? exasperated drama teacher, and Logan Miller?? Martin, a theater kid so eager to please he practically turns himself inside out.

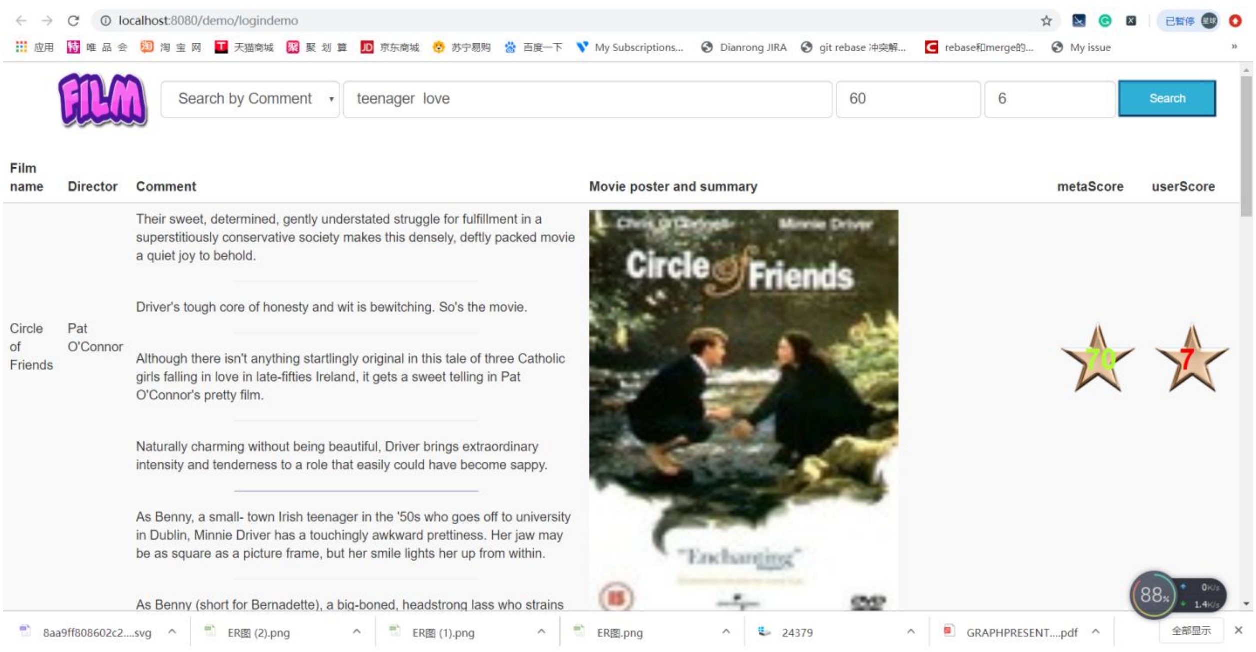
It?? a canny blend of ??egrassi?? and John Hughes, but here the kids mostly behave like angels. Love, Simon is the rare, feel-good gay movie.

As it turns out, the Ferris wheel is the other perfect parallel to Love, Simon, not the most thrilling ride in the park, a little slow, utterly predictable, perhaps even welcoming the label of ??oring.?? But like the chorus of a latter-day Taylor Swift song, it will lift you up

Everyone deserves a great love story. But for 17-year-old Simon Spier it's a

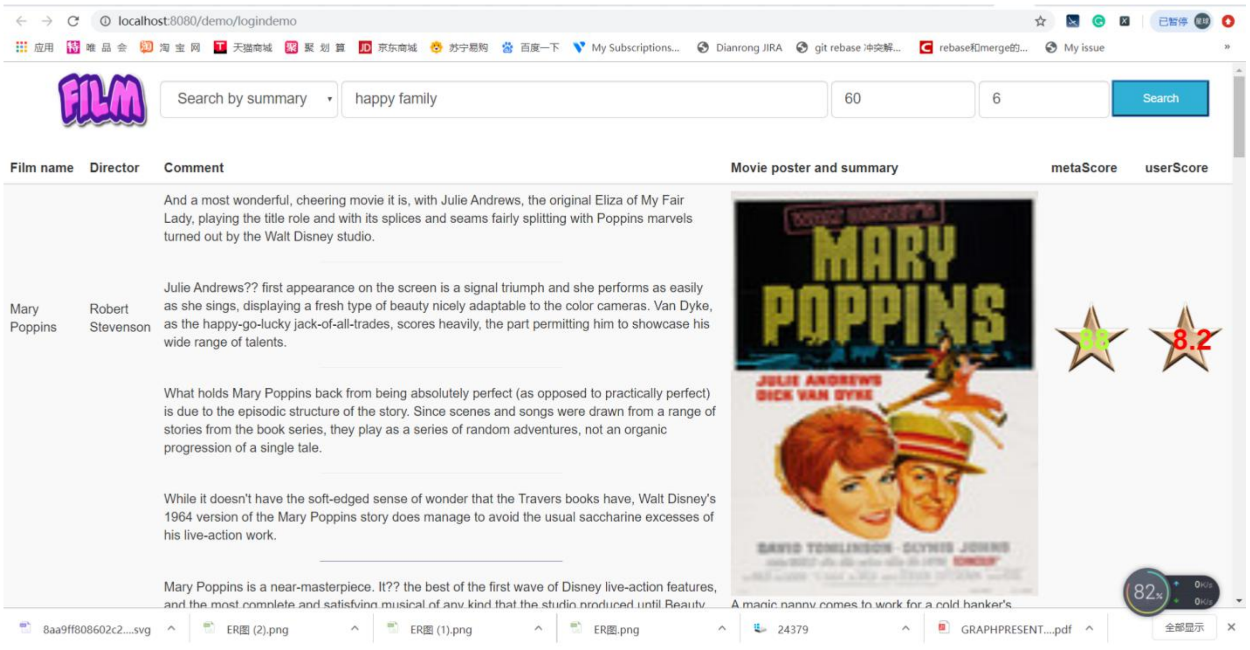
72 | 8.1

② Search By Comment:


localhost:8080/demo/logindemo
应用 唯品会 淘宝网 天猫商城 聚划算 京东商城 苏宁易购 百度一下 My Subscriptions... Dianrong JIRA git rebase 冲突解... rebase和merge的... My issue
FILM
Search by Comment
teenager love
60
6
Search
8aa9ff808602c2....svg ER图 (2).png ER图 (1).png ER图.png 24379 GRAPHPRESENT....pdf 全部显示

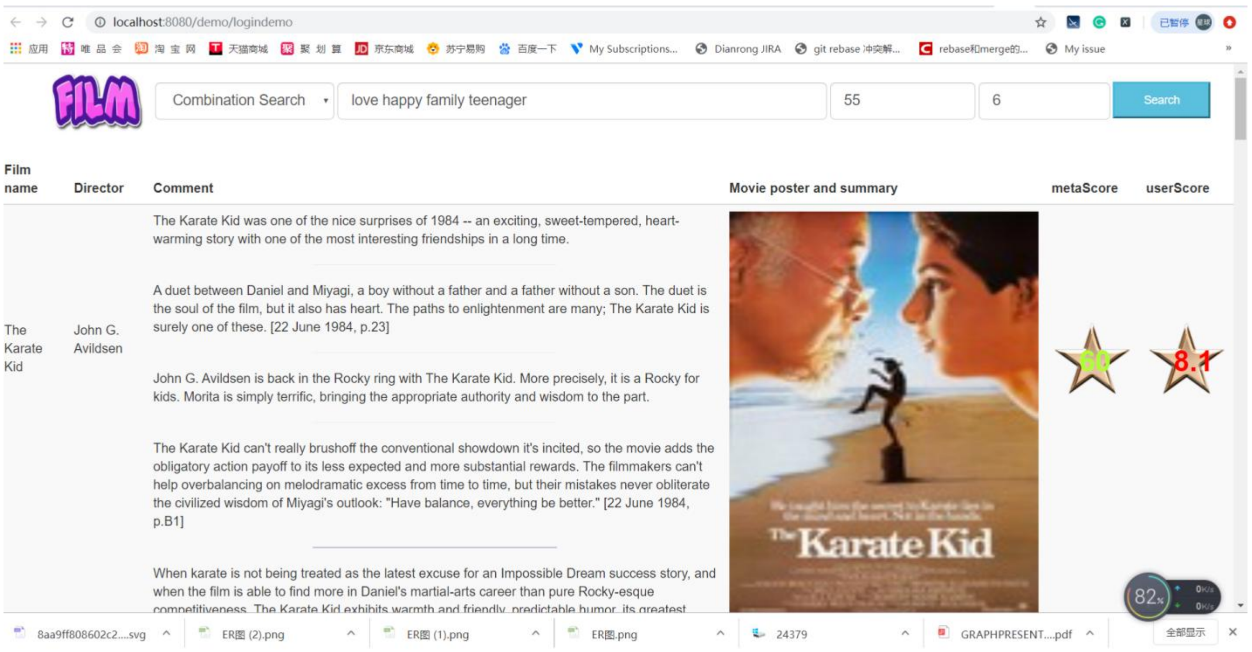
Film name	Director	Comment	Movie poster and summary	metaScore	userScore
Circle of Friends	Pat O'Connor	Their sweet, determined, gently understated struggle for fulfillment in a superstitiously conservative society makes this densely, deftly packed movie a quiet joy to behold. Driver's tough core of honesty and wit is bewitching. So's the movie. Although there isn't anything startlingly original in this tale of three Catholic girls falling in love in late-fifties Ireland, it gets a sweet telling in Pat O'Connor's pretty film. Naturally charming without being beautiful, Driver brings extraordinary intensity and tenderness to a role that easily could have become sappy. As Benny, a small- town Irish teenager in the '50s who goes off to university in Dublin, Minnie Driver has a touchingly awkward prettiness. Her jaw may be as square as a picture frame, but her smile lights her up from within. As Benny (short for Bernadette), a big-boned, headstrong lass who strains	Chris O'Donnell Minnie Driver Circle of Friends "Enchanting"	70	7

③Search By Summary


localhost:8080/demo/logindemo
应用 唯品会 淘宝网 天猫商城 聚划算 京东商城 苏宁易购 百度一下 My Subscriptions... Dianrong JIRA git rebase 冲突解... rebase和merge的... My issue
FILM
Search by summary
happy family
60
6
Search
8aa9ff808602c2....svg ER图 (2).png ER图 (1).png ER图.png 24379 GRAPHPRESENT....pdf 全部显示

Film name	Director	Comment	Movie poster and summary	metaScore	userScore
Mary Poppins	Robert Stevenson	And a most wonderful, cheering movie it is, with Julie Andrews, the original Eliza of My Fair Lady, playing the title role and with its splices and seams fairly splitting with Poppins marvels turned out by the Walt Disney studio. Julie Andrews?? first appearance on the screen is a signal triumph and she performs as easily as she sings, displaying a fresh type of beauty nicely adaptable to the color cameras. Van Dyke, as the happy-go-lucky jack-of-all-trades, scores heavily, the part permitting him to showcase his wide range of talents. What holds Mary Poppins back from being absolutely perfect (as opposed to practically perfect) is due to the episodic structure of the story. Since scenes and songs were drawn from a range of stories from the book series, they play as a series of random adventures, not an organic progression of a single tale. While it doesn't have the soft-edged sense of wonder that the Travers books have, Walt Disney's 1964 version of the Mary Poppins story does manage to avoid the usual saccharine excesses of his live-action work. Mary Poppins is a near-masterpiece. It?? the best of the first wave of Disney live-action features, and the most complete and satisfying musical of any kind that the studio produced until Beauty	Mary Poppins Julie Andrews Dick Van Dyke David Tomlinson · Glynis Johns A magic nanny comes to work for a cold banker's	88	8.2

④ Combination Search



Film name	Director	Comment	Movie poster and summary	metaScore	userScore
The Karate Kid	John G. Avildsen	The Karate Kid was one of the nice surprises of 1984 -- an exciting, sweet-tempered, heart-warming story with one of the most interesting friendships in a long time. A duet between Daniel and Miyagi, a boy without a father and a father without a son. The duet is the soul of the film, but it also has heart. The paths to enlightenment are many; The Karate Kid is surely one of these. [22 June 1984, p.23] John G. Avildsen is back in the Rocky ring with The Karate Kid. More precisely, it is a Rocky for kids. Morita is simply terrific, bringing the appropriate authority and wisdom to the part. The Karate Kid can't really brushoff the conventional showdown it's incited, so the movie adds the obligatory action payoff to its less expected and more substantial rewards. The filmmakers can't help overbalancing on melodramatic excess from time to time, but their mistakes never obliterate the civilized wisdom of Miyagi's outlook: "Have balance, everything be better." [22 June 1984, p.B1] When karate is not being treated as the latest excuse for an Impossible Dream success story, and when the film is able to find more in Daniel's martial-arts career than pure Rocky-esque competitiveness. The Karate Kid exhibits warmth and friendly, predictable humor, its greatest		60	8.1

The users can use the four kinds of search function to find their target film in the search engine. They will get all the related film after clicking the search button. There are several components of a film’s description: film name and poster,director,comment, summary and film scores given by professionals and general users.

These are the target films that the system think the user may interested in based on the query.