Diversification in Recommendation System

Manjela Toppo

A Thesis Submitted to

Indian Institute of Technology Hyderabad

In Partial Fulfillment of the Requirements for

The Degree of Master of Technology

भारतीय प्रौद्योगिकी संस्थान हैदराबाद

Indian Institute of Technology Hyderabad

Department of Computer Science Engineering

June 2018

Declaration

I declare that this written submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.

Manjela

(Signature)

Manjela Toppo

(Manjela Toppo)

CS16MTECH11007

(Roll No.)

Approval Sheet

This Thesis entitled Diversification in Recommendation System by Manjela Toppo is approved for the degree of Master of Technology from IIT Hyderabad

(Dr. Srijith P.K) Examiner
Computer Science Engineering
IITH

M.V. Pandurga

(Dr. M V Panduranga Rao) Examiner
Computer Science Engineering
IITH

Maunendra Sankar Desarkar

(Dr. Maunendra Sankar Desarkar) Adviser
Computer Science Engineering
IITH

B L Reddy

(T. Bheemarjuna (Reddy) Chairman
Computer Science Engineering
IITH

Acknowledgements

Firstly, I would like to express my gratitude to my adviser Dr. Maunendra Sankar Desankar for the useful guides, comments, remarks and engagement through the learning process of this master thesis.

Abstract

Diversification in Recommendation system However, if it shows many similar items that might become monotonous for the user To handle this scenario is to diversify the recommended list. Diversification helps in recommendation without data(cold start problem) .Diversification maintain the trade off between popularity, freshness and relevance items. In real time Diversification helps in better coverage of items in the recommendation list. It can give emphasis to both novelty and relevance. Novelty means items that contain new information when compared to previously seen ones and covers all the topics. Relevance include top ranked item of the search results.

Contents

Chapter 1

Introduction

Now a days recommendation systems are becoming extremely popular in variety of areas such as news articles, movies, social tags, music, search queries, and products etc. In such recommendation systems user's past history is used for recommendation. Let us consider the case of a movie recommender system. The system will try to recommend movies that are similar to the ones that the user has watched earlier. This similarity can be computed in various different ways. After comparing with other items recommendation system will recommend the best matching items to the user such as what movies to watch. Suppose user is not interested only in one genre and having various interests. Then if recommendation system shows many similar kind of movies that might become monotonous for the user [1]. This scenario can be handled by diversifying the recommendation list. For example user has watched many movies related to genre Adventure and Action then it may happen that all the top entries in the recommendation list contains movies related to genre Adventure and Action. Recommendation appears excellent since the active user clearly appreciates movies related to genre Adventure and Action. However, if active user have several interests other than genre Adventure and Action e.g Animation, Comedy and Fantasy etc. Then the recommendation list of movies will appear poor, owing to its lack of diversity. Diversity is helpful in situations where the users interest shift over time, user is willing to explore different tastes of items, and also when the user does not have fixed or narrow range of interest.

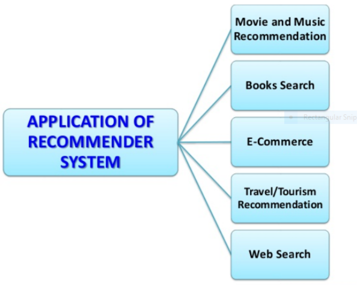
APPLICATION OF RECOMMENDER SYSTEM
Movie and Music Recommendation
Books Search
E-Commerce
Travel/Tourism Recommendation
Web Search

Figure 1.1: Application of recommendation systems [2]

Chapter 2

Motivation

Diversification helps in better coverage of items in the recommendation list. Diversification can give emphasis to both novelty and relevance. It helps in reducing monotony in the recommendation and provide better coverage of items. Novelty helps in recommending items that contain new information when compared to previously seen ones and covers all the topics in the recommendation list. Novelty is needed for a list to get the different number of topics represented in the list. Relevance take top ranked item of the search results in the recommendation list. The idea behind diversified recommendation is to identify a recommendation list of items that are dissimilar with each other but at the same time relevant to the user's interests.Quality of recommendation list is evaluated from more than one metrics. Since user satisfaction is important, accuracy of predicted results is not enough for customer satisfaction, metrics such as novelty, relevance, and diversity are also used to measure the quality of the recommendation list [3]. Diversification improves user satisfaction with recommendation list generated using the common item-based collaborative filtering algorithm.Diversification approach helps in decreasing the intra-list similarity in the recommendation list by diversifying the list. [4]

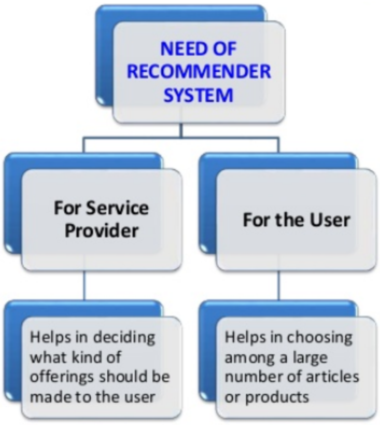
NEED OF RECOMMENDER SYSTEM
For Service Provider
For the User
Helps in deciding what kind of offerings should be made to the user
Helps in choosing among a large number of articles or products

Figure 2.1: Motivation behind recommendation system [2]

Chapter 3

Literature Survey

Recommendation systems typically produce a list of recommendations in one of two ways – through collaborative filtering or through content-based filtering.

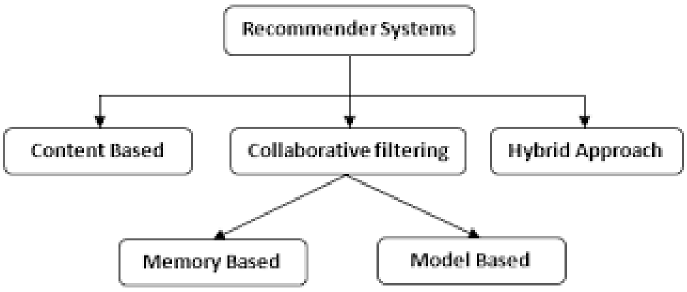

Figure 3.1: Recommendation system [2]

3.1 Collaborative Filtering

Collaborative-Filtering systems concentrate on the relationship between users and items. Similarity of items is determined by the similarity of the ratings of those items by the users who have rated both items.

The main idea behind the Collaborative-Filtering systems is if a person A likes movies Harry Potter, Star war, Avengers and B likes movies Star war, Avengers, Iron man then they have similar interests and A may like movie Iron man and B may like movie Harry Potter.

Several types of Collaborative Filtering :

3.1.1 User-User Collaborative Filtering

User-User Collaborative Filtering find similar customers (based on some similarity measure) and offer products which those similar customers have chosen in past.

This algorithm is very effective but takes a lot of time. It requires to compute every customer pair information which takes time. Therefore for large systems with lots of customers and products this algorithm is hard to implement.

3.1.2 Item-Item Collaborative Filtering

Item-Item Collaborative Filtering will take an item, and then find the users who liked that item and find other items that those users or similar users also liked. It takes items as input, and generates as output other items as recommendations.

It is quite similar to User-User Collaborative Filtering but instead of finding customer look alike, Item-Item Collaborative Filtering finding item look alike. Once item look alike matrix is formed easily alike items recommend to the customer.

This algorithm is better than User-User Collaborative Filtering in resource consumption, as typically the number of products in a system is much lesser than the number of customers/users. Hence for a new customer it takes far lesser time than User-User Collaborative Filtering as we don't need all the similarity scores between customers.

3.1.3 Other algorithms

There are other approaches like Market Basket Analysis, which generally do not have high predictive power than other algorithms.

3.2 Content Based Filtering

Content-based systems construct a profile for each item based on important features of that item. Content-based systems focus on features of the items. Similarity of items is determined by measuring the similarity in their features by using Jaccard distance or Cosine distance.

The profile consists of some features of the item. For example, consider the features of a movie that might be relevant to a recommendation system.

- Some viewers watched movies based on genre like romance, action, comedy, adventure,drama, horror etc.
- Some viewers watched movies based on set of actors working on that movie.
- Some viewers watched movies because of their favorite director's work.
- Some viewers watched movies based on years, some viewers like old movies but some like new releases.

A substitute that has been useful in practice is the identification of words that characterize the topic of a document. Firstly, eliminating stop words means the several hundred most common words, which tend to say little about the topic of a document. For the remaining words, the TF.IDF score is computed for each word in the document. The ones with the highest scores are the words that characterize the document. As features of a document, the n words with the highest TF.IDF scores can be considered. It is possible to pick n to be the same for all documents, or to let n be a fixed percentage of the words in the document. Its also possible to choose all words whose TF.IDF scores are above a given threshold to be apart of the feature set.

Now, documents are represented by sets of words. These words can be expected to express the subjects or main ideas of the document. For example, in movies we take the plot summary of each movie and create a feature set for each movie based on their highest TF.IDF score.

To measure the similarity of two movies, there are several natural distance measures :

1. The Jaccard distance between the sets of words.
2. The cosine distance between the sets, treated as vectors.

To compute the cosine distance of the sets of high TF.IDF words as a vector, with one component for each possible word. Considering the vector has 1 if the word is in the set and 0 if not. Since between two documents there are only a finite number of words among their two sets.

The dot product is the size of the intersection of the two sets of words, and the lengths of the vectors are the square roots of the numbers of words in each set. That calculation lets us compute the cosine of the angle between the vectors as the dot product divided by the product of the vector lengths.

Chapter 4

Solution

4.1 MAXSUM Diversification

The MAXSUM Diversification is to maximize the sum of the relevance and dissimilarity of the candidate set C. It is defined as:

$$f(C) = (k-1)\sum_{x\in S} w(x) + 2\lambda \sum_{x,y\in S} d(x,y)$$

Here $|C| = k$ and $\lambda > 0$ is a parameter specifying the trade-off between relevance and similarity. [5] The objective function need to scale up the fact that there are k(k-1)/2 numbers in the similarity sum and k in the relevance sum. [5]

Algorithm 1 Algorithm for MAXSUM [5]

```
Input: Universe U,k
Output: Set C ($|C|$ = k) that maximizes f(C)
Initialize the set C = $\Phi$
for $j \leftarrow 1$ to $\lfloor k/2 \rfloor$ do
    Find (x,y ) = $argmax_{u,v\in U}$ d(u,v)
    Set C = C $\cup$ {x,y}
    Delete all edges from E that are incident to x or y
end for
If k is odd, add an arbitrary document to C
```

In MAXSUM Diversification algorithm out of all items selecting k candidates and then recommending it to the user. Based on λ value the objective function decide either to concentrate on relevance or novelty. By increasing λ value the objective function will increase means it is concentrating more on novelty .

4.2 MAXMIN Diversification

The main idea of MAXMIN Diversification is to maximizes the minimum relevance and dissimilarity of the selected candidate set C. [5] It is defined as:

$$f(C) = \min_{x \in C} w(x) + \lambda \min_{x,y \in C} d(x, y)$$

Here $|C| = k$ and $\lambda > 0$ is a parameter that specifying the trade-off between relevance and similarity. Here in MAXMIN Diversification the objective function tries to maximizes the minimum relevance by taking the minimum weight of the item among the candidate set and increase the dissimilarity by taking the items from the candidate set which is having minimum distance means they are more similar.

Algorithm 2 Algorithm for MAXMIN [5]

```
Input: Universe U,k
Output: Set C ($|C|$ = k) that maximizes f(C)
Initialize the set C = $\Phi$; Find
(x,y) = $argmax_{u,v \in U}$ d(u,v) and set C = {x,y}; For
any u$\in U \setminus C$, define d(u,C) = $min_x$ $\in$C d(u,v);
while do$|C|$ < k do
    Find u$\in U \setminus C$ such that u= $argmax_{x \in U\ C}$ d(u,C);
    Set C = C $\cup$ {u}
end while
```

In MAXMIN Diversification algorithm out of all items selecting k candidates and then recommending it to the user. Based on λ value the objective function decide either to concentrate on relevance or novelty. By increasing λ value the objective function will increase means it is concentrating more on novelty .

Chapter 5

Results

5.1 Data

5.1.1 Movielens

Dataset is taken from Movielens dataset recommended for education and development . Dataset contains 20000263 ratings and 465564 tag applications across 27278 movies. Data was created by taking details of 138493 users between January 09, 1995 and March 31, 2015. All users in this dataset had rated at least 20 movies. Link to the dataset - http://grouplens.org/datasets/movielens/20m/

The data are contained in six files, genome-scores.csv, genome-tags.csv, links.csv, movies.csv, ratings.csv and tags.csv. Final dataset is prepared by combining all datasets based on Movie ID and IMDB ID.

Table 5.1: This is an example table of Genome-scores.csv file

movieId	tagId	Relevance
1	1	0.025
2	2	0.03975

Table 5.2: This is an example table of Genome-tags.csv file

tagId	tag
1	7
2	007(Series)

Table 5.3: This is an example table of Links.csv file

movieId	imdbId	tmdbId
1	114709	862
2	113497	8844

Table 5.4: This is an example table of movies.csv file

movieId	title	genres
1	Toy story (1995)	Adventure—Animation—Children—Comedy—Fantasy
2	Jumanji (1995)	Adventure—Children—Fantasy

Table 5.5: This is an example table of Ratings.csv file

userId	movieId	rating	timestamp
1	122	2	945544824
2	441	2	1008942733

Table 5.6: This is an example table of tags.csv file

userId	movieId	tag	timestamp
28	63062	Angelina jolie	1263047558
40	4973	poetic	1436439070

Table 5.7: This is an example table of final combined Data.csv file

Parameters	**Movie1 Values**	**Movie2 Values**
movieId	1	10
tagId	1028	1
userId	243342	8667
title	Toy story	GoldenEye
genres	Adventure,Animation, Children,Comedy,Fantasy	Ation,Adventure,Thriller
imdbId	114709	113189
tmdbId	862	710
tag	Time travel	7
timestamp	1223304729	1161199943
rating	5	2.5
ratings_timstamp	1223264174	1161199942
relevance	0.1585	0.9995
released_year	1995	1995

5.1.2 Additional data

[!htb] OMDB API is used for the plot summary of movie and requested owner for API key. By using this API key and by providing movie details like Title, IMDB ID, Released year etc we can get the plot summary of that movie in txt file. By using API key and movie details we got only 1112 plot summaries. Released year is extracted from the title of the movie to get the plot summary of that particular year . Plot summaries from year 2004 to 2016 got extract by using OMDB API.

Table 5.8: This is an example table of Data2014.csv file

Parameters	**Values**
movieId	107769
tagId	406
userId	253445
title	Paranormal Activity:The Marked Ones
genres	Horror—Thriller
imdbId	2473682
tmdbId	227348
tag	franchise
timestamp	1396124094
rating	2
ratings_timstamp	1396124082
relevance	0.99725
released_year	2014

5.2 Evaluation Metrics

5.2.1 Precision

Precision is the fraction of the retrieved items that are relevant. For example : After recommending items,how many items user actually liked. Suppose 10 items are recommended to the user out of which only 4 items are liked by the user then precision will be 0.4 .

$$Precision = \frac{\mid \{relevant\ items\} \cap \{retrived\ items\} \mid}{\mid \{retrived\ items\} \mid} \tag{5.1}$$

	relevant	nonrelevant
retrieved	true positives $t(p)$	false positives $f(p)$
not retrieved	false negatives $f(n)$	true negatives $t(n)$

$$Precision = \frac{tp}{tp + fp} \tag{5.2}$$

5.2.2 Recall

Recall is the fraction of the relevant items that are actually retrieved.Items that were actually liked by the user are recommended. For example : Suppose user likes 10 items and the recommendation system shows 5 out of them then recall in this case is 0.5 .

$$Recall = \frac{\mid \{relevant\ items\} \cap \{retrived\ items\} \mid}{\mid \{relevant\ items\} \mid} \tag{5.3}$$

$$Recall = \frac{tp}{tp + fn} \tag{5.4}$$

5.3 Preprocessing steps

- First, removing underscore or non-alphanumeric and then Tokenization is performed on the plot summary by chopping it up into pieces, called tokens. By doing this it will throw away certain characters such as punctuation.

- Stop words dropping the common words present in the plot summary such as a, an , that ,from , were etc. And then removing the words whose length is less than 2.

- Stemming is used to reduce inflectional forms to derive or base form of word such as organize, organizes, organizing , organization etc considering all as single word.

5.4 LDA

In Natural Language Processing (NLP), latent Dirichlet allocation (LDA) is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar.LDA can be used for topic modelling. Topics are probability distribution over words.

For example, if observations are words collected into documents, it posits that each document is a mixture of a small number of topics and that each word's creation is attributable to one of the document's topics. Top keywords: accept, accident, charact, death, deal , young, world,warrior, etc.

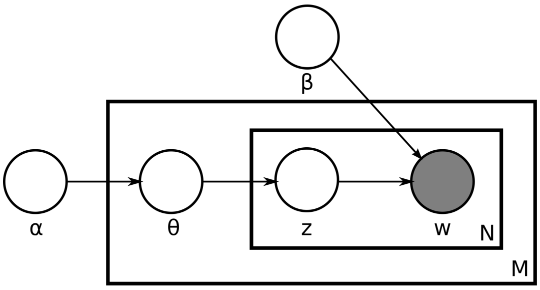

Figure 5.1: Graphical model representation of LDA. The boxes are "plates" representing replicates. The outer plate represents documents, while the inner plate represents the repeated choice of topics and words within a document. [6]

LDA representation contains three level of representation . The parameters α and β are corpus level parameters, assumed to be sampled once in the process of generating a corpus. The variables θ_d are document-level variables, sampled once per document. Then at last, the variables z_{dn} and w_{dn} are word-level variables and are sampled once for each word in each document [6].

5.5 Performance Evaluation

Top-N performance can be directly measured by different methods based on accuracy metrics such as precision and recall. [7]

Table 5.9: This is result table.

Average Precision	**Value**
Precision@top10	0.028
Precision@top15	0.041
Precision@top30	0.055

Table 5.10: This is result table.

Average Recall	**Value**
Recall@top10	0.018
Recall@top15	0.027
Recall@top30	0.040

Results of average precision and recall is not good because of spare data. Plot summaries of movies were very limited.In final data few users has watched only 1 or 2 or 5 movies and so on.If fixed some threshold like if taking only those users who has watched atleast 30 or 50 movies then precision and recall will increase and give better results.

Chapter 6

Conclusion

Diversification helps in better coverage of recommendation list.Diversification minimize the query abandonment means user will find at least one relevant document in the recommendation list. Diversification can increase the number of satisfied users. In case of popularity based recommendation system using diversification algorithms ”cold start” problem can be handle. It is also useful for anonymous users who want to hide their identities .

References

[1] M. S. Desarkar and N. Shinde. Diversification in news recommendation for privacy concerned users. In 2014 International Conference on Data Science and Advanced Analytics (DSAA). 2014 135–141.

[2] Shivam. Recommender System 2005.

[3] M. Karakaya and T. Aytekin. Effective methods for increasing aggregate diversity in recommender systems .

[4] Improving recommendation lists through topic diversification. 2005 .

[5] S. Gollapudi and A. Sharma. An axiomatic approach for result diversification. In WWW. 2009 .

[6] M. I. J. David M. Blei, Andrew Y. Ng. Latent dirichlet allocation. In The Journal of Machine Learning Research. 2003 993–1022.

[7] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-N recommendation tasks. 2010.