CSE515 Multimedia and Web Databases Phase 1 Group 15

XIANGYU GUO, Arizona State University
SIDDHANT TANPURE, Arizona State University
CHENCHU GOWTHAM YERRAPOTHU, Arizona State University
ALFRED GONSALYES, Arizona State University
GREGGORY SCHERER, Arizona State University

Abstract

In this report, we are introducing a method to map actors/users/genres retrieved from IMDB database to a vector space with tags. When perform the mapping we used two different models, term frequency (TF) and term frequency-inverse document frequency (TF-IDF). After the mapping, we try to compare two different genres using three models, term frequency-inverse document frequency-difference (TF-IDF-DIFF). We discover that by using properly selected weight function or formula, the user can easily discover a tag with most discriminative power for one specific genre. Furthermore, the user can understand the difference between two genres in a quantitative way, like based on the p-norm result, the dot product, and the cosine value/angle of two vectors. At last, we created a program encoding those solutions, also provide a command line interface(CLI) for the user to run their query.

Keywords: Vector Model, Term Frequency (TF), Term Frequency–Inverse Document Frequency (TF-IDF), TF-IDF-DIFF, P-DIFF1, P-DIFF2.

Introduction

A. Terminology

Here are the terminologies may appear in the report.
Vector space
a representation of all objects described by N dimension features.
Features
a set of dimensions describes the vector space.
Objects
a set of objects appears in/represents by the vector space.
Term Frequency(TF)
TF<f> = k/N. Where N means the total number of frequency of all the features, and k means the frequency of the certain feature f.
Inverse Document Frequency(IDF)
IDF<f> = log(N/m). Where N means the total number of objects, and m means the number of objects have feature f.
Term Frequency-Inverse Document Frequency(TF-IDF)
TF-IDF<f> = IDF<f> * TF<f>.
Term Frequency-Inverse Document Frequency-Difference(TF-IDF-DIFF), TF-IDF-DIFF<o1, o2> = {v_i}, for all i, v_i = $o1_i$ – $o2_i$.
Probability Difference(P-DIFF)
P-DIFF<o1, o2> = {wi}, for all i,
wi = $\log \frac{r1,j/(R-r1,j)}{(m1,i-r1,i)/\,(M-m1,j-R+r1,j)} * \left|\frac{r1,j}{R} - \frac{m1,j-r1,j}{M-R}\right|$

B. Goal description

After this project, we will learn how to map different types of objects(entities) to one same features vector space or map same objects(entities) to different features vector space, and how to explains difference between entities based on vector representation.
Particularly, this phase focus on a movie database, so we are mapping actors/genres/users to tags vector space using model TF or TF-IDF. Then we are comparing two genres represent by tags vector using model TF-IDF-DIFF, P-DIFF1, and P-DIFF2.
After this phase, we will provide a tool/program for the user to run certain query described by the project specification. At last, the query result should be obtained within a reasonable time.

C. Assumptions

Few assumptions are made during this project implementation.

1. Invalid input, will output nothing.

This including invalid command, object id, and model. For example, the user_id 1, he/she doesn't contribute a tag to any movie, so the query on this user will give empty output. The actor_id 13993, he/she only played on movie 3962, but this movie didn't have a valid tag in mltags, so the query on this actor will give empty output. For genre Film-Noir, no movies related to this genre been tagged, so the query on this genre will give empty output.

2. Weight function on actor movie rank.

We use weight(rank) = 1 / rank to calculate the weight related to actor’s rank in one movie. This function keep the weight simple and gives higher weight on lower rank. However, maybe an actor ranking at 3 in a movie with 3 actors is different with an actor ranking at 3 in a movie with 10 actors. We need to more data, like how long this actor shows in the movie to give a fairer weight function.

3. Weight function on timestamp.

We use weight(ts) = (ts - MIN(TS)) / (MAX(TS) - MIN(TS)) to calculate the weight related to tag's timestamp for one tag record. TS stands for the timestamps vector from mltags table. "ts" stands for the timestamp of current record. To fix the edge case, we modified our weight function into weight(ts) = (ts - MIN(TS) + 1) / (MAX(TS) - MIN(TS) + 1). So, we can avoid the weight to be zero.

4. Weight function on combining two weights

5. In task1, we use the total actors from the movies been tagged, not from all actors.
The actor played in a movie without any tag, makes no contribution to his/her tag vector.

6. In task2, we use the total genres from the movies been tagged, not from all movies. (18 not 19)
The genre no user tagged, with a tag vector all 0. It doesn't make any contribution.

7. In task3, we define the movie "watched" by the user from mltags and mlratings.
User watched a movie then gave the tag and/or rating to that movie.

8. In task3, we use the total user from the users gave the tag/rating to a movie, not from all users. (75K vs 30K)
The user didn't have any tags, so the tag vector for this user will be all 0. It doesn't make any contribution.

9. In task4, we use movies from mltags to compute Movies(g1) and Movies(g2), not the movies from all movies.
The movie no user tagged, didn't contribute to related genre.

10. For all tasks, we use 10 as the base of all log functions.
The base to the log function only matters on the scaling part, so if we use the same base for all log operations, it will be the same.

Description of the proposed solution/implementation

During the implementation, I load all data into the memory. Then calculate the weighted tag vector for actors/genres/users, as well as the idf value of each documents. Based on this precalculated result, I can compute the tf and tf-idf on the fly.

For the tf-idf-diff, I use the p-norm, dot-product, cosine, angle to calculate the difference between two vectors.

For the P-DIFF1/2, we need to fix the edge cases, based on the 12.4.4 from the textbook [Kasim Selcuk Candan and Maria Luisa Sapino. 2010.], we choose fixed contribution 0.5 on all edge cases. And plug it into the formula, we will get,

$$wi = \log \frac{(r1,j+0.5)/(R-r1,j+0.5)}{(m1,i-r1,i+0.5)/\ (M-m1,j-R+r1,j+0.5)} * \left| \frac{r1,j+0.5}{R+1} - \frac{m1,j-r1,j+0.5}{M-R+1} \right|$$

Then we use this formula to calculate the wi on edge cases.

Interface specifications

Our program created a command line interface (CLI) to interact with users. It supports four types queries, “print_actor_vector”, “print_genre_vector”, “print_user_vector”, and “differentiate_genre”. Each command’s parameters separated by space.

1. print_actor_vector actor_id model
   actor_id, a valid actor id, without quote, e.g. 1582699
   model, a valid model name in small letters without the quote, either tf or tf-idf
   example query
   print_actor_vector 1582699 tf-idf
2. print_genre_vector genre model
   genre, a valid genre name, without quote, e.g. Thriller
   model, a valid model name in small letters without the quote, either tf or tf-idf
   example query
   print_ genre _vector Thriller tf-idf
3. print_user_vector user_id model
   user_id, a valid user id, without quote, e.g. 146
   model, a valid model name in small letters without the quote, either tf or tf-idf
   example query
   print_ user _vector 146 tf-idf
4. differentiate_genre genre1 genre2 model
   genre1, a valid genre name, without quote, e.g. Thriller
   genre2, a valid genre name, without quote, e.g. Romance
   model, a valid model name in small letters without the quote, it can be tf-idf-diff, p-diff1 or p-diff2.
   example query
   differentiate_genre Thriller Romance tf-idf-diff

System requirements/installation and execution instructions

0. Running environment, any machine/OS with Python3.0 installed.
1. Put all csv data file under the “data” folder.
2. Go to the “Code” folder using your command line tool. (Shell or CMD)
3. Execute the program with “python project1.py”.
4. Wait for around 20 seconds, you will see the following information during this progress.
“loading csv data into memroy...
loading completed!
preprocessing data...
preprocessing completed!”
5. After you see a “query>”, you can running your query now.
6. How to run each query command, please refer the section “Interface Specifications” for more detail.

Related work

N/A

Conclusions

We discover that by using properly selected weight function or formula, the user can easily discover a tag with most discriminative power for one specific genre. Furthermore, the user can understand the difference between two genres in a quantitative way, like based on the p-norm result, the dot product, and the cosine value/angle of two vectors.

Bibliography

[1] Kasim Selcuk Candan and Maria Luisa Sapino. 2010. Data Management for Multimedia Retrieval, pages 404 – 408.

Appendix

A. Specific roles of the group members

Xiangyu Guo, Group discussion, Independent implementation
Siddhant Tanpure, Group discussion, Independent implementation
Chenchu Gowtham Yerrapothu, Group discussion, Independent implementation
Alfred Gonsalyes, Group discussion, Independent implementation
Greggory Scherer, Group discussion, Independent implementation