SYSTEM FOR INTERPRETATION OF REVIEWER FEEDBACK ON SOCIAL WEB

by

SHRIKANT DESAI

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE & ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2012

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Gautam Das, giving me opportunity to work on this project and also for his constant guidance and support. It was privilege to be a part of DBXLab last two years. I am also grateful to Dr. Bahram Khalili and Dr. Leonidas Fegaras for serving on my committee.

I would like to specially thank Saravanan and Mahashweta for all the support and help during my thesis.

I would also like to thank my family members who have always been a source of inspiration and support and friends for their support for this project and throughout my academic career.

November 9, 2012

ABSTRACT

SYSTEM FOR INTERPRETATION OF REVIEWER
FEEDBACK ON SOCIAL WEB

SHRIKANT DESAI, M.S

The University of Texas at Arlington, 2012

Supervising Professor: Gautam Das

The increasing popularity of social media web sites such as Amazon, Yelp and others has influenced our online decision making. Before making selection decisions on movies and restaurants, we investigate its reviewer feedback. Social media web sites provide reviewer feedback in the form of ratings, tags, and user reviews. However, overwhelming feedback details will leave the user in a quandary as to decide whether the item is desirable or not. Potential buyers either make a snap judgment based on the aggregate ratings/tags or spend a lot of time in reading reviews.

In this thesis, we build a system that can analyse the reviewer feedback in the form of ratings or tags and generate meaningful interpretations. One of major component is rating interpretation that generates meaningful interpretation of the reviewer ratings associated with the item of interest. For example, given the movie *Titanic*, our system returns results such as, *Young female from California* like this movie instead of average rating 7.6 from all reviewers. Furthermore, end users will be allowed to systematically explore, visualize, and observe rating patterns. Additionally, our system can also explore the social tagging behavior on the input items. For

example, our system can identify movies where similar users have assigned similar tags on diverse items. The tagging behavior of different subpopulation is compared using tag clouds. We use movieLens data set to demonstrate our experiments.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

Most of the users on social web rely on collaborative reviewer feedback to make online purchasing decisions. For example, one can examine ratings and tags associated with restaurants from web sites like Yelp[1] before visiting it, and online buyers shop items relying on reviews of Amazon[2]. Large Number of feedbacks in terms of ratings, tags and/or reviews may be received on one single item. The movie *Dark Knight Rises* received more than 400,000 ratings and 2000 reviews within 4 months on IMDb. Such an overwhelming feedback information makes impossible for a user to make informed decisions about the movie. For a movie-goer , it is not possible to read each and every review. In such situation users makes decision about the item of interest by overall aggregate ratings(average rating for *Dark Knight Rises is 8.8*). It is obvious that most of the users choose aggregate ratings over spending lot a of time reading reviews.

Most of collaborative feedback sites provide some options/features to reduce the information overload to end users. For example, Amazon web site for buying an electronic item provides aggregate ratings but these aggregate ratings do not facilitate detailed decision making. However, IMDb provides single aggregate ratings distributions over predefined user demographics such as age and gender but these ratings do not justify ideal decision making.

In other words, most of the collaborative feedback sites will not allow users to explore the customer feedback and generate meaningful patterns that will help users.

[1]http://www.yelp.com

[2]http://www.amazon.com

Users are left on their own to make a decision about their item of interest. Most of the collaborative sites contain profile information of reviewers who provide feedback on items they purchased/experienced. Such information can be utilized to provide better and more meaningful interpretation of the reviewers feedback.

Meaningful Interpretation of Reviewer Feedback: The mail goal of work is to build a system that can provide meaningful interpretation of customer feedback in terms of ratings and tags by utilizing the additional details about items and user demographics available. We utilize the user demographics from the social web to interpret the feedbacks. [1] and [2] provide frameworks for interpretation of ratings and tags respectively. we build a practical system based on [1] and [2] to help movie goers make better decisions. We achieve this goal by, providing meaning interpretation of reviewer feedback and allowing user to explore and visualize the feedbacks. We introduce two different approaches used in this thesis for the analysis of reviewer feedbacks.

Interpretation of Ratings: We introduce two mining measures, *Similarity Mining(SM)* and *Diversity Mining(SM)* for the interpretation of ratings. The task of *Similarity Mining(SM)* is to identify the user groups who agree on each others ratings.

Consider the movie *Batman* that has received more than 150,000 ratings in IMDb. Sites like IMDb either provide overall aggregate information (Batman has 8.8 average rating) or aggregate rating along pre-defined populations (such as Male users give 9.1 rating). In Similarity mining, we identify groups of users who have similar rating/tagging behavior. For example, given the query Batman, we might identify *Young Female users from California* and *Male students from New York* have rated the movie similarly.In *Diversity Mining* task, we identify groups of users who have different rating/tagging behavior. query *Batman*, we identify *Old male from Texas* and *Young Male from California* have rated differently. We apply these mining mea-

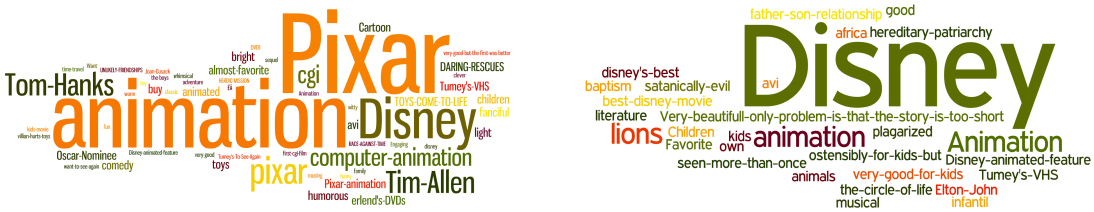

Figure 1.1. Tag Cloud for California and New York.

sures for multiple input items such as ratings of all the *Jack Nicholson* movies. End users can also provide conjunctive query like ratings of all *drama* movies by *Jack Nicholson*.

Furthermore, results of such input query will be displayed on a geographic map, where user groups are rendered over the map as icons. Users are further allowed to explore the user groups and rating distribution of sub-populations of user group and observe the change in rating patterns over time period.

Interpretation of Tags: We utilise the algorithms defined in [2] to build a system that can mine interesting tagging behaviors by applying the notion of similarity/diversity on all the three dimensions (user,item,tags). As an example our system can identify that *Young male users* and *old male users* have similar tagging behavior for the movie *Godfather* but different behavior for *Star Wars*. Also, *women* tag *Quentin Tarantino's* movies negatively while *men* tag it positively. We use tag clouds to visualize the tagging behavior of users. End users will be allowed to explore and compare the tag clouds belonging to different user groups.

Figure 1.1, shows the tag cloud for California and New York generated for the input *Toy Story* Movies. We can observe that tags such as *Disney, Animation, children* are common to both. In this thesis, we allow end users compare the tag clouds and identify interesting patterns.

We explain data models used for system in section 2. In section 3, we explain dual mining approach used in interpretation of tags. Dual mining for tags interpretation will be discussed in section 4. Furthermore, we discuss architecture and user interface of our system in section 5 and 6 respectively.

CHAPTER 2

DATA MODEL

In this section we define data model of a reviewer feedback system. First, we define data model for rating interpretation system. Second, we define data model for tags interpretation system.

Data Model For Rating Interpretation: For a collaborative rating site D, we represent model as a triple $\langle I, U, R\rangle$, that represents a set of *items*, *reviewers*, and *ratings* respectively. A rating $r \in R$, is a triple $\langle i, u, s\rangle$, where $i \in I$, $u \in U$ and $s \in [1, 5]$ (represents a scalar rating given to item i by a reviewer u). Ratings are converted in to a scale ranging from one to five. Each item $i \in I$, is represented as schema, $I_A = \{i.a_1, i.a_2, \ldots\}$ where $i.a_j$ represents the attributes of the item i. Furthermore, each tuple i is presented as, $i = \{i.v_1, i.v_2, \ldots\}$ where each $i.v_j$ is the value for the item attribute $i.a_j$. Similarly, every reviewer $u \in U$ is described by a set of attributes represented as schema, $U_A = \{u.a_1, u.a_2, \ldots\}$ where $u.a_j$ represents the attributes of user u. Furthermore, each tuple in u is presented as $u = \{u.v_1, u.v_2, \ldots\}$ where each $u.v_j$ is the value of the attribute reviewer attribute $u.a_j$. Therefore each rating $r = \{i, u, s\}$ is a set $\{i.v_1, i.v_2, \ldots, u.v_1, u.v_2, \ldots, s\}$ which is concatenation of item attributes from I, reviewer attributes from U, and an integer rating s. This rating set is represented as $A = \{a_1, a_2, \ldots\}$.

Collaborative rating sites such as Yelp, CNET, Amazon, and IMDb usually provide item attributes. For an instance, in IMDb web site, for a movie: *Dark knight* as an item, provides attributes such as Genre, Title, and Released date and some

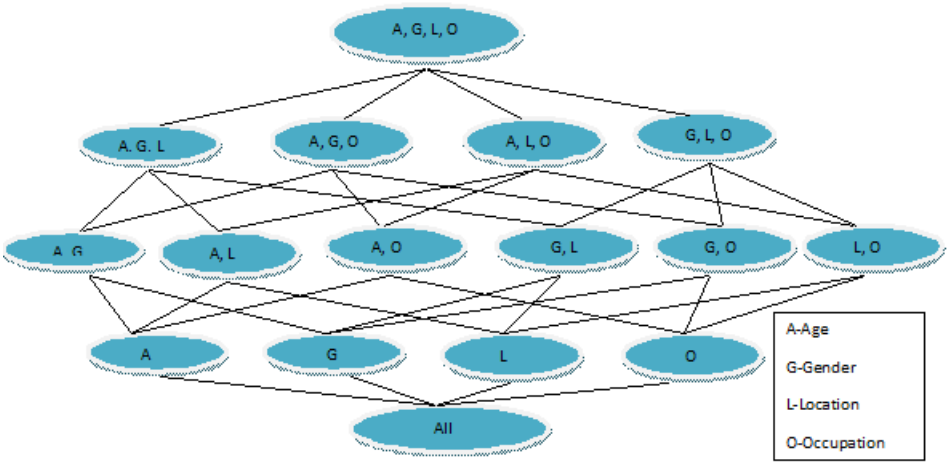

Figure 2.1. Sample data cube lattice.

multi valued attributes such as actors and directors. Similarly,Yelp web site describe the items by attributes such as category, cuisine, price range, and Attire.

The availability of user demographic information is rarer. However some sites such as movieLens, Yelp etc provide anonymized user demographic details. Common attributes are age, gender, location, and occupation. Sometimes the user attributes can collected from social network sites such as twitter, Facebook as many of collaborative sites integrated with social network sites. In this thesis, we present a system that accepts inputs as item attributes and show meaningful results as groups of user attributes that allows reviewers to decide on item desirability. Our system provides geo-visualization of the user groups and allows the reviewers explore the ratings.

We present the results as data cube. The notion of user groups will be modeled as data cube [1] [3]. The data cubes can be constructed from user and item dimensions. Each cube contain a combination of one or more user/item attributes with the average rating. A sample user data cube look like

$\langle Location, Texas\rangle$,$\langle Actor, TomCruise\rangle$,4.2

means, average rating for actor *Tom Cruise* by users of *Texas* is *4.2*. Cube can be formally defined as $c = \{\langle a_1, v_1\rangle, \langle a_2, v_2\rangle, \ldots\}$ where v_i is the value of an attribute $a_i \in A$ and A is the set of all attributes from reviewers,items and rating. Figure 2.1, represents the sample lattice constructed out of four attributes A ,G, L, and O corresponds to user attributes age, gender, location, and occupation respectively.

Data Model For Tags Interpretation: In this section we model data on social tagging site as a triple $\langle U, I, T\rangle$, Where U, I, and T represents set of *users*, a set of *items* and a set of *tags* respectively. Furthermore, tagging action is again triple represented as $\langle u, i, t\rangle$, where $u \in U$, $i \in I$, $t \subset T$. Therefore, we denote tagging action group as $g = \{\langle u_1, i_1, t_1\rangle, \langle u_2, i_2, t_2\rangle, \ldots, \}$. We represent user schema as, $U = \langle a_1, a_2, \ldots\rangle$, attributes values for user schema is denoted as $u = \langle u.a_1, u.a_2, \ldots\rangle$, where each $u.a_x$ is a value for the attribute $a_x \in U$. For example, let $U = \langle age, gender, state, city\rangle$, a sample tuple for such schema is represented as $\langle Young, Male, Texas, Dallas\rangle$ shown. Similarly, we define Item schema as $I = \langle a_1, a_2, \ldots\rangle$, attributes values for user schema is denoted as, $i = \langle i.a_1, i.a_2, \ldots\rangle$, where each $i.a_y$ is a value for the attribute $a_y \in I$.

Therefore, every tagging action is concatenation of item attributes, user attributes, and the set of tags. A tuple of tagging action is represented as, $r = \langle r_u.a_1,$ $r_u.a_2, \ldots, r_i.a_1, r_i.a_2, \ldots, t\rangle$. For example, a sample tuple from movieLens data set look like

$\{\langle Gender, male\rangle, \langle Location, California\rangle, \langle Movie, Toy\ story\rangle,$

$\langle Disney, Children, Anime\rangle\}$.

Where,

$\{\langle Gender, male\rangle, langleLocation, California\rangle\} \in u$,

$\{\langle Movie, Toy\ Story\rangle\} \in i$ and

$\{\langle Disney, Children, Anime\rangle\} \in t$

We define T is a set of such tuples. Social web contains large number of such tuples.

CHAPTER 3

COLLABORATIVE RATING INTERPRETATION

In this section, we describe our approach to generate meaningful interpretation of reviewer ratings. Our work is inspired by the description and difference mining problems in [1] [4]. In our system, we provide interpretations through the prism of similarity and diversity. As noted before in the section 1 existing social content sites provide either overall aggregate ratings or aggregate based on predefined user demographics. Our main goal is identifying the user describable groups dynamically that can help users to make immediate decisions on item of interest. For example, suppose movie is *Social Network*, we display user cuboid that describe *Male Student from Massachusetts like movie social network with aggregate rating 8.2* instead of just saying aggregate rating is 7.9.

3.1 Dual Mining on Collaborative Ratings

Our system allows users to specify search query as an item or set of items and return user groups represented as data cubes. We select such user groups by building a lattice structure.Data cube in lattice represents the user groups. For example, we build a lattice from the user attributes, Age(A), Gender(G), Location(L), and Occupation (O). Each cube in the lattice corresponds to a distinct number of tuples(user groups) and their average rating. The edges between the nodes represent parent and child relationship. One example of such lattice built on user demographics is as shown in Figure. 2.1.

Even for a single input item the number of possible cuboids generated is too large. It is not possible for an end user to browse all the cuboids. Therefore, we select a *good groups* of cuboids, k from the all the available cuboids, n. We adopt following techniques to identify the *good groups* [1]. A group belongs to *qualifying group* by satisfying following criteria. First, ratings must be consistent within the group, the consistency within a group is the significance of similar opinion within the group where as opinion across the groups is the measure of diversity among the user groups. Second, collective ratings by the user groups must cover significant portion of available ratings. Third, groups must be easy to understand and interpret. For the purpose of visualization of the groups on geo-conditions, we discard the data cubes without location attribute. This factor adds in reducing the number of cuboids.

3.1.1 Similarity Mining for ratings

In this section we define *Similarity Mining*. Given a set if items I, our main task is to identify the user groups such that user groups agree each others ratings. To achieve this, we consider following factors. First, the number of user groups(*number of cuboids*), k identified must be a small number because end users must not be confused with overwhelmed cuboids. End users will be given a choice to restrict the number of cuboids while specifying search query. Second, user groups provided to the users must cover enough part of ratings which is mentioned as a part of end users query. Third, for the purpose of visualization, we select the cuboids at least having attributes containing *geo-location*. Finally, we select the group which that has minimum aggregate error.

We define notion for coverage and aggregate error [1] as follows:

$$coverage(C, R_I) = \frac{|C_{RI}|}{|R_I|} \tag{3.1}$$

$$error(C, R_I) = \sum_{r \in R_I} (E_r) = avg(|r.s - avg_{c \in C \wedge r < c}(C)|) \tag{3.2}$$

where $ri.s$ is the score for the tuple i

3.1.2 Diversity Mining for ratings

In this section we define another measure to interpret reviewers rating called *diversity mining*. In this method we identify the set of user groups who disagree each others rating. More formally, given a set of items I and ratings set R_I identify the set of user groups (cuboids), U_A such that opinion across the user group is differing. To identify the such groups, we divide rating set R_I into two sets i.e, ${R_I}^+ = \{r | r \in R_I \wedge r.s \geq \theta^+\}$ and ${R_I}^- = \{r | r \in R_I \wedge r.s \geq \theta^-\}$, where θ^+ and θ^- are the thresholds defines rating positivity and negativity. According to mean and variance of rating set, R_I θ^+ and θ^- are calculated statistically or dynamically. In our system we use dynamic approach to set θ^+ above the standard deviation and θ^- below standard deviation.

Given sets ${R_I}^+$ and ${R_I}^i$, and set of cuboids C, we measure the diversity of user groups using *balance* factor defined as [1]:

$$balance(C, {R_I}^+, {R_I}^-) = \frac{\sum_{r_1 \in {R_I}^+ \wedge r_2 \in {R_I}^-} I(r_1, r_2)}{|{R_I}^+| \times |{R_I}^-|} \tag{3.3}$$

where, $I(r_1, r_2) = 1$ if and only if $\exists c \in C$ s.t $r1 < c \wedge r2 < c$. The high balance is indication that ratings are *mingled togethers* and low balance indicates ratings are *separated apart*.

Implementation of Similarity and Diversity Mining: We describe the algorithms used in our system for *similarity mining* and *diversity mining*. Our goal is, given a set of items I and the set of R_I identify k number of user groups such that 1) aggregate error $error(C, R_I)$ is minimised and ratings cover α percentage of ratings (k and α are

specified by the end users). 2) balance factor, $balance(C, {R_I}^+, {R_I}^-)$ is minimised and coverage constraint, $coverage(C, {R_I}^+) \geq \alpha$ and $coverage(C, {R_I}^-) \geq \alpha$ is satisfied.For this purpose we use *Randomized Hill Exploration (RHE)* algorithm [1].

Randomized Hill Exploration (RHE): Randomized Hill Exploration algorithm selects k cuboids randomly as a first step initialization. In the next steps, it starts exploring the parent/child nodes that satisfy the coverage constraints α instead of improving the aggregate error (E_r) immediately. Once new set of cuboids that satisfy coverage constraint is identified, new cuboids will be replaced. Furthermore, these algorithms are used for error optimization and balance optimization on cuboids for similarity mining and diversity mining respectively.

In our system, end user specify the input set I, constraints like number of cuboids, k and percentage of coverage α as search query. Furthermore, we build a lattice for set R_I as shown in Figure 2.1. *RHE* algorithm will be run on the lattice. Suppose lattice contain C cuboids, k random cuboids will be selected from the set C. coverage constraint, $coverage(C, R_I)$ for k cuboids will be checked for coverage constraint, α. If $coverage(C, R_I)$ is not satisfied, each cuboid c_i that belongs to C will be replaced with its neighbor(parent/child) cuboid c_j until $coverage(C, R_I)$ is greater than or equal to α.

For similarity mining, the k cuboids that satisfies $coverage(C, R_I) > \alpha$ with minimum aggregate error $error(C, R_I)$ are returned to the end user. For diversity mining, the k cuboids that satisfies $coverage(C, {R_I}^+) \geq \alpha$ and $coverage(C, {R_I}^-) \geq \alpha$ with minimum aggregate balance $balance(C, {R_I}^+, {R_I}^-)$ are returned to the end user. Our system support visualization of the such cuboids on geographical map. We make *location* attribute mandatory while constructing the lattice that reduces number of cuboids and achieve computational efficiency also.

CHAPTER 4

COLLABORATIVE TAGS INTERPRETATION

4.1 Dual Mining on Collaborative Tags

In this section we explore tagging behaviour on social web. A typical tagging action involves three components: users, items, and, tags. We build our tagging interpretation system by applying two alternative measures similarity and diversity on tagging components [2]. These mining measure produce groups of similar or diverse items tagged by similar or diverse user groups with similar or diverse tags. For example, *Male aged under 18 and female aged more than 45 use diverse tags for the movie Mission Impossible*. A large number of such problem instances arise as we would like to explore similar and diverse nature of the each components.

In the following sections we describe our approach of dual mining on tags component. We consider user and item dimensions together as they resemble to each other while mining. However, mining tag dimension will be different from user/item mining.

Dual Mining on Tag Components: User group is a set of $\langle attribute, value \rangle$ pairs. Therefore given user groups, we compute similarity/diversity (Dual mining) between the user groups in two different ways, 1) Structural distance between user group description and 2) Set distance between the user groups based on the items rated by them.

Given two user describable tagging action groups, u_1 and u_2 and U_A is the set of attributes shared between u_1 and u_2. $a.v_1$ and $a.v_2$ are set of user attribute value pair, where $a \in U_A$. Therefore, user similarity function is defined as shown in 4.1.

$$f(u_1, u_2, users, similarity) = \sum_{a \in U_A} sim(v_1, v_2) \tag{4.1}$$

sim is the domain aware similarity function that computes distance between two or more values. Similarly, $f(u_1, u_2, users, diversity)$ will function that measures diverse user groups.

Suppose $u_1.I$ and $u_2.I$ is the set of items tagged by user groups u_1 and u_2. The set difference will be percentage of items tagged by both groups. Similarity function set difference is as shown in 4.2.

$$f(u_1, u_2, users, similarity) = \frac{\{r | r \in u_1.I \wedge r \in u_2.I\}}{\{r | r \in u_1.I \vee r \in u_2.I\}} \tag{4.2}$$

Similarly, $f(u_1, u_2, users, diversity)$ will function that measures diverse user groups.

Unlike user and item dimensions, two or more tag dimensions cannot be compared with each other easily. The reason being, tags are chosen freely from diverse vocabularies as a result tag dimension might have large number of tags and tag dimension will not contain fixed set of attributes as compared to user/item dimension.In this thesis, we follow two step process handle tag dimensions [2]. First, we build set of tag signatures (a smaller representation) from the tagging actions. Second, we compare two or more tag signatures by comparing the distance between them.

Tag Signature: A typical tagging action will be as defined in section 2 represented as $g = \{\langle u_1, i_1, t_1\rangle, \langle u_2, i_2, t_2\rangle, \ldots\}$. Let $t_g = \{t_1 \cup t_2 \cup \ldots\}$ represents a set of all tags from tag component. There are several ways to build a tag signature. For example, if tags are manually chosen then number of unique tags will be small. In such case tag signature is represented as $T_s = \{t, freq(t) | t \in t_g\}$ where $freq(t)$ is number of occurrences of tag $t \in t_g$.

However, most of the tags created in social web are diverse and sparse which makes t_g very large. Therefore we use technique called Latent Dirichlet Allocation (LDA) for generating tag signature. LDA a reduces the tag set by aggregating tags into topics based on their relevance hence by reducing tags long tail issue [5]. In our work we decide the topics such that all tag signatures generated must belong to predefined topics.

Tag Signature Comparison: The second step will be comparing the tag signatures. The distance between tag signature is computed using cosine similarity function as define in 4.3, where θ is angle between tag vectors $T_s(g_1)$ and $T_s(g_2)$.

$$f(u_1, u_2, users, similarity) = \cos(\theta(T_s(g_1), T_s(g_2))) \tag{4.3}$$

Similarly, $f(u_1, u_2, users, similarity) = \cos(\theta(T_s(g_1), T_s(g_2)))$ can be defined as diversity.

We use algorithms Locality Sensitive Hashing(LSH) based algorithm for similarity mining and FDP facility dispersion problem (FDP) based algorithm for diversity mining [2]. Our main goal is given input from item/user components with constraints like similarity/diversity, identify set of tag components that maximize similarity and diversity. In the following sections we explain each of the algorithms briefly.

Similarity Mining of Tag Component by LSH Algorithm: We use LSH based algorithm to perform similarity mining on tag components. LSH algorithm solves nearest neighbor problem in higher dimensions [6]. The main idea of LSH is similar input items are hashed into same bucket such that input items that are close to each other in higher dimension gets mapped to the same item in lower dimension with higher probability. LSH guarantees that probability of two close points falling into same bucket is higher than probability of probability of two far points falling into same bucket. We use a family of hash function based on cosine similarity [7]. Co-

sine similarity between tag signature will be computed as defined in 4.3. We apply pairwise comparison function, $f(u_1, u_2, users, similarity)$ on tag signatures vectors to optimizes the tag similarity. Furthermore, k closest vectors with minimum pair wise distance, are identified as the result set that maximize the tag similarity. It is obvious that diversity of cannot be computed using hash function with retention of LSH properties. Therefore, we use FDP based algorithms to maximize the diversity of tag components.

Diversity Mining of Tag Component by FDP Algorithm: We adopted the Facility dispersion problem (FDP) based algorithm from [2] in this thesis to maximize the tag components diversity. We can compute the average pair-wise distance from FDP problem. The tagging action groups having maximum average pair-wise distance are considered to de dissimilar groups. We consider n tagging actions groups with d-dimensional tag signature groups and our aim is to identify k vectors with maximum average pair-wise distance. We use pairwise comparison function, $f(u_1, u_2, users, diversity)$ to compare the distance between tagging action groups.

CHAPTER 5

ARCHITECTURE

In this section we explain system architecture. Our system will have three major components: front end, web service, and back end as show in the Figure 5.1. We explain each of these components in detail with respect to rating mining and tags mining.

5.1 Architecture of Rating Mining

The architecture of the rating mining system as shown in Figure 5.1. The four major components of this architecture are explained in the following sections.

Front end: We design our front end using technologies like HTML5, Javascript, Jquery, raphael, bootstrap and ajax. User interface of rating system is divided into three components: input, visualization, and exploration.

First, the input screen will have an option for user to enter the query. User can enter the query by specifying the one or more item attributes from set I_A. End user can specify the parameters like number of cuboids (k), coverage constraint (α),and the time interval. Second part, will be a visualization of the k cuboids. Visualization will have tabs for *similarity mining* and *diversity mining*. Each tab contains, a geographical map. Cuboids will be overlayed on the map as icons displaying the user attributes like age, gender, and occupation. Third part will be, exploring the icons. End users are allowed to click on the icons rendered over the geographical map. Furthermore, statistical distribution of rating of the group will be displayed on the screen. Similarly, diversity mining tab can be selected and experience the

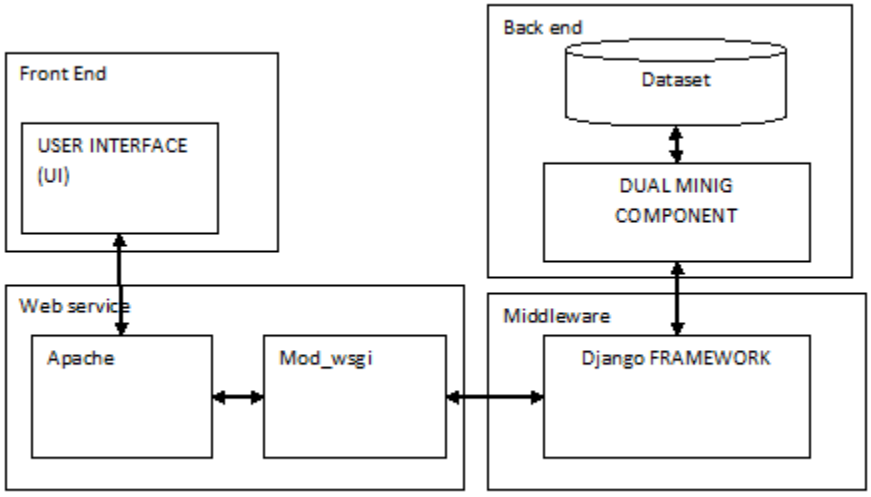

Figure 5.1. Architecture of Reviewer Feedback System.

visualization as well as exploration options.

Web Service: We host our system in using *Apache* web server. We use *mod_wsgi* along with Apache because of its suitability to host high performance production web sites that run on python applications.

Back end: In the back end, we use *Drongo*, a open source web 2.0 application framework which follows model-view-controller architectural pattern. As shown in the 5.1 django framework acts as intermediate system between dual mining system and web service. Input sets from user interface will be sent dual mining system through django framework. Dual mining system perform similarity mining and diversity mining on the available data set.

Data set: We use movieLens 1M [1] data set for experiments which contains, 3900 movies and 6400 users with one million ratings. The data set contain three files. First, user file with details such as age, gender, zip code, and occupation. Second,

[1]http://www.grouplens.org/node/73

movie file containing attributes such as movie id, movie title, and genre. Third, Ratings file contains movie id, user id, rating and time stamp.

We require more additional information such as actors, directors and their pictures for the purpose of user interface. We extract these information from IMDb API [2] which provides a json with addition detail such as actors, directors, and poster. As our user interface allows users to query on any of the movie attributes (Movie Title, Actors, Director, and Genre), we create separate rating files for each of the attribute to improve the performance and avoid latency.

5.2 Architecture of Tags Mining

The Architecture of the tag mining framework is almost similar to the rating mining architecture as shown in Figure.5.1. In this section we explain the Tags mining architecture.

Front End: We use the same technologies used in rating mining. However, input for tags mining is more complex than the rating mining. In this case end users are allowed to specify the search query that involves attributes of user dimension(u_a) and/or item dimensions(i_a). End users are also given an option to specify *threshold* for user/item attribute inputs and *maximization* constraint for tags similarity or diversity. Furthermore, similarity and diversity mining is performed on tagging components based on input query. Output of such query will be a results corresponding to similarity and diversity measures. Results will be displayed on geographic map. User and item dimensions in the result set overlayed on the map. Furthermore, we display tag clouds associated with each result on screen to allow users to compare the results.

Web Service: We use the same web service used in rating mining.

[2] http://www.imdbapi.com

Back End: We use same technologies used in rating mining except the dual mining framework. We use different set of algorithms to process tagging components.

Data set: For tag interpretation we use movieLens 10M [3] data set which contain, 100,000 tags by applied to 10,000 movies by 72,000 users. The data set contain tag file with 100,000 tags, user IDs and movie IDs associated with it.

[3]http://www.grouplens.org/node/73

CHAPTER 6

USER INTERFACE DESCRIPTION

6.1 Rating Interpretation System

We build rating interpretation system similar to MapRat system [4]. Our rating system can work on any of the collaborative rating site which provides data as described in section 2.

Our rating interpretation system will have web based user interface. Users will allowed to enter one or more items attributes. The list of input parameters include: *movie, actor, director, genre* and mandatory parameters such as *coverage* and *number of groups*. As an additional input we allow user to specify the time frame with a slider. As shown in the Figure 6.1, user enters the movie as *Toy Story* with number of cuboids, $k = 10$ and coverage $\alpha = 30$ and time frame between *Jan 2000* to *Dec 2001*. Clicking on *Explain Ratings* button generates the output.

The result of such query is as shown in the Figure 6.2. The groups of similar and diverse users are placed in two separate tabs. The tabs are the solutions to the two sub-problems (similarity mining and diversity mining) described in the section 3.1. For the visualization of user groups on geographical map, our systems always identifies the user groups which have at least location attribute in it. We use raphael[1] to build and color the geographical map. We represent the average ratings with colors such red for rating 1 and green for rating 5. We use icons to describe the user attributes. Color of pin holding icons describe the age group of the user group. We display other attributes like Gender and occupation with the specific icons. Legend for ratings age

[1] http://raphaeljs.com/

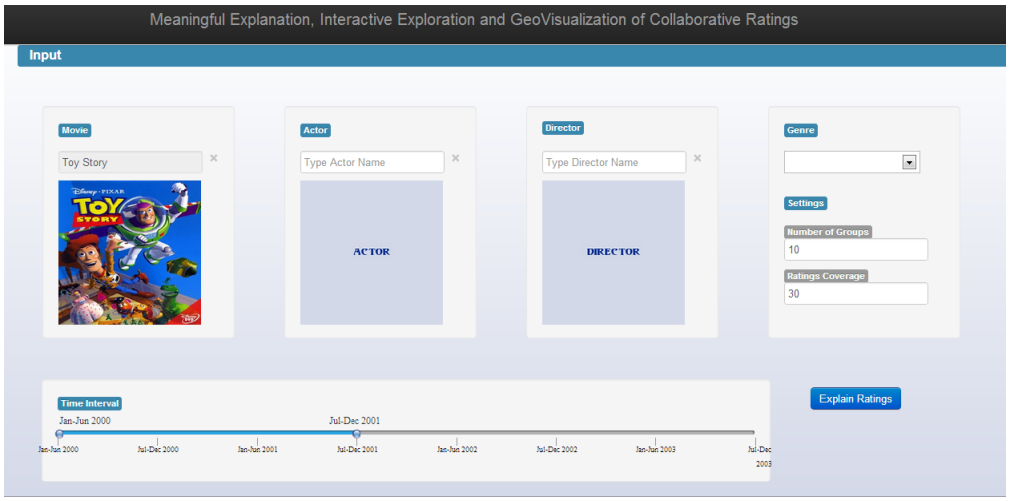

Figure 6.1. User Interface for Ratings mining .

group and icons will be shown on the right hand side of the map. User can hover over the groups for the description of the group.

As shown in the Figure 6.2, 10 user groups who agree on each other ratings is shown on the geographical map. The example of user hovered over icon on Michigan a tool tip with a description *Young Male student from Michigan*. Similarly, user can identify each of the user group by moving cursor or manually with the help of legend.

User can click on one of icons (user groups), to explore the ratings. Additional statistics about the user group rating will be displayed on the screen. For example, End user can click on *male icon on California*.User groups statistics will be displayed on the screen. Which contains a ratings patterns for *all male user from California*, *Rating distribution of all California users*, and *Rating distribution of all male users*. Hence, such user groups and statistics helps user in decision making about the item of interest.

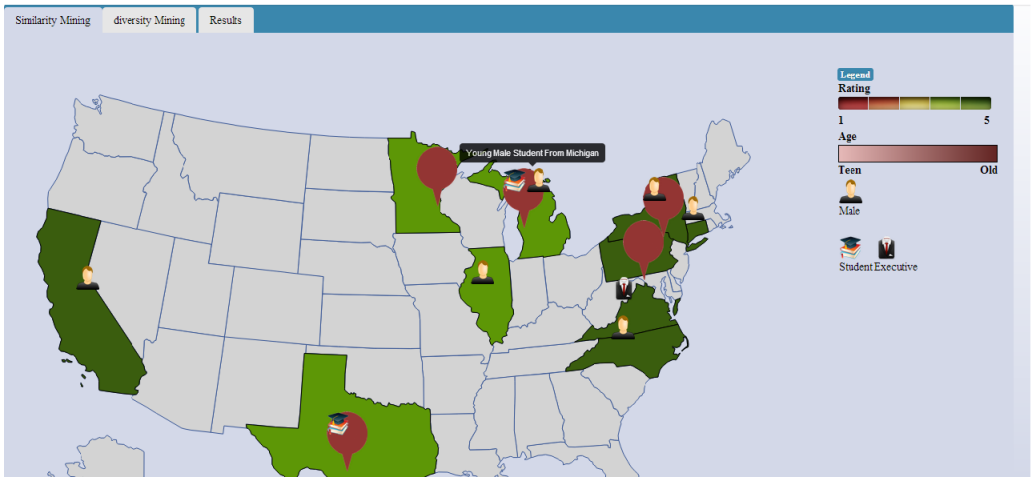

Figure 6.2. Geo-visualization of Ratings mining .

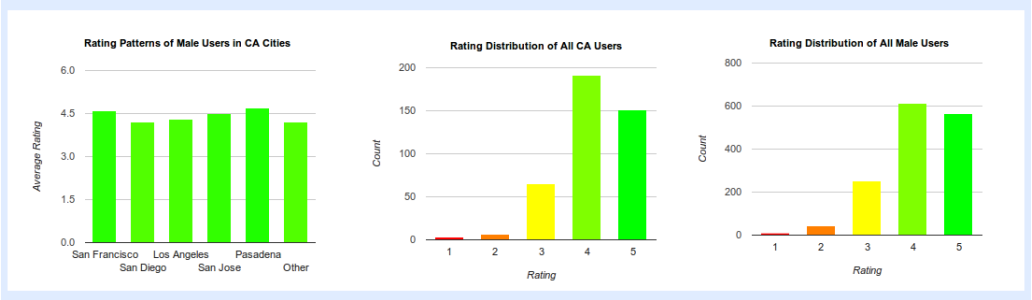

Figure 6.3. User Group exploration. .

6.2 Tags Interpretation System

Our tags interpretation works on social web that provides the data model described in section 2.

Tagging interpretation will have a web based interface as shown in the figure 6.4. The left hand side of the input screen contain a drop down that allows a end user to choose different attributes from user and item dimension. On selecting user/item attributes, it will be added to the query part of user interface. Right hand side of the input screen contains settings for the mining(*Similarity/Diversity*) of tag components. In this way user can specify query to identify similar(or diverse) tags on similar(or

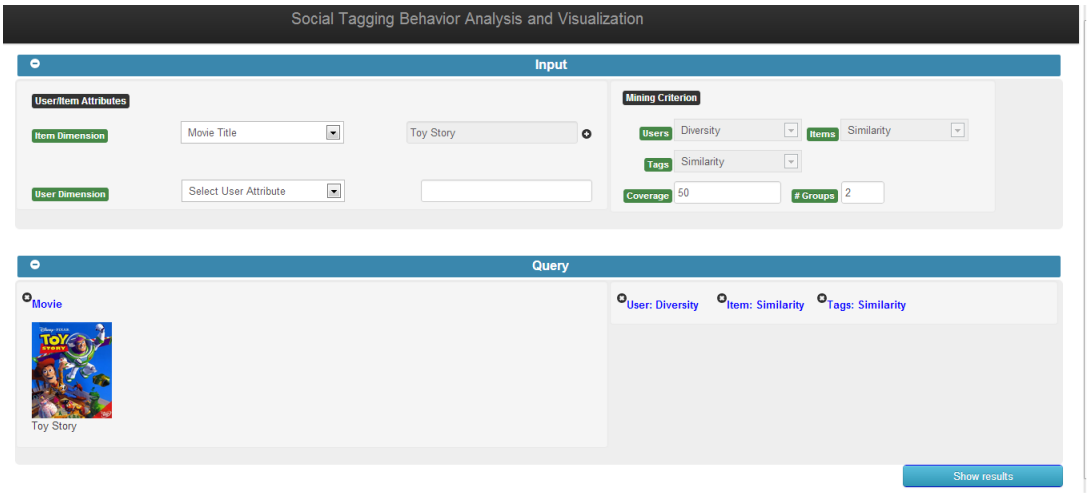

Figure 6.4. User Interface for Tags Mining. .

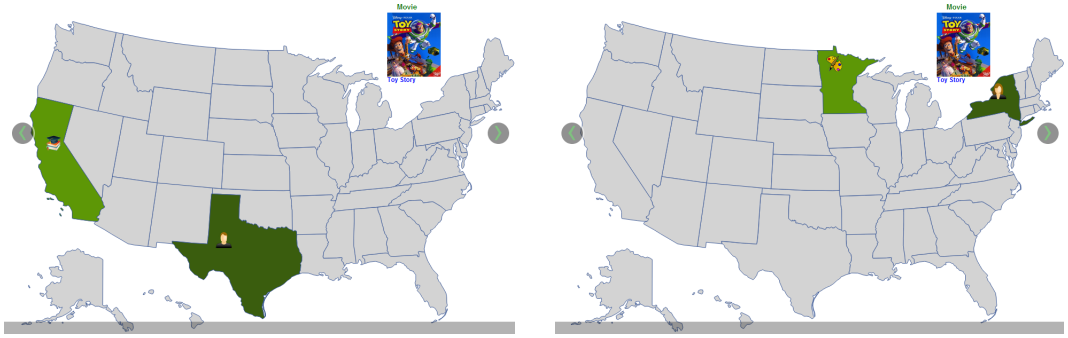

Figure 6.5. Geo-visualization of tags mining .

diverse) items tagged by similar (or diverse users). In addition, Settings also include *coverage* and *number of groups*. For example, as shown in the figure 6.4, suppose end user selects item attribute, *Toy Story* and apply settings such as *item:similarity*, *user:diversity*, and *tags: similarity* with *coverage* as *50%* and *Number of groups* as *2*. The conjunctive query for such input will be displayed input screen. Before user clicks on *show results* button, she can add more or remove user/item attributes and change settings if required. On finalizing the query user can click on the *show results* button for the results.

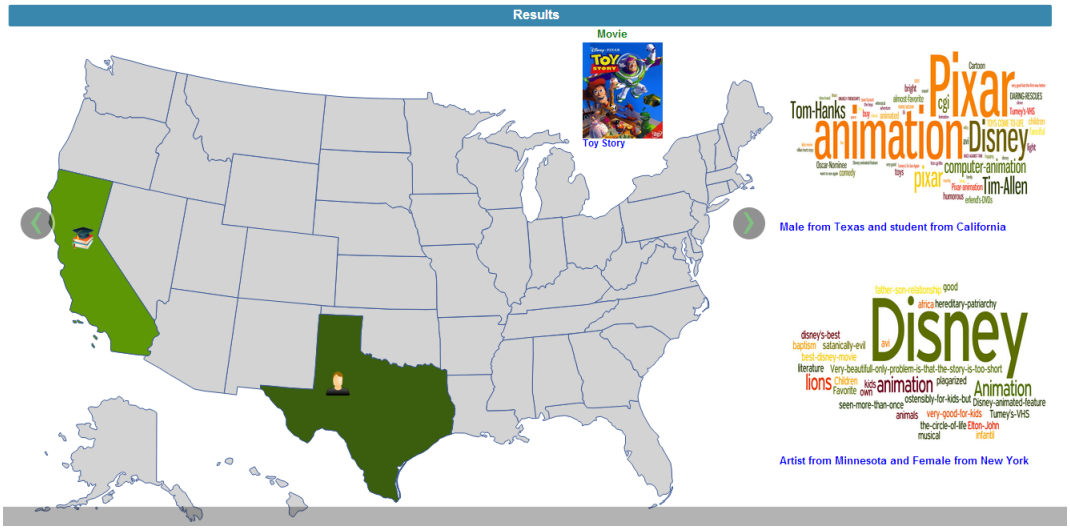

Figure 6.6. Tag Clouds for Tags Mining. .

Result of such input query will be displayed on geographical map and tag clouds side by side. Output screen will be divided into two parts. Left hand side of the screen contain a carousel that contains geographical map. User components that describe user attributes will be rendered as icons on the geographical map. The number of items in carousel will be equal to number of groups, k specified in input query. Right hand side of the output contain tag clouds. The number of tag clouds is also same as the number of user groups. Each tag cloud represents, set of tags given to an item by users from the result set.

Suppose, a user wants two user groups who tagged similarly on movie *Toy Story*, output of such input query will be two diverse user groups, *(Male from Texas and student from California)* and *(Artist from Minnesota and Female from New York)* who tag similarly on movie *Toy Story* as shown in Figure. 6.5. Both the items in carousel are expanded for the clarity purpose. Attributes of user components such as *gender, location, and occupation* will be rendered over geographical map. The tag clouds on the right hand side represents tags by each of the group on item *Toy Story*.

As user selected similar tags, it can be observed that most of the tags are similar on both tag clouds (such as $disney, animationetc$)as shown in Figure 6.6. This helps end user to interpret the tagging behaviour over set of items and decide upon the item of interest.

CHAPTER 7

CONCLUSION

In this thesis we built system that can interpret the rating and tagging behaviours on social web. First, we described our rating interpretation system which takes item or set of items as input and on dual mining, returns user groups as cuboids. These cuboids are rendered over geographical map to allow end users to visualize ratings. We also provide option for end users to explore ratings by providing aggregate statistics of user sub-population rating. Second, we introduce tags interpretation system which accepts attributes from user/item dimension and on dual mining, returns set of results that contain user and/or item dimensions, and also tag component. Every Item/User dimensions are visualized on geographical map and set of tags are displayed as tag clouds. This allows user to compare tag clouds and make informed decision about interested items.

REFERENCES

[1] M. Das, S. Amer-Yahia, G. Das, and C. Yu, "Mri: Meaningful interpretations of collaborative ratings," *PVLDB*, vol. 4, no. 11, pp. 1063–1074, 2011.

[2] M. Das, S. Thirumuruganathan, S. Amer-Yahia, G. Das, and C. Yu, "Who tags what? an analysis framework," *PVLDB*, vol. 5, no. 11, pp. 1567–1578, 2012.

[3] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh, "Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals," *CoRR*, vol. abs/cs/0701155, 2007.

[4] S. Thirumuruganathan, M. Das, S. Desai, S. Amer-Yahia, G. Das, and C. Yu, "Maprat: Meaningful explanation, interactive exploration and geo-visualization of collaborative ratings," *PVLDB*, vol. 5, no. 12, pp. 1986–1989, 2012.

[5] S. Amer-Yahia, J. Huang, and C. Yu, "Building community-centric information exploration applications on social content sites," in *SIGMOD Conference*, 2009, pp. 947–952.

[6] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *STOC*, 1998, pp. 604–613.

[7] M. Charikar, "Similarity estimation techniques from rounding algorithms," in *STOC*, 2002, pp. 380–388.

BIOGRAPHICAL STATEMENT

Shrikant Desai received his bachelors in computer science and engineering from Visvesvaraya Technological University, India, in 2006. He began his graduate career in University of Texas at Arlington from Fall 2010. He received his masters degree in computer science and engineering in Fall 2012. His research interests include database exploration and data analytics.