Responsive Matrix Layout Visual Recommender

Rajesh Surana
Arizona State University
Tempe,
Arizona
+1 (480) 289 - 8922
rajesh.surana@asu.edu

ABSTRACT

In this paper, we present first of its kind - personalized responsive visual recommender in the matrix form with user-controllable interface. Major research in recommendation system has been focused on algorithm logic providing more accurate and personalized recommendations. Several studies have established that user satisfaction can be dramatically increased with the help of interactive visualization endowing capability to change the system content catering individual's need. Our approach enables users to compare recommended results for different categories as well as public interest vs personal affinity for topics. We have also integrated filter in our model to customize and remove unwanted results contributing towards user controllability. As a part of personalized recommender we also provided hide feature to store preferences of the users. To induce user's trust in our system we tried to come up reasonably transparent system logic. This recommender model tries to reach maximum audiences with its responsiveness thereby adding value towards usefulness of an associated website.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval {relevance feedback; H.5.2 [Information Interfaces and Presentation]: User Interfaces {graphical user interfaces (GUI), user-centered design

General Terms

Algorithms, Measurement, Design, Experimentation, Human Factors.

Keywords

Visual Recommender, stackoverflow, Personal recommender, Responsive Web Design (RWD)

1. INTRODUCTION

The emergence of e-commerce and rapid growth of World Wide Web has led to the flourishing development of recommendation system. Also, there is a growing trend in the online discussion forums. This has led to popularity of many Question & Answer (Q&A) sites like StackOverflow (a part of StackExchange Network), IndiaBix, Ask.com, BlikBook, Brainly, Experts-Exchange, Spring.me and so on along with Yahoo! Answers forum, Wiki Answers and Google Questions and Answers. With the onset of these websites, comes the challenge how accurately and quickly can these sites cater to the needs of the users. Is it possible in a way to predict the needs of a specific user based on his previous search or is it possible to filter out similar results based on users search for a particular topic. This was the driving force for the growing need of recommender systems for Q&A sites. Till now there are myriad number of recommender systems built-in along with development of different types of algorithm starting from content based, collaborative to hybrid recommendation. Keeping aside the different implementation aspects, recent studies have found that along with algorithms, visualization is a clinching factor in recommendation system. Very little research is dedicated to that explores the human computer interaction aspect and more specifically the user controllability and ease of access to recommender system. Keeping this idea in mind we explored how visualization plays an important role in recommendation and developed a visual recommender prototype. Furthermore we narrowed down our research to the development of a device adaptive application that is visually responsive and easy to understand.

While researching for the project we found that in USA 25% of the people use mobile as the only device to surf the internet. Hence, with the notion of developing a visual recommender system prototype we thought it would be more useful for the users if we build a mobile adaptive prototype that can work in desktop, laptop and mobile devices. We developed a novel system Responsive Matrix Layout Visual Recommender that is robust in nature and is a unique model to visually recommend for Q&A websites. Our model is displayed in the form of a matrix grid layout where user has the capability to get a comparative display of both personalized and public recommendation. Another important research question that we tried to deal with this model is how to convey the order or ranking of the recommended data without using numbers or sorting by numeric values. We used the Stack Overflow Unbounded dataset for implementing this model. We categorized the recommendation into public and private categories by maintaining user profiles and divided these categories into more specific genres of recommendation by using attributes like popularity, trusted, verbatim etc. The recommendation scores were calculated and normalized based on the required tuples from the dataset and subsequently the results were ranked and visualized using heat map.

Rest of the paper is organized as follows: Firstly we discuss our motivation behind this research project. In section 3, we take an overview of state of the art techniques who tried to implement similar prototype. Further in section 4, we elaborate our implementation details about data processing and knowledge representation. In the subsequent section, we cover visualization aspect of our prototype. Next, we explain our designing methodology behind this model. In section 7, we propose evaluation plan to analyze results obtained from our model. Finally, we conclude our work with endnote on future work.

2. MOTIVATION

Today's websites are inundated with myriad varieties of information with large number of sub-categories to cater viewer's ever-growing need. To enhance viewer's information search experience, recommendation system comes into picture which can suggest some selected topics based on the heuristic that two topics are similar in genres. Moreover, to improve precision in recommended topics it is necessary to customize them according to individual's interest in those topics. That's why we decided to come up with hybrid recommender which would generate recommendations for a particular user based on content based filtering with equal influence of personal tastes for each of those topics.

On another frontier, power of such recommender system seems to be limited with respect to the user experience on different screen size interfaces such as desktop computer, mobiles, phablets or smart-TVs, etc. With exponential increase in the selling of small screen devices it had become mandatory to design responsive website which can readjust itself irrespective of any screen size. Recent report by comScore [1] states that U.S. users consume 60% of digital media using mobile devices. This fact alone is a big motivation to go for responsive recommender.

Conventionally, recommender systems have rank form where recommendation results are ranked according to ranks and displayed in the form of list. Well known websites like LinkedIn, Amazon, EBay and Facebook use this kind of recommendation system. Even though these recommender systems help viewer to smoothly navigate through forest of information, as the information in recommender itself increases, this overload defeats recommender's original motif. This overload of information type may constrain the way items are recommended. Better way to mitigate this problem is to take advantage of human brain's ability to process and grasp complicated data easily when represented in the form of good visualization [3]. Our brain is good at edge detection, shape recognition, and pattern matching which favors visual information over mere texts or numbers. Hence, we decided to go for matrix form visual recommender which will provide comparisons of recommended products based on different attributes like votes for some question or popularity of it based on number of answers to it. We also used heat map to represent respective ranking of each result with respect to others to help user grasping relative ranks of each of those.

Study of Denis et al. [2] have emanated very interesting aspect of data visualization that Computer Interaction plays very important role in user satisfiability over recommender system. Hence, we decided to provide a visual user-controllable interface for our hybrid recommender.

3. RELATED WORK

While building the idea for the project we studied and went through works of different ongoing and previous researches on recommendation system and found out Real Time Web was the trending topic which is thought by many an important resource of recommendation data [5]. McCarthy et al. [5] have been working on Twitter Data to develop their content based news recommendation. They built a prototype in which they extracted data from user profiles based on their Twitter posting and used this profile information to recommend products and services for these users. Apart from that we also went through the works of Shuguan Han [6] , who has focused his recommendation system on mobile device with multi touch interactivity. His research focused on the differences in exploration pattern of web search in desktop and mobile devices.

Experimental results from conferences suggests that search pattern has a clear goal when surfing for websites like stackoverflow, Yahoo Answers, Ask.com but while reading Twitter , Facebook and other social networking sites and news websites they lacked a clear well defined goal. Furthermore, researches highlights that the mobile device applications differs from traditional desktop application in terms of click patterns, query lengths, search time distance, information needs, visual interactivity where mobile devices have a lot more interactions with system, touch, tap and zoom in and out.

Bostandjiev et al. [14] have also worked on similar taste of visual recommender. They focused on modeling recommendation system for music by leveraging social data available for a user on websites like Facebook and Twitter. Along with visualization they also focused on semantic analysis of information available on internet.

Having enlightened after going through these researches we decided to build a prototype that is responsive in both desktop and mobile devices. We extracted data from the stackoverflow API and visualized the recommended data based on the attributes like popular, trusted, rated and verbatim. We calculated the recommended scores of each of them based on the required tuples from the dataset and also from the user profile information that we maintained in our system by mining the profile information from the stackoverflow API. We maintained a separate preference table in our database where we can keep a check of the particular posts that a user either liked or disliked based on his own interactivity with the model.

4. IMPLEMENTATION

In this section we discuss data collection, preprocessing and analysis of it.

4.1 Data Collection

We used stackoverflow API to extract data for our research project. Following are the fields from our extracted data. Each field is described by stackoverflow [7] as below-

4.1.1 type - Stackoverflow has three types of data viz., question, answer and accepted-answer. One answer is marked as accepted when user who has asked the question chose it as correct answer. Rest of the answers are categorized as mere answer.

4.1.2 title - It is the heading of the question. It gives quick idea about what the question and answers are talking about.

4.1.3 user_id - It is the primary key which uniquely represent each user on stackoverflow website. With contrast to user_name which is not unique, user_id is unique for each user.

4.1.4 vote - It represents the number of up votes or down votes given by a user to any question or respective answers [15]. One user can give at the most one vote for any question or answer. Positive number (up votes) represent highly anticipated content by stackoverflow community.

4.1.5 reputation - It conveys how much the community trusts you [16]. The primary way to gain reputation is by posting good questions and useful answers. Votes on these posts cause you to gain (or sometimes lose) reputation.

4.1.6 tag - A tag is a keyword or label that categorizes your question with other, similar questions [17]. Using the right tags makes it easier for others to find and answer your question.

4.2 Data Processing and Knowledge Representation

In this section, we describe how we preprocessed the data, converted it to appropriate knowledge representation. Throughout the report we will call each entry such as question or answer as 'tuple'.

4.2.1 Data pruning - In this stage, we removed the entries having null title field. Next, we deleted tuples having user_id as 0. This prevented from unwanted results and recommendation process breakdown.

4.2.2 Processing - Our model recommends top 10 questions with respect to different criteria using full-text index feature inbuilt in MySQL database. We will see more about full-text index feature in subsequent subsections. We will call these criteria as recommendation 'attributes'. Following is the list of these 4 attributes that we are using in our recommendation matrix -

4.2.2.1 Trusted - For this we consider user's reputation who has asked the question. Higher the reputation of user higher the Trusted score for that question.

$$Trusted\ score = \frac{(reputation_i - minreputation)*10}{(\text{maxreputation} - \text{min}reputaton)}$$

Where $minreputation$ = minimum reputation among 10 questions
$maxreputation$ = maximum reputation among 10 questions
$reputation_i$ = reputation of a current question's asker

Each formula normalizes the trusted score in the range 1 - 10.

4.2.2.2 Rated - Higher the votes to the question higher is the Rated score.

$$Rated\ score = \frac{(vote_i - minvote)*10}{(\text{maxrvote} - \text{min}vote)}$$

where $minvote$ = minimum vote score among 10 questions
$maxvote$ = maximum vote score among 10 questions
$vote_i$ = vote obtained by current question

4.2.2.3 Verbatim - Here, we take into consideration maximal string match of the question's title and tags field with searched query. Top recommended results will have almost the same text as that of searched query.

$$Verbatim\ score = \frac{(verbatim_i - minverbatim)*10}{(\text{maxverbatim} - \text{min}verbatim)}$$

where $minverbatim$ = minimum match score among 10 questions with respect to query
$maxverbatim$ = maximum match score among 10 questions with respect to query
$verbatim_i$ = match score for current question

For text matching we used similar_text function() of PHP which implements World's Best Algorithms by Oliver [9].

4.2.2.4 Popular - If number of answers to a particular questions are high that means that topic is highly popular among the community. We calculated number of answers to each question and added new field in our data as 'ans_count'.

$$Popular\ score = \frac{(anscount_i - minanscount)*10}{(\text{maxanscount} - \text{min}anscount)}$$

where $minanscount$ = minimum ans_count among 10 questions
$maxanscount$ = maximum ans_count among 10 questions
$anscount_i$= ans_count of current question

All these four formulas are used to calculate recommendation score in public column. For personal column we compute average of score given by above formula and personal domain interest score.

4.2.3 Database selection and knowledge representation

While recommending questions, principal challenge was to find most relevant top 10 questions. We needed some tool to match searched query with our tuples' relevant fields. We choose 'title' and 'tags' for matching against searched query. Now, next difficulty was to implement some text matching algorithm which matches text at various layers such as letters, words and complete sentence. After extensive search, we found that MySQL relational database has such search functionality over full-text index. FULLTEXT indexes are created on text-based columns (CHAR, VARCHAR, or TEXT columns) to help speed up queries on data contained within those columns, omitting any words that are defined as stopwords [8]. Full-text searching is performed using MATCH() ... AGAINST MySQL query. This motivated us to choose MySQL for our data storage. Knowledge representation in MySQL

```
`id` int(11) NOT NULL AUTO_INCREMENT,
`type` varchar(15) CHARACTER SET utf8 NOT NULL,
`title` varchar(112) CHARACTER SET utf8 NOT NULL,
`text` varchar(1980) CHARACTER SET utf8 NOT NULL,
`user_id` int(7) NOT NULL,
`vote` int(1) NOT NULL,
`reputation` int(6) NOT NULL,
`tag` varchar(59) CHARACTER SET utf8 NOT NULL,
`user_tags` text CHARACTER SET utf8 NOT NULL,
`ans_count` int(3) NOT NULL,
PRIMARY KEY (`id`),
FULLTEXT KEY `title` (`title`,`tag`,`user_tags`),
FULLTEXT KEY `title_2` (`title`,`tag`),
FULLTEXT KEY `title_3` (`title`)
) ENGINE=MyISAM AUTO_INCREMENT=50001 DEFAULT CHARSET=latin1
```

4.3 Data Analysis

In order to provide our users with personalized recommendations, it was needed to come up with a strategy to calculate domain interests of each user. Most intuitive solution will be either asking user to provide domain interests while registering at the website or calculating based on user interaction history on website. Asking too many input from user while creating profile will negatively affect website's membership. Hence, we opted for second solution. For the same purpose we added one more field 'user_tags'. We used tags associated with each tuple which are similar to domain. 'user_tags' column contains all the tags corresponding to the questions asked and answered by that particular user. To compute user_tags for each user, we mined the distinct user_ids from our table and then for each user mined the unique tuples where she has asked the question or answered it. From all such tuples we took tag field and inserted the concatenated result into 'user_tags' field. If you look carefully many of the tags will be found repeated in this field and which is what we wanted. FULLTEXT index results in higher matching if certain word is found more number of times. For example, if 'android' word is found more than once in x user's 'user_tags' field then it means she has more interaction with android tag questions or answers and hence it can be safely concluded that she is more interested in 'android' questions. We rank results in personal column based on the average of recommendation score calculated by formula of particular attribute and personal domain interest score.

Again to store online feedback of the user we provided hide symbol in our model in front of each recommended question. Once user click on that symbol we store user_id and question id into the preferences table. Now, each time recommender engine search for most relevant top 10 results in database it excludes the tuples with ids stored in our preferences table.

5. VISUALIZATION DESIGN

With aim in mind to make our recommendation responsive, we had many options to develop responsive, mobile-first project on web. We end up choosing bootstrap [10] framework which is made for devices of all shapes, and projects of all sizes. We used container-fluid structure to dynamically adjust our content in website according to screen size. Bootstrap uses jQuery [19] framework as its foundation. For server-side processing we used PHP. PHP has functionality to create sessions which we used to remember logged in user as well as searched query while navigating through web pages. To transfer data from front-end to back-end and get back result, we used AJAX [11] technology. AJAX is the art of exchanging data with a server, and updating parts of a web page - without reloading the whole page.

5.1 Visualization Grid

Moving towards visualization elements here is our recommender at first look-

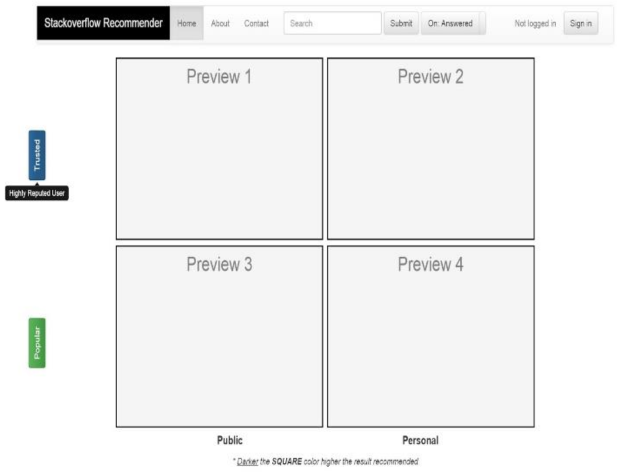
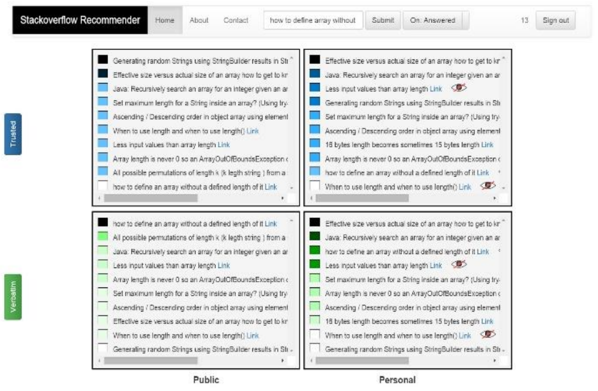


Figure 1. Recommender at first glance

We divided our recommendation space into grid of four boxes. First row corresponds to attributes 'Voted' and 'Rated'. Second row is for 'Popular' and 'Verbatim'. By default, we set our left side buttons to 'Trusted' and 'Popular' attributes. You just need to click on the buttons to change their type. We have also provided tooltips for each button to explain user the meaning of each attributes. Further, our grid is divided into two columns viz., Public and Personal. If user has not logged in, then we replicate the result of Public column into Personal column. Swearingen and et al. [12] have already emphasized on the importance of visual aspect of recommender system. We followed their advice while designing our prototype.

5.2 Recommendation box

Here is the screenshot of our model when user searched for 'how to define array without specifying length' –



Generating random Strings using StringBuilder results in Str
Effective size versus actual size of an array how to get to kr
Java: Recursively search an array for an integer given an ar
Set maximum length for a String inside an array? (Using try-
Ascending / Descending order in object array using element
When to use length and when to use length() Link
Less input values than array length Link
Array length is never 0 so an ArrayOutOfBoundsException c
All possible permutations of length k (k legth string) from a
how to define an array without a defined length of it Link

Effective size versus actual size of an array how to get to kr
Java: Recursively search an array for an integer given an ar
Less input values than array length Link
Generating random Strings using StringBuilder results in Str
Set maximum length for a String inside an array? (Using try-
Ascending / Descending order in object array using element
16 bytes length becomes sometimes 15 bytes length Link
Array length is never 0 so an ArrayOutOfBoundsException c
how to define an array without a defined length of it Link
When to use length and when to use length() Link

how to define an array without a defined length of it Link
All possible permutations of length k (k legth string) from a
Java: Recursively search an array for an integer given an ar
Less input values than array length Link
Array length is never 0 so an ArrayOutOfBoundsException c
Set maximum length for a String inside an array? (Using try-
Ascending / Descending order in object array using element
Effective size versus actual size of an array how to get to kr
When to use length and when to use length() Link
Generating random Strings using StringBuilder results in Str

Effective size versus actual size of an array how to get to kr
Java: Recursively search an array for an integer given an ar
how to define an array without a defined length of it Link
Less input values than array length Link
Set maximum length for a String inside an array? (Using try-
Array length is never 0 so an ArrayOutOfBoundsException c
Ascending / Descending order in object array using element
16 bytes length becomes sometimes 15 bytes length Link
When to use length and when to use length() Link
Generating random Strings using StringBuilder results in Str

Figure 2. Recommender with search results

To maintain consistent coloring we had blue color for complete first row and green color for second one. This consistency in color helps user understand that each box in a row has recommendation result with respect to one attribute. Whereas different colors in each column suggest that results belong to different attributes.

5.3 Heat map

Traditional ranking system would have been ineffective in our case if user wanted to take advantage of grid system to compare results simultaneously across different domains. That's why we came up with heat map. Each question is preceded by a box symbolizing recommendation score in that box. Darker the box higher the result recommended. Two or more squares with same shade of color depicts questions with comparable or similar recommendation score. We used SVG [18] for building squares.

5.4 Hide Symbol (Eye with red slash)

In order to hide the question which user does not want to see in future, we came up with this hide symbol which succinctly does portray desired meaning. It is the part of implementing personalized recommendation system with online feedback and update. We used SVG [18] for building hide symbol.

5.5 Filter Button: Answered

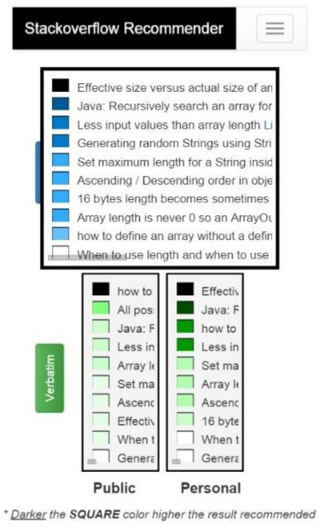
If user is merely interested in answer to the question then she can filter out those questions which do not have any answer yet. Our toggle button right after submit button instantly updates the recommended results according to the current value.

5.6 Navbar

Bootstrap has collapsible navbar which can detects screen size and collapses if browser width goes below 768px. It means small screen devices will have collapsed navbar where large screen devices will have full-fledged navbar. In the collapsed mode user has to click on toggle button to expand and see all the navbar menus and buttons. We included search and other navigation buttons in the navbar so that 90% of the space in any screen size will be occupied by recommendation grid. This was one of the tricky decision in order to maximize available space usage for recommendation results.

5.7 Mobile View

Below is a screenshot of our recommender when opened in iPhone 6 –



Effective size versus actual size of an
Java: Recursively search an array for
Less input values than array length Li
Generating random Strings using Stri
Set maximum length for a String insid
Ascending / Descending order in obje
16 bytes length becomes sometimes
Array length is never 0 so an ArrayOu
how to define an array without a defin
When to use length and when to use

Public	Personal
how to	Effectiv
All pos	Java: F
Java: F	how to
Less in	Less in
Array le	Set ma
Set ma	Array le
Ascenc	Ascenc
Effectiv	16 byte
When t	When t
Genera	Genera

* *Darker the **SQUARE** color higher the result recommended*

Figure 3. Recommender opened in iPhone 6

You can notice that first row box has consumed most of the width of the screen. This happens when user click on the box in mobile. Same effect we get in big screen devices on hovering the particular box. Again the reasoning behind it was to maximize space usage and focus on current box by showing maximum content without scrolling sideways. Here, we implemented Sneiderman's [13] details on demand strategy - showing details only when required.

6. Methodology

Based on the previous studies that we did, the primary research question we had in mind was whether we can build a prototype that can make a recommender system that works well in any kind of screen-sized devices. This led us to the development of a responsive visual recommender that is not only mobile adaptive but also adaptive to desktops and laptops. Secondly, we implemented a single flexible window to display both personalized and generalized recommendation at the same time. This also answers our research question where we thrived to achieve a quick comparative analysis between two modes of recommendation. Most Recommendation systems generally rank recommended results in a numeric manner. For this we tried to focus on some other aspect to efficiently convey the message through visualization. Hence, we used heat map to depict results in the window. And having a high percentage of users using the Q&A websites on their mobile devices this responsive layout can help them in their search query much more quickly and precisely while they are on the go. Although our primary focus area of research was on visual recommender, we also focused on the analytics part such as finding the particular genres of preference of search for a particular user.

A typical use case that can be generated from our model is that an instructor can set questions in a much better way by following the top trending questions and answers in that particular genre. He can understand the areas where the students were finding it difficult to solve and can help them by focusing more on those areas.

7. Evaluation Plan

We propose the following plan to evaluate our visual recommendation model. A feedback can be asked to the user after she has interacted with the system for decent amount of time. Considering target audience to be stackoverflow user, it can be taken in the form of pop-up on the same page as that of recommender or asked through email registered for that user. It will be a good idea to take feedback from registered user as personal column will be effective only after logging in. So, feedback provided by unregistered user will be of no/less use. Feedback will consist of below questions to evaluate effectiveness of the system-

Objective questions with 1 to 5 rating (5 being best):

1. When you searched for 'xyz', were the recommended results relevant?
2. Was the ranking shown in the form of heat-map helpful in comparing different recommendations?
3. Was the result in matrix form easily comprehensible?
4. Was the matrix form helped you better understand which questions will be highly relevant for you?
5. If you accessed our recommender on small screen devices (smartphones, tablets, ipad), then how was your experience? How much you are satisfied with user interface? (5 being the most satisfied.)
6. How easy it was to understand overall design of the system?

Above questions will help us understand how effective the overall design is.

Subjective questions:

1. What kind of attributes you would like to be added to our recommender in addition to trusted, voted, etc.
2. Do you have any additional comments in order to improve our system in terms of visualization?

This feedback will help us understand what other important features our viewers are missing and any other unaddressed issues with respect to our system.

Checkbox questions:

1. What devices you would prefer to access our recommender in general?

a) Smartphone
b) Desktop
c) Tablet
d) Smart-TV
e) Others

2. Will you recommend our recommender to others?

Yes / No

8. Discussions and Future Work

We have described the design and implementation of the recommendation system so far. However, we acknowledge that there is a huge scope for improving our model. There can be addition of more attributes thereby providing more user control and filtering. Also, by combining the recommendation score of all the attributes we can provide the best of all recommendation feature to our visualization window. Significant amount of study will be needed to come with the formula for the same. Furthermore, even though system logic of our implementation is transparent, there is a scope left to enhance it by adding more of visual analytics into our system.

9. Conclusion

Throughout our project we have adhered to the visualization information seeking mantra - overview first, filter and zoom, details on demand. We have used the Stack Overflow Unbounded Dataset and tested our model in different sized data and found out that system scales pretty well irrespective of size. We had the above mentioned research questions in mind while building this novel Recommender System and so far achieved our domain specific objectives. However, we would like to explore if we can orient this model into some other application domains and measure the performance accordingly. That seems to be further research in this scope and we look forward to improvise our model so that it can cater to a larger audience rather than only those percentage of people who deal with technical questions and answers.

10. Acknowledgement

We would like to take this opportunity and thank Dr. Sharon Hsiao for her valuable guidance and constructive feedback thereby helping us to implement our idea effectively. This research project was part of the course CSE 591: Data Visualization in Arizona State University.

11. References

[1]Sarah Perez. (Aug 2014). Majority Of Digital Media Consumption Now Takes Place In Mobile Apps.

http://techcrunch.com/2014/08/21/majority-of-digital-media-consumption-now-takes-place-in-mobile-apps/

[2] Parra, D., Brusilovsky, P., & Trattner, C. (2014). See what you want to see: visual user-driven approach for hybrid recommendation. Paper presented at the Proceedings of the 19th international conference on Intelligent User Interfaces, Haifa, Israel.

[3]Noah Iliinsky. (Feb 2012). Why Is Data Visualization So Hot?

http://blog.visual.ly/why-is-data-visualization-so-hot/

[4]Stackexchange API

https://api.stackexchange.com/

[5] Owen Phelan, Kevin McCarthy, Mike Bennett, and Barry Smyth. (2011). Terms of a Feather: Content-Based News Recommendation and Discovery Using Twitter , Advances in Information Retrieval Lecture Notes in Computer Science Vol 6611, pp 448-459

[6] Han, S., Hsiao, I. and Parra, D. (2014). A Study of Mobile Information Exploration with Multi-touch Interactions, In SBP. 269-276, Springer.

[7] Stackoverflow Help documentation

http://stackoverflow.com/help

[8]MySQL FULLTEXT index documentation https://dev.mysql.com/doc/refman/5.6/en/innodb-fulltext-index.html

[9] PHP similar_text() function documentation

http://php.net/manual/en/function.similar-text.php

[10] Bootstrap documentation

http://getbootstrap.com/

[11]AJAX documentation

http://www.w3schools.com/ajax/

[12]Swearingen, K., & Sinha, R. (2001, September). Beyond algorithms: An HCI perspective on recommender systems. In ACM SIGIR 2001 Workshop on Recommender Systems, (Vol. 13, No. 5-6, pp. 1-11).

[13]Ben Shneiderman. (1996), The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations.

[14]Bostandjiev, S., O'Donovan, J., and Höllerer, T. (2012). TasteWeights: a visual interactive hybrid recommender system. In Proceedings of the sixth ACM conference on Recommender systems (RecSys '12). ACM, New York, NY, USA, 35-42.

[15] Stackoverflow votes documentation

http://stackoverflow.com/help/why-vote

[16] Stackoverflow reputation documentation

http://stackoverflow.com/help/whats-reputation

[17] Stackoverflow Tags documentation

http://stackoverflow.com/help/tagging

[18] SVG documentation

http://www.w3.org/Graphics/SVG/

[19] jQuery documentation

https://jquery.com/

[20] Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1), 5-53