A Dissertation
On

" Expert Recommender System "

Submitted in partial fulfillment of the requirement
for the award of degree of

MASTER OF TECHNOLOGY
Information Systems
Delhi Technological University, Delhi

SUBMITTED BY

INDU BALA
2K15/ISY/09

Under the Guidance of

DR. AKSHI KUMAR

Assistant Professor
Department of Computer Engineering
Delhi Technological University

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
2017

Certificate

This is to certify that the dissertation entitled **"Expert Recommender System Using trust metric and Collaborative Filtering "** has been submitted by **Indu Bala (Roll Number: 2K15/ISY/09),** in partial fulfillment of the requirements for the award of Master of Technology degree in Information Systems at **DELHI TECHNOLOGICAL UNIVERSITY**. This work is carried out by her under my supervision and has not been submitted earlier for the award of any degree or diploma in any university to the best of my knowledge.

Certificate

This is to certify that the dissertation entitled **"Expert Recommender System Using trust metric and Collaborative Filtering "** has been submitted by **Indu Bala (Roll Number: 2K15/ISY/09),** in partial fulfillment of the requirements for the award of Master of Technology degree in Information Systems at **DELHI TECHNOLOGICAL UNIVERSITY**. This work is carried out by her under my supervision and has not been submitted earlier for the award of any degree or diploma in any university to the best of my knowledge.

Akshi Kumar 23/6/17

(DR. AKSHI KUMAR)
Project Guide
Assistant Professor
Department of Computer Engineering
Delhi Technological University

2

Scanned by CamScanner

(DR. AKSHI KUMAR)

Project Guide

Assistant Professor

Department of Computer Engineering

Delhi Technological University

Acknowledgement

I am very thankful to **DR. AKSHI KUMAR** (Assistant Professor, Deptt. of Computer Engineering) for providing immense support and guidance throughout the project.

I would also like to express gratitude to Mrs. Arunima (Research Scholar, Delhi Technological University) for providing me continuous support and guidance during this project.

I would also like to express my gratitude to the university for providing us with the laboratories, infrastructure, testing facilities and environment which allowed us to work without any obstructions.

I would also like to appreciate the support provided by our lab assistants, seniors and our peer group who aided us with all the knowledge they had regarding various topics.

INDU BALA

(2K15/ISY/09)

Abstract

In online question & answer communities, People seek for expert opinions on their problems and share their expertise in a particular field. Hence expert finders are the building blocks of online Q&A communities and finding expert in such communities is one of the biggest challenges and an highly active research problem. Expert finding is the process of finding expertise level of each user and identifying erudite people on a given topic. Usually experts are find out in two ways - social network analysis or concept map. The recommendation system we have proposed is called **Expert Recommendation system** which incorporates two important features of recommender systems that are trust and similarity achieved by using a well known global trust metric Page rank and collaborative filtering respectively.

For collaboration of experts, Firstly it is necessary to find experts in the community using social network analysis by using a variation of pagerank which will determine the expertise of a particular user by determine the expertise of users whom he has helped and score of the posts, he has contributed to. Spearman's rho is applied to these two lists of top-k users(one extracted from the community and other calculated by applying pagerank) to determine the correlation between them. Once, we have top-k experts of a community, using collaborative filtering we will recommend these experts to each other. Stack Overflow is a classic example of online Q&A Community which has 14M questions, 22M answers, 58M comments, 49K tags and 73M users till date. It assigns Reputation points to each of its users. So, We have used stack overflow to test our framework.

List of Figures

List of Tables & Graphs

Contents

Chapter - 01

Introduction

1.1 Introduction

With the emergence of Internet, Any kind of Information which is accessible is available at an inexpensive price and the size of internet and reach of search engines are both increasing at a shift pace. Therefore, there is an imperative requirement of developing systems that not only retrieve information but relevant information for each user. This problem is solved by using information filtering tools that are widely known as recommender systems. The essence of recommender systems lies in the fact that it should precisely predict the needs and likes of every user[1] but the objectives of recommender system are dynamic in nature and keep changing according to users' needs. Sometimes, the opinion of similar users to a particular user are helpful and sometimes user does not trust a complete stranger and wants to know from a trusted source about his choices and in some scenarios, user will trust the opinion of an expert in the respective field.

In online question & answer communities, People seek for expert opinions on their problems and share their expertise in a particular field. Hence expert finders are the building blocks of online Q&A communities and finding expert in such communities is one of the biggest challenges and an highly active research problem. Online communities are a great source of information but it is necessary to determine the authenticity of this information to make the knowledge shared on these platforms reliable.

Expert finding is the process of finding expertise level of each user and identifying erudite people on a given topic. Usually experts are find out in two ways - social network analysis or concept map.

- Social network analysis involves determining relationships between users by creating a network of users in which users are the nodes of the graph and relationship between the users acts as the links between the nodes. PageRank, a well known global trust metrics is widely used in social network analysis which determines the prestige of a user using the prestige of all the other users associated with this user.

- Concept map works on content based recommender systems as it uses the content of the post to identify the concept of the answer and compare it with the concept in given question.

Expert finding has many applications such as availability of relevant questions to an expert so that he can respond timely and does not waste his time on simple questions and summarization of information so that each user is provided with information related to topic he is interested in.

One of the applications of expert finding is collaboration of experts that is by using collaborative filtering finding experts with same expertise level and interest in same topics. It is a two-fold process which involves identifying experts in an online community such as stack overflow and then using collaborative filtering, recommending posts of interest to one expert on the basis of preference of posts of the similar experts.

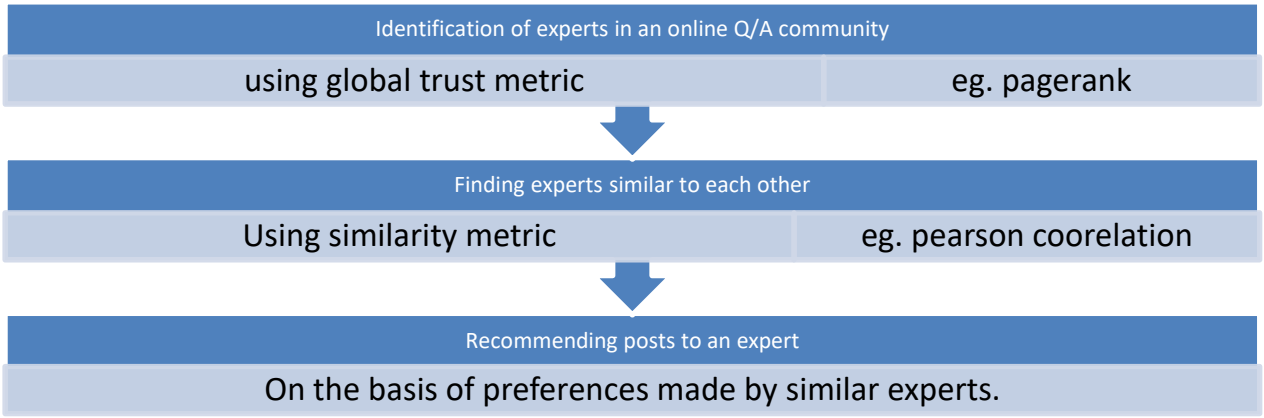


Fig. 1.1 Collaboration of Experts

Expert Recommender system

The recommendation system we have proposed is called **Expert Recommendation system** which incorporates two important features of recommender systems that are trust and similarity achieved by using a well known global trust metric Page rank and collaborative filtering respectively. Online Q&A communities are perfect example of platforms where people participate to seek expertise on their interested topics. People do not look for personal advices but expert views on such platforms therefore, expert finding is an integral part of these communities. In order to trust someone's opinion who is not known in person by the users of the community, it is necessary to state credibility of such experts by setting some

parameters for eg. Stack overflow(a well known online Q&A community) assigns reputation score to each of its users which determines the prestige of all the users in the community.

For collaboration of experts, Firstly it is necessary to find experts in the community using social network analysis by using a variation of pagerank which will determine the expertise of a particular user by determine the expertise of users whom he has helped and score of the posts, he has contributed to. Details of Top-k users of the community are extracted from the given API of the community and page rank is applied on each of these users to rank them accordingly then spearman's rho is applied to these two lists of top-k users(one extracted from the community and other calculated by applying pagerank) to determine the correlation between them. Higher the spearman's rho , higher will be the correlation which shows the effectiveness of pagerank in determining expertise.

Now that we have top-k experts of a community, using collaborative filtering we will recommend these experts to each other. It means that for each expert in the list, top-z neighbors (based on similarity between them) is determined. Similarity is based upon the scores they have earned on top-j tags of the community.

Using top-z neighbors list of each user , they will be recommended with posts on which their neighbors have contributed recently.

The utility of this system lies in the fact that these communities always face the problem of information abundance and to get right person indulged in right threads of questions and answers is one of the biggest challenges till date but this system makes sure that an expert will be provided with relevant questions of his fields of interest and due to contribution of people of same level of expertise and same kind of interests, the threads will be information rich which is beneficial for all the users of the community.

1.2 Motivation

Search Engines are widely used for information retrieval and query processing but with search engines, in order to get relevant results corresponding to a query, the query is required to be very specific and appropriate keywords have to be selected. It is quite possible that the search outcomes are not arranged in a sequence, so the user has to scroll to all the pages to find relevant link to his problem. It often happens that the query is misinterpreted due to use of wrong keywords in the query posted.

Online Q&A communities (eg. Java forum, Stack overflow, Quora etc) emerged as an elegant solution to these shortcomings of search engines. On these social platforms, people can use

natural language instead of keywords to ask questions[23] but these platforms are open to wrong use and lack in the mechanisms to state the authenticity of information provided. There is no automatic way to determine the expertise level of answerers. Therefore it is required to include a mechanism that will determine the expertise level of each user and identify the experts in the community .

There are many challenges in successful implementation of these communities such as

- In online communities, the expertise level of the users is not clear and therefore whenever a n user posts any answer or comment, it cannot be determined automatically whether that information can be trusted or not.
- Abundance of Information is another important challenge that is due to overload of information, there is no criteria to make right information available to right crowd. It may happen that a person with high level of expertise in certain area is unable to see all the important posts in that area. So when an user posts a query it might take a large amount of time to get accurate and relevant response.
- For every query, a large number of people respond whose level of expertise in the given field is not known, hence all the true and false answers are accepted without any restrictions, hence creating confusion for the users that which answer should be trusted and which should not be.

All of these problems are addressed by the expert finding mechanisms used in these communities. By determining expertise levels of users, authenticity of their posts can be easily determined. By identifying experts, each expert will be shown relevant posts to indulge in so that he can use his adequate knowledge in given field and does not waste his time on simple queries. For users, it is easy to find reliable answers, once they get to know the expertise level of the answerers.

Collaboration of Experts

One of the most interesting and effective application of expert finding is collaboration of experts which has not been discovered in this research area. If traditional collaborative filtering is applied to experts and a matrix is created with experts and their scores in various subjects then users with same scores in a given subject will be considered similar to each other.

Users/tags	C	C++	Java	SQL
Jon	5	-	4	5
Ani	-	5	-	-
Bob	4	-	-	4
Siya	-	4	1	-

Table 1.1 a matrix user × tags

Here in this matrix, these are the top users in an online community and tags that are most popular among the user of this community. To find similar users, tag score of each user is compared with the tag scores of other users in the community. For example , tag score of Jon in C is 5 and that of Bob is 4, Intuitively these two can be said experts in C whereas tag score of Jon in java is 4 and that of Siya is 1, so it can be deduced that Jon is an expert in java but siya is not.

Using similarity metric such as pearson's correlation, it is possible to find top-k similar experts to an given expert and preferences of posts of these top-k neighbor experts will be recommended to a particular expert.

In this way expert will be indulged in most relevant posts and also will be challenged by same level of experts and for all the other users, if large number of experts will comment on a particular post, it will make that post more information rich, hence benefitting all the users of the community.

This study focuses on making the online Q&A communities more efficient for the expert users and making the overall community information rich for all the users of the community.

1.3 Scope

It is an cumbersome task to find reliable information in online communities where the identity of an user is not disclosed and users know each other through the knowledge shared by them. To decide which user is an expert in a given topic, It requires a deep knowledge of the topic itself. If an user with adequate knowledge over some topic joins an online community, It is necessary that he should be indulged in discussion where his knowledge can

benefit a large number of people and he also should increment his knowledge in the process of sharing it.

This requires a system which can personalize the online communities for the experts so that everybody can take advantage of his knowledge to the full extent. The proposed framework in this study called as Expert recommender system will make efficient use of the two requirements of the recommender system and these are:

- **Trust :** Although trust has various aspects and it is not easy to determine the definition of trust but due to arise of social web, there is ample amount of information about a person available on internet that can tell the relationship of that person with other people and also trust he/she has in other people. With this more refined information about someone, way to provide much more confident predictions has been found as a person tends to believe in information coming from a trusted source rather than believing in suggestions coming from a stranger.

  - In online communities people are not looking for personalized opinion but expert advice therefore, global trust metrics eg. pagerank are used to find experts in such communities. Global trust metrics decides the reputation of a particular user in a given community. It does not take the subjective views of each user but average the views of all users on a global level[8] , hence determining the trustworthiness of a user from viewpoint of a group of users as a community.

- **Similarity :** If two people are similar, it is highly probable that they tend to like same kind of things. As one of the most successful approaches to building recommender systems, collaborative filtering (CF) uses the known preferences of a group of users to make recommendations or predictions of those unknown preferences for other users[3]. It firstly finds out the similar users to a particular user 'A' and then recommends the items liked by those users to 'A'. For example if A & B rates a movie 4 out of 5 then intuitively their preference over movies should be same and if A rates a movie 4 out of 5 and C rates the same movie 1 out of 5 then intuitively their preference over movies should differ.

Expert recommender system uses these two properties of recommender systems very efficiently and find out the experts well known in the community and recommend their

preferences to each other. It provides a great application in expert finding which is one of the biggest challenge in online Q&A communities.

1.4 Goals of Project

General Goal : To design a framework for finding experts in an online community using social network analysis and then by using collaborative filtering , finding similar experts based on their level of expertise and their topics of interest to a particular user.

Specific Goals :

1. To explore the global trust metric 'Page Rank' in order to find out the experts in an online community.
2. To use collaborative filtering to find experts who are similar to a given expert based on their score in provided topics.
3. To create a hybrid recommender system which efficiently uses both properties i.e. similarity and trust by combining Page Rank (for global trust) and collaborative filtering(for similarity) algorithms .
4. Personalization of online communities for the experts so that their knowledge can be used upto its best extent.

1.5 Organization of Report

This report comprises of 6 chapters and 2 appendices.

Chapter 1 is introduction which will introduce to the underlying technologies used for the project and gives the motivation and scope of the project and also determines the general as well as specific goals of the research.

Chapter 2 will give an extensive Literature review required for the study which includes recommender systems, collaborative filtering, Trust aware recommender system, pagerank, structure of online communities and need of expert finding in online communities and related work in the field.

Chapter 3 describes the proposed framework along with the system architecture.

Chapter 4 describes all the details of the platform used to study the effectiveness of the framework.

Chapter 5 contains the results and analysis of the proposed system.

Chapter 6 has conclusion along with limitations of the system.

Appendix A contains the code snippets and appendix B contains the snapshots of the system.

Chapter- 02

Literature Review

2.1. Recommender Systems

With the advent of Web 2.0 also known as Social web, user is actively participating in contributing to the content of a particular platform that means that one can get to know enough about a particular user i.e. his taste in various domains such as clothing, movies, music, books, food, places etc so that he can be suggested with new genre of various things which he may like but this process of recommendations on basis of activities performed by a particular user often faces a barrier that is known as Information Overload.

This problem is solved by using information filtering tools that are widely known as recommender systems. The essence of recommender systems lies in the fact that it should precisely predict the needs and likes of all the users.

Any recommender system is based upon either the information/features provided about a particular object or similarity between two users or items[2]. Therefore Recommender systems are classified widely into two categories :

- Content based Recommender Systems : Content-based systems are based on features of the items recommended. For instance, if an Amazon user has ordered many electronics items, then recommend an item classified in the database as electronics item.[1]
- Collaborative filtering systems : Collaborative filtering finds out similarity between users and items based on their profiles. The items recommended to a user are those that are recommended by similar users.

  Collaborative Filtering is the most popular recommendation technique which works in two steps.

  1)First of all it finds out the most similar user to a particular user using a similarity metric. The similarity metric is an algorithm which find outs the similar users to user based on the items rated by him. One of the most widely used similarity metric is Pearson's correlation

  2) the second step is to predict the rating of a target item by the active user on the basis of ratings given by similar user to the target item or similar items to target item.

This process works quiet successfully for the users who have rated a good number of items as it is easy to find neighbors (similar users) of such user hence increasing the precision of a prediction but this method has several shortcomings such as

- it is not much of a help for new users as they have not rated much items, it is difficult to find out their neighbors because one of the intrinsic demand of similarity metric is mutual items to be rated.
- It cannot handle data sparsity too well and these methods work on probability so they do not know the confidence of their prediction.

Therefore similarity is not enough a factor to design an intelligent system that can handle the process of information filtering. To solve this problem Social web provides a prominent solution by providing reflection into user's personal interests whether it is people or places or things. This information provides by social platforms can be used to provide recommendations to users in various domains that is well known as social recommender systems but social recommender systems too fight with one of the intrinsic characteristics of human nature that is people tend to believe their known and trusted sources than unknown people on internet which are suggested to them as similar by an algorithm. Here comes the need of recommender systems to be more personalized and this problem is well tackled by involving factor of trust in recommender systems. User is much more satisfied by knowing that the recommendation is coming from a source that is trusted by him or someone whom he trusts.

The utility of social information of users in recommender systems has introduced a new class of recommender systems well known as social recommender systems. These systems uses the information provided by the users on various social networking platforms. It judges their lifestyle and their needs by their activities on social media.

This system works very well but it also faces some drawbacks due to intrinsic human behavior such as most people tends to rely on people they trust and this invites the need to include trust factor in recommender systems.

In [8] Massa and Avesani proposed a trust based recommender system i.e. users will explicitly give trust statements about other users which includes how much an active user finds other user's ratings trustworthy. The elicitation of these trust values forms a web of trust called as trust network and an algorithm called trust metric is used whose goal is to predict, the trustworthiness of an unknown user (user in which a certain user did not express a trust

statement) based upon the trust network by exploiting trust propagation over the trust network.

TaRSs also is a twofold process as collaborative filtering in which only the first step is different from CF in the sense that CF use similarity as a measure to find out neighbors of an active user while TaRSs uses trust as a measure. The second step is same as in both the processes the items liked by the neighbors from step 1, are then recommended to the active user.[8] called this technique, trust-aware recommender system.

In trust aware recommender systems, many well known techniques are used such as fuzzy logic , neural networks, graph theory, ant colonization and trust metrics. Using Trust metrics is most preferable and evolved technique in calculating trust between two unknown users in TaRSs although trust metrics have many other applications.

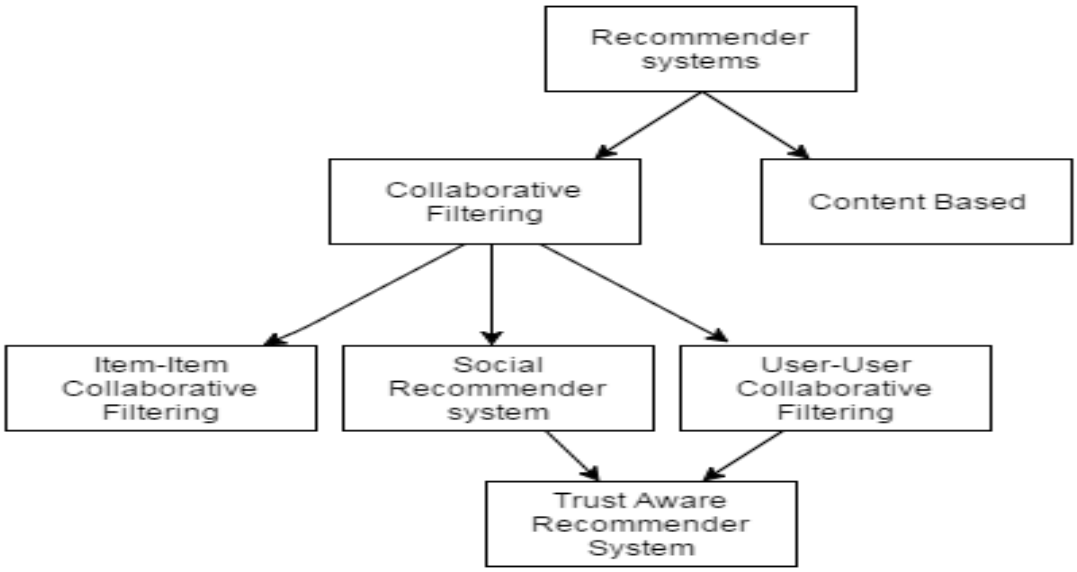


Fig. 2.1 Classification of Recommender Systems

2.1.1 Collaborative Filtering

As one of the most successful approaches to building recommender systems, collaborative filtering (CF) uses the known preferences of a group of users to make recommendations or predictions of those unknown preferences for other users[3]. It firstly finds out the similar users to a particular user 'A' and then recommends the items liked by those users to 'A'. For example if A & B rates a movie 4 out of 5 then intuitively their preference over movies should be same and if A rates a movie 4 out of 5 and C rates the same movie 1 out of 5 then intuitively their preference over movies should differ.

There are three main categories of Collaborative filtering techniques:

- Memory based CF : It takes a database with 'n' users and 'm' items where ratings are given for each item that a particular user has ever rated. On the basis of these ratings, the neighbors of a user are found(users who are similar to a particular user) and if a item is not rated by a user but it has been rated by his neighbors then this item is recommended to the user. This process comprises of two steps:

  1) Similarity Computation : Similarity between two users if calculated by using various methods such as Correlation-Based Similarity, Vector Cosine-Based Similarity, probability-based similarity. Mainly correlation based similarity is used in which similarity $w_{u,v}$ between two users u and v, or $w_{i,j}$ between two items i and j, is measured by computing the Pearson correlation or other correlation-based similarities[4]

  Pearson correlation between users u and v is

$$w_{u,v} = \frac{\sum_{i \in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i \in I}(r_{v,i} - \bar{r}_v)^2}},$$

  where I is the total set of items rated by both u and v. $r_{u,i}$ and $r_{v,i}$ are the rating provided for item i by user u and v respectively.

  2) Recommendations: Once top-k neighbors of a particular user 'A' are found, preferences of those users that are not rated by 'A' are recommended to 'A'. It is generally done by taking weighted sum of ratings of all the neighbors.

- Model based Collaborative Filtering: With the advent of Machine learning algorithms, it is easy to remember complex patterns analyzed in training data and then performing collaborative filtering tasks on test data. Model-based CF algorithms such as Bayesian models, clustering models, and dependency networks are commonly used to solve the short comings of memory based collaborative filtering.

  To provide an user with top-k recommendations, Associative rule mining is generally used.[3].

Generally in complex systems, Collaborative Filtering does not work alone and used along with other techniques. Hybrid CF systems combine CF with other recommendation techniques (typically with content-based systems) to make predictions or recommendations[3].

2.1.2 Trust Aware Recommendation Systems

Trust is one of the fundamental factor on which human beings behavior is based upon. If A trust B, then A will commit to something believing that B's suggestions or actions will not bring any bad outcome but Trust is multidimensional in nature so before developing an algorithm to apply trust factor in recommender systems, it is necessary to make clear all the dimensions of trust[10].

Trust is very subjective in nature, it depend on various factors that are also variant to specific domains for eg. "I trust my friend X for her choices in formal clothes but not in casual clothes." Trust can be explicit (direct trust statements given by one user to other users) or implicit (inferred from activities done between two users).

A general structure of TaRSs is as follow:

A trust matrix of user×user is created either by explicitly asking users to give all the known users a trust value or implicitly inferring trust between two users by analyzing their activities on internet. This matrix is passed as an input to an algorithm which will determine trust between two unknown users. Then on basis of these results the most trusted people of a particular user are determined and he provided with recommendations made by these people.

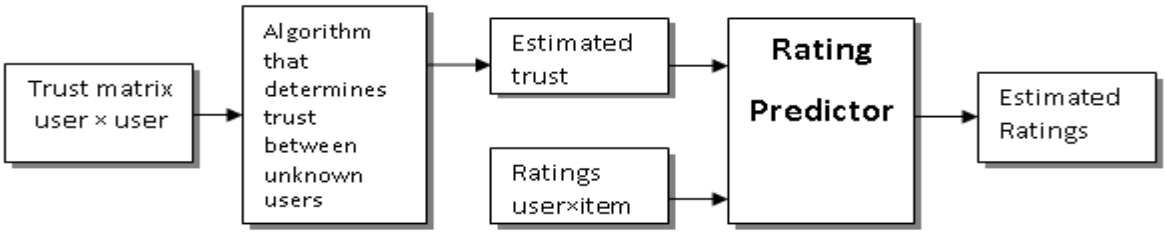


Fig. 2.2 General structure of TaRSs

Importance of trust in Recommender systems

People do not want to get recommendations from people who know everything , they want to get recommendations from people who know enough to meet their needs. This fact is the driving force behind applying trust in recommender systems. Traditional recommender systems are totally based upon the similarity factor between two users or two items and not at all utilizes the trust factor between two users[9]. Although trust has various aspects and it is not easy to determine the definition of trust but due to arise of social web, there is ample amount of information about a person available on internet that can tell the relationship of that person with other people and also trust he/she has in other people. With this more refined information about someone, way to provide much more confident predictions has been found

as a person tends to believe in information coming from a trusted source rather than believing in suggestions coming from a stranger.

There are levels of trust incorporated in recommender systems.

1) Profile level trust : This strategy suggests that trust should be considered as the reputation of a user i.e. his profile, his previous recommendations.

2) Topic level trust : This strategy suggests trust should be topical i.e. for different topics different users should be trusted. For e.g. may be A is trusted for his choices of movies but he may not be trusted with his choices of clothes.

Trust Metrics

In sociology, a trust metric is a measure of how much an individual can trust another (unknown to him) individual. Aggregation and propagation of trust are two building blocks of trust. In the context of recommender systems, a trust metric is a algorithm which determines the trust of a user for another user in the network whom he has not known or trusted explicitly. The strength of any trust based recommender system lies in the trust metric it uses. A good trust metric is supposed to exhibit some properties such as transitivity of trust(i.e. if a trust b and b trust c then a also trust c) , scalability(ability to handle any size of trust network),attack resistance(ability not to be overly influenced by agents who try to manipulate trust in bad faith) etc[10].

Trust metrics can be classified depending on various aspects.

1) On the basis of value of Trust- Trust values may be binary i.e. either a person is trusted or distrusted or there may be levels of trust i.e. trust values can be a part of a fuzzy set.

2) On the basis of medium of getting trust values - Trust values may be explicitly stated by users or trust can be implied by the various activities of users.

3) On the basis of coverage : Trust can be global or local and individualistic.

Generally ,There are two kind of trust metrics on the basis of coverage.

1) Global trust metrics : These trust metrics decides the reputation of a particular user in a given community. It does not take the subjective views of each user but average the views of all users on a global level[4] , hence determining the trustworthiness of a user from viewpoint of a group of users as a community. Google's Page Rank is one of the most popular used trust metric.

2) Local trust metrics : These trust metrics take into account of subjective and personal opinions of users and predict how much a particular user should trust every other user i.e. personalization of trust for each user. Local trust metrics are computationally more expensive but much more suitable than global ones for trust based recommender systems as one user who is trustworthy to one user may be completely distrusted by another user. Mole trust , tidal trust are known examples of local trust metrics.

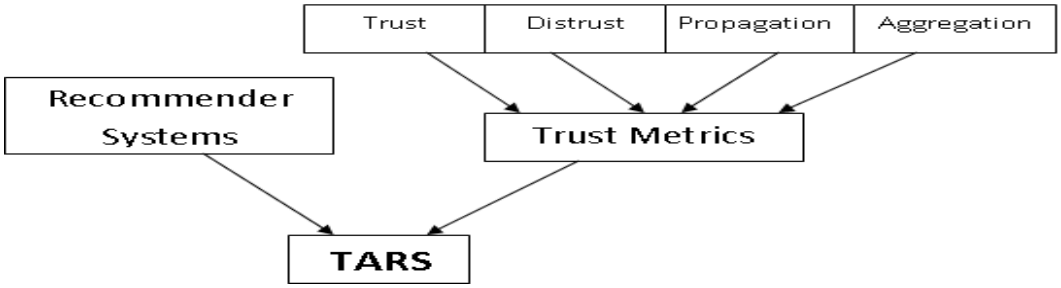


Fig. 2.3 Role of Trust metrics

2.2 Page Rank

There are two algorithms used to rank web pages based on links, PageRank and HITS (Hyperlink Induced Topic Search). Whenever a query is posted on internet there are a lots of web links that can answer that query and that creates abundance problem i.e. is if there are 100 pages that might be related to a query which one should be recommended to a user. This problem is solved by pagerank and HITS which works upon the notion of prestige in social media analysis.

Pagerank of a particular web page is the probability of being on that page and it is independent of any textual content or query[11]. It is link analysis algorithm and assigns a weight to each link that determines the importance of that page on the web. It works as follow:

Let be N nodes are connected to each other, forming internet.

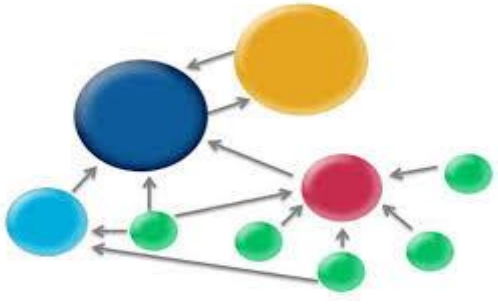


Fig. 2.4 Web of N pages

- Initialize pagerank PR(X) = 100%/N where N is the total number of pages.
- For each page X,
  PR(X) = (1-x)/N + x *∑PR(Y)/out(Y)
  It means that there are two possibilities that are
  - if randomly any node is selected with a probability (1- x) then chances of being on X are (1-x)/N
  -or if previous page Y has a link pointing to X then the chances of being on X currently are
  (PR(Y)/Total no. of links on Y). It means that Y distributes its prestige among all of its linking nodes equally.
- Pagerank of all the pages should always be equal to 100%.

The PageRank algorithm, proposed by Page et al. [11] for ranking web pages calculates the value of a web page by calculating the value of its neighboring links and provides a kind of peer assessment of the value of a Web page by taking into account not just the number of pages linking to it, but also the number of pages pointing to those pages, and so on[12]. Thus, a link from a popular page is given a higher weighting than one from an unpopular page.
Although Pagerank successfully removes the abundance problem on web but it is strongly criticized for defining the prestige of a page via a single random walk and for not taking into account the content of web page to rank it. It is also criticized for the ad hoc manner in which it put together relevance and quality at query time for the sake of efficiency.

Application in Recommender Systems

In Recommender systems, a variation of pagerank is used as global Trust metric which evaluates the trust shown by a community in a particular user of that community. If a user is trusted by important users of a community, his /her views are more valued than an ordinary user. Using this Intuition, Incorporating Pagerank as a trust metric in recommender systems arises a new application of recommender systems that is Expert finding in online communities where it is very difficult to find out relevant information. If users of a community declares someone expert in certain field, it becomes easier to rely on his views about a particular topic. These recommender systems are mostly adequate in scenarios where user is not looking for personalized recommendations but is seeking for expertise in certain area.

2.3 Online Q/A Communities

Search engines emerged as a platform where people can post their queries and seek information from various resources but it faces the difficulty where user cannot clearly express his problem with few key words and it tends to misinterpret the meaning of query, it arises the need of platform on which user can express his problem using natural language instead of keywords to ask questions. These platforms are well known as community question answering(CQA) portals. Some of the well known CQAs are stack overflow, java forum, Quora etc.

These communities are special type of network where individual participates for knowledge sharing and seeking. Those who share common interest voluntarily work together so that they can enrich their knowledge by participating in discussions related to particular topics of their interest.

It works as follow:

1) Any user can post a query on the community platform, hence starting a new thread.

2) Those who are interested in topic upon which query is posted participate in the thread by answering the question according to their expertise.

3) Some answers are considered helpful and upvoted by other users of the community and some are criticized and downvoted by the community members.

Bow-Tie Structure of Online communities

The bow tie structure, first proposed by researchers at IBM, AltaVista, and Compaq, yields insights into the complex organization of the Web network structure. Its key idea is that the web is a bow tie and has four distinct components: Core, In, Out, and ‘Tendrils‘ and ‘Tubes’ [12][14]. This structure of web graph has a central strong connected core(SCC), a subgraph(IN) with directed path coming into SCC, a component(OUT) leading out of SCC and relatively isolated tendrils attached to one of the three subgraphs. To frame online question answer communities in bow tie model, it is assumed that the central core contains users that are active and frequently ask questions and give answers.

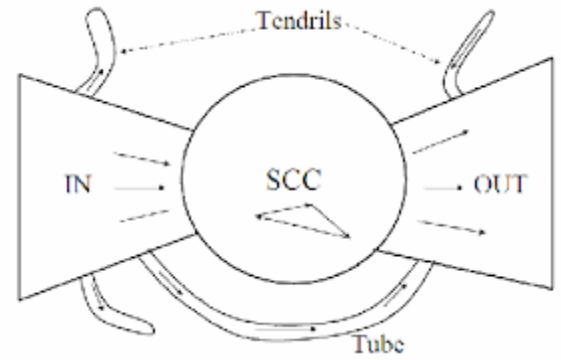


Fig. 2.5 Bow-Tie Structure

These users are the most influential users of any online community. It is a strongly connected component (SCC), meaning that one can reach every user from every other by following questioner-answerer links. The 'In' component contains users that usually only ask questions. The 'Out' consists of users that usually only answer questions posted by users in the Core. Other users, the 'Tendrils' and 'Tubes', connect to either the 'In' or 'Out' clusters, or both, but not to the Core. They are users who only answer questions posed by 'In' users or whose questions are only answered by 'Out' users[14].

In real world scenario, in mostly communities, the IN component is very large as compared to OUT and SCC that a large number of users use these platforms only when they seek help, hence making the IN component of Bow-Tie structure largest of all. The essence of these communities lie in the fact that the questioner should get an answer relevant to his problem in minimum time as possible.

2.3.1 Expert Finding on CQA sites

Community question answer platforms provide people freedom to ask their problems in natural language where question can be elaborated to a extend that can be easily understood by anybody but there is an intrinsic problem with these communities that is abundance of information and no measure for the relevance and quality of an answer.

To choose the right expert to answer a question posted by a user is one of the most challenging problems in the CQAs. To find experts is important to many real application such as identification of best answers and identification of best questions for a user to answer. In online communities the level of knowledge of each user is not known hence it is difficult to decide the quality of an answer. Therefore by determining expertise level of users, sophisticated systems for knowledge sharing can be built to make answers more reliable.

Expert finding algorithms have various applications such as:

- making questions visible to the experts who are able to respond to them .It will make the response time of a question shorter, hence increasing efficiency of the platform.
- making simple questions unseen to the experts so that experts do not waste their time in answering trivial questions[12].

In general , there are two approaches for finding experts: social network analysis and content analysis.

Social Network Analysis

In any network-based ranking algorithm individuals are considered as nodes and relationship between them are considered as links of a network. When information is exchanged between two nodes a link between them is shaped. For example, if person A responds to person B a link from A is drawn to B. After creating all possible links between all individuals a network which is called the Expertise Network (EN) is established[14].It is done as follow:

1) Creation of users network : All the users are considered as the nodes of the network and if a user respond to other, there is a link pointing from one to other.

2) Ranking users and finding experts : It involves distribution of prestige among users for example if a user 'A' has a reputation score of 5000 and he has been linked with 5 different users then his score will be uniformly distributed among all the 5 users with each getting a reputation score of 1000 from 'A'.

It has an underlying structure of Pagerank algorithm used to find rank of each user.[14]

Concept Map

It starts with extraction of concept of user's post and using it find the expertise level of the user[15].

- First of all each user's posted content is analyzed in order to create a data structure containing concept and keyword related to each post to which a particular user is associated.
- Calculating distance between the concepts : the concept in a question is mapped to that of an answer associated with it. The output of this stage is a two-dimensional matrix that holds distance between concepts.

It calculates the similarity between the concept given in question and the concept in the respective answer.

2.4 Related work

[14] test a set of network-based ranking algorithms, including PageRank and HITS, on large size social network java forum in order to identify users with high expertise and then use simulations to identify a small number of simple simulation rules governing the question-answer dynamic in the network. It also proposed a pagerank alike algorithm to find experts in online communities.

A variation of pagerank works as follow to find expert in an online community:

Let user X has helped users $U_1, U_2, U_3 ... U_N$

Then rank of X can be calculated as follow**:**

$$\mathbf{PR(X) = (1\text{-}d) + d(PR(U_1)/Y + PR(U_2)/Y + ...+PR(U_N)/Y)}$$

where with probability d, the page rank of X is the summation of the ranks of all the users he has helped divided by the total number of users Y who helped them.

[16] proposes a novel method based on social network analysis is proposed to find the experts in different contexts.[18] formulates problem of cold-start expert finding in CQA systems. It first utilizes the "following relations" between the users and topical interests to build the user-to-user graph in CQA systems. Next, It proposes the Graph Regularized Latent Model (GRLM) to infer the expertise of users based on both past question-answering activities. [19] proposes a method to find expert in an online community using both document based relevance and the prestige of the user in his knowledge community.[20] exploits user's feedback about the answers provided by a particular user and determine his rank in the community. .[21] proposes a novel method to recommend experts to the users of an question-answering online community by considering both content of user and social features. [23] proposes Competition Based Expertise Networks (CBEN), a novel community expertise network structure based on the principle of competition among the answerers of a question. It also shows that way to determine experts largely depends upon the type of community.

Chapter - 03

Proposed Framework

We propose a framework for finding experts in an online community using social network analysis and then by using collaborative filtering , finding similar experts based on their level of expertise and their topics of interest to a particular user. Once we have top- k similar experts to a given expert, he is recommended with posts to collaborate upon, on the basis of activities done by his top-k neighbor experts. We call this system **'Expert Recommender System'**. The utility of this system lies in the fact that an expert should be provided with the posts that may interest him most. If an expert collaborates with other users of same expertise level, then it will make the post information rich and will be beneficial for all the other users of the community.

3.1 Model of Expert Recommender System

A general structure of Expert Recommender system is as follow:

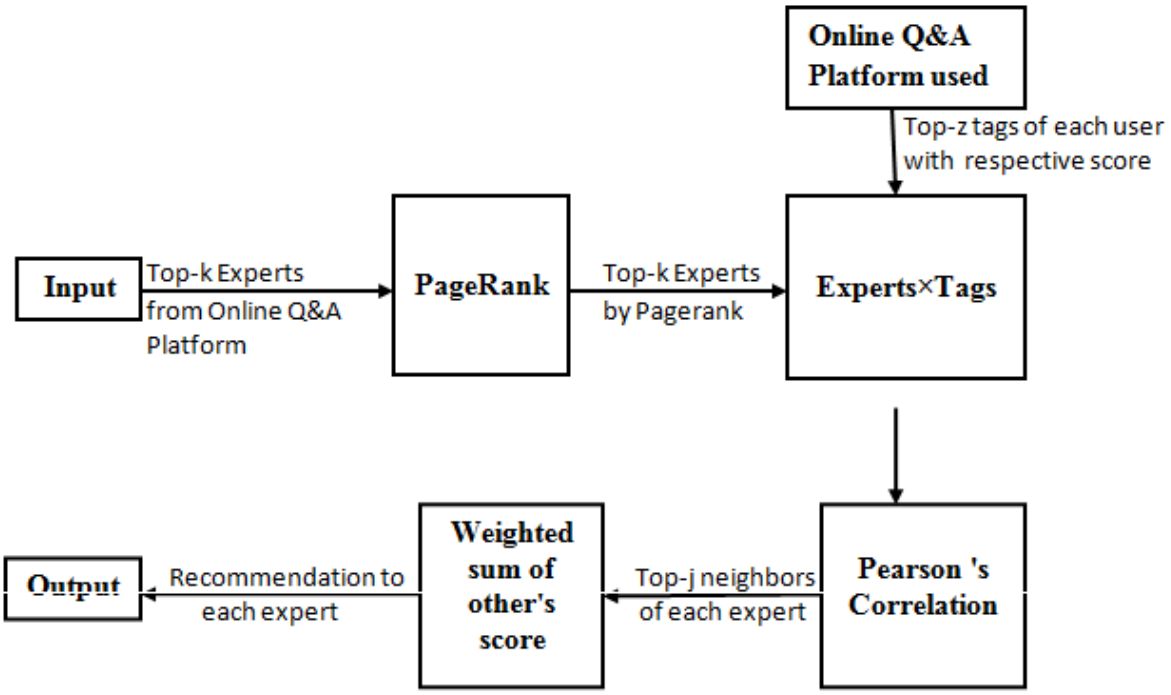


Fig. 3.1 System Architecture

The system framework is described in following steps:

1. Finding Expert in an Online community:

PageRank is used for social network analysis in order to find experts in an online community. In any network-based ranking algorithm individuals are considered as nodes and relationship between them are considered as links of a network. When information is exchanged between two nodes a link between them is shaped. For example, if person A responds to person B a link from A is drawn to B. After creating all possible links between all individuals a network which is called the Expertise Network (EN) is established[14].It is done as follow:

-Creation of users network : All the users are considered as the nodes of the network and if a user respond to other, there is a link pointing from one to other.

-Ranking users and finding experts : It involves distribution of prestige among users for example if a user 'A' has a reputation score of 5000 and he has been linked with 5 different users then his score will be uniformly distributed among all the 5 users with each getting a reputation score of 1000 from 'A'.

A variation of pagerank works as follow to find expert in an online community:

Let user X has helped users U_1,U_2,U_3...U_N

Then rank of X can be calculated as follow**:**

$$\mathbf{PR(X) = (1-d) + d(PR(U_1)/Y + PR(U_2)/Y + ...+PR(U_N)/Y)}$$

where with probability d, the page rank of X is the summation of the ranks of all the users he has helped divided by the total number of users Y who helped them.

We studied the well known online Q&A community "Stack Overflow" to apply the proposed framework. The process of finding experts using Pagerank works as follow:

- Firstly, a list of Top-k users on the basis of their reputation points earned in the community is extracted from their website using their API.
- for each user in Top-k list, Top-z posts are extracted on the basis of score of answer provided by the user in that particular post.
- for X, (X is amongst the top-k users)

  {

  1. for each post in the top-z list, these fields are needed to be extracted -

     -Post score(no. of users who find that particular post helpful)

     -Reputation of user who originally posted the question.

     -Answer count of the post(no, of answers on that particular question)

2. There are 2 parameters which can be used to determine the rank of a particular user 'A' - post score and user reputation who asked the question to which 'A' has responded. These two parameters has been considered to cover the two aspects of reputation that are - total number of people helped through the post i.e. popularity

- How prestigious member of the community is helped through the post i.e. prestige of asker in the community.

Therefore two ranks will be determined using pagerank as follow:

1. $PR(X) = (1-d) + d(PR(Q_1)/Y + PR(Q_2)/Y + ...+PR(Q_N)/Y)$

Here ,

$PR(Q_i)$ is the score of the i_{th} question (score is the number of people who found this particular question helpful) to which A has provided an answer.

Y is the total number of answers which are accepted for this question.

&

2. $PR(X) = (1-d) + d(PR(U_1)/Y + PR(U_2)/Y + ...+PR(U_N)/Y)$

Here ,

$PR(U_i)$ is the reputation of the i_{th} user to whose question A has responded to.

Y is the total number of answers which are accepted for U_i 's question.

d is the probability factor used to evaluate pagerank (d= 0.85 as fixed by [14] as optimum probability)

3. Once these two ranks are evaluated for a user 'A', these ranks are normalized in order to bring on the same scale .

eg.- if minimum post score is 0 and maximum post score is 2100,

& 0 corresponds to 0

2100 corresponds to 100

then

a post score of 1830 will correspond to 87

.- if minimum reputation of asker is 0 and maximum reputation is 220000,

& 0 corresponds to 0

220000 corresponds to 100

then

a reputation of 118670 will correspond to 54

4. Once both the ranks are on same scale i.e. 0-100 , equal weightage of 0.5 is given to each rank

and aggregate rank will be computed as:

PR(X) = 0.5 * rank based on post score + 0.5 * rank based on asker's Reputation

eg. for user 'A', rank based on post score = 1830

rank based on askers' reputation = 118670

then normalized rank based on post score = 87

normalized rank based on askers' reputation= 54

& aggregate Rank of A will be (0.5*87)+(0.5*54) = 70.5

2. Correlation of Ranks

We have extracted data of top- k users of the platform used according to their inbuilt reputation system and then we applied our framework on these users and get our own list of experts as an output. To study the effectiveness of the proposed system, we compared these two lists by using a correlation measure known as Spearman's rho.

Spearman's Rho is a product–moment correlation coefficient devised as a measure of the degree of agreement between two rankings.

$$r_s = \left[1 - \frac{6 \sum D^2}{N^3 - N} \right]$$

where

- D, is the difference between the two ranks of each observation.
- *n* is the number of observations

This correlation proves the effectiveness of Page rank in finding the experts in online Q&A communities.

3. Collaborative Filtering

In traditional collaborative filtering, It uses the known preferences of a group of users to make recommendations or predictions of those unknown preferences for other users[3]. It firstly finds out the similar users to a particular user 'A' and then recommends the items liked by those users to 'A'. For example if A & B rates a movie 4 out of 5 then intuitively their

preference over movies should be same and if A rates a movie 4 out of 5 and C rates the same movie 1 out of 5 then intuitively their preference over movies should differ.

Mainly correlation based similarity is used in which similarity $w_{u,v}$ between two users u and v, or $w_{i,j}$ between two items i and j, is measured by computing the Pearson correlation or other correlation-based similarities[3]

Pearson correlation between users u and v is

$$w_{u,v} = \frac{\sum_{i \in I} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i \in I} (r_{v,i} - \bar{r}_v)^2}},$$

where I is the total set of items rated by both u and v. $r_{u,i}$ and $r_{v,i}$ are the rating provided for item i by user u and v respectively.

We used a slight variation of collaborative filtering in which

User = Experts determined using page rank

Items = Tags most popularly used in the community.

The resulting matrix will be Experts × Tags and let this matrix is called M, then each entry in M will be

$m_{u,v}$ = Score of expert 'u' in tag 'v'

Now the pearson's correlation will be modified as :

$$w_{u,v} = \frac{\sum_{i \in I} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i \in I} (r_{v,i} - \bar{r}_v)^2}},$$

Here,

I = set of tags used and each 'i' belongs to I.

u and v = experts

$r_{u,i}$ = score of expert u in tag i

r_u = average score of user u in all the tags

$r_{v,i}$ = score of expert v in tag i

r_v = average score of user v in all the tags

Using this variation of pearson's correlation, for a expert 'X' , all the top-k neighbors will be identified.

4. Recommendations

For Recommendations, Traditional method which is used is Known as Weighted Sum of Others' Ratings. To make a prediction for the active user, a, on a certain item, i, a weighted average of all the ratings on that item is taken according to the following formula

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u \in U} \left(r_{u,i} - \bar{r}_u\right) \cdot w_{a,u}}{\sum_{u \in U} \left| w_{a,u} \right|},$$

where

$P_{a,i}$ = prediction for the active user, a, on a certain item, i

U = Set of all the users

r_a and r_u = average ratings for the user a and user u on all other rated items

$w_{a,u}$ = weight between the user a and user u determined by similarity metric used.

The summations are over all the users $u \in U$ who have rated the item i.

For our system we have modified the meanings of recommendation, Once we have tok-k neighbors to each expert, they will be recorded in the database Now,

Let A be an expert and U is the set of similar experts.

whenever a user who belongs to U, Performs an activity that is post a question, post an answer or comment or participate in various ongoing competitions, user A will be informed about the same and he will be given chances to earn more reputation score by indulging in discussion with people having same level of expertise as him.

3.2 Algorithm Proposed

The pseudocode for proposed framework is as follow:

Input : Top-k users along with their reputation score from an online Q&A community

Output : Top-z neighbors(on basis of similarity) of each of the Top-k experts

Steps:

1. Extract top-k users from an online Q&A community (as we used stack overflow).

2. Applying a variant of pagerank to find experts. It includes two measures to compute two scores.

2.1.Using Post score(Coverage) : Post score (no. of users helped) is divided equally among all of them who have answered in that post

2.2.Using Reputation of asker(Prestige) : Reputation of asker is divided equally among all of them who have answered in that post

3. Finding aggregate score: It involves following steps:

3.1. Normalize both the scores obtained from previous step on a scale of 1-100

3.2.Giving equal weightage to both coverage and prestige, these two scores are multiplied by 0.5 respectively and then added to obtain aggregate score.

4. Correlation of two ranks: Spearman's rho is used to determine correlation between two ranks(one obtained from stack overflow and other calculated by our variant of pagerank) to show the efficiency of used variant of pagerank and also to show effect of outliers on data.

5. Determining our rank for each user on the basis of score obtained by them.

6.. Extract top-w tags for each expert and produce matrix experts × tags.

6. After completing the matrix with available score of tags for each user from their profiles in the community.

7. Apply pearson's correlation to obtain top-z most similar experts to a said expert.

Chapter - 04

Implementation

4.1 Platform Used

Stack OverFlow is a classic example of online Q&A Community which has 14M questions, 22M answers, 58M comments, 49K tags and 73M users till date. It assigns Reputation points to each of its users[24].

Reputation is a rough measurement of how much the community trusts a particular user. It is earned by convincing the community members that an user knows what he is talking about. Reputation score is measurement of one's expertise in particular fields. However, Basic use of the site, including asking questions, answering, and suggesting edits, does not require any reputation at all. But the more reputation an user earns, the more privileges he gains. The primary way to gain reputation is by posting good questions and useful answers. Votes on these posts cause the respective user to gain (or sometimes lose) reputation[25]. **An User gains reputation when:**

- question is voted up: +5
- answer is voted up: +10
- answer is marked "accepted": +15 (+2 to acceptor)
- suggested edit is accepted: +2 (up to +1000 total per user)
- bounty awarded to user's answer: + full bounty amount
- one of user's answers is awarded a bounty automatically: + half of the bounty amount
- site association bonus: +100 on each site (awarded a maximum of one time per site)
- example user contributed to is voted up: +5
- proposed change is approved: +2
- first time an answer that cites documentation user contributed to is upvoted: +5

Voting is central to stack overflow's model of providing quality questions and answers; it helps

-good content to rise to the top[25]

-incorrect content to fall to the bottom

-users who consistently provide useful content accrue reputation and are granted more privileges on the site

Voting up a question or answer signals to the rest of the community that a post is interesting, well-researched, and useful, while voting down a post signals the opposite: that the post contains wrong information, is poorly researched, or fails to communicate information. The more that people vote on a post, the more certain future visitors can be of the quality of information contained within that post and upvotes are also way to thank the author of a good post for the time and effort put into writing it.

Tags : A tag is a word or phrase that describes the topic of the question. Tags are a means of connecting experts with questions they will be able to answer by sorting questions into specific, well-defined categories.Tags can also be used to help users identify questions that are interesting or relevant to them[25]Each question may only contain 5 tags at a maximum, so ,the ones that best describe question should be chosen. **Post Score** is total number of upvotes minus downvotes. It represents how many users are helped by a particular post.

A sample post from stack overflow is as follow:

Fig. 4.1. Sample Post

Here,

Upvote : This question shows research efforts and it is clear & useful.

Downvote : This question does not show any research efforts and it is unclear or not useful.

Post Score: Upvotes - Downvotes

Tags: All the tags that are talked about in this particular post.

Asker: One who is owner of the question in the post

Reputation: Reputation of the asker.

4.2 Sample Implementation

A sample implementation of expert recommender system on the online community "Stack Overflow" is given in this section. Stack Overflow API is used to extract required data from the community. The implementation comprises of following steps.

1. Extraction of Top-50 users of stack overflow on the basis of reputation : Using SQL query top-50 users of stack overflow are extracted from their database through the API provided by stack exchange which is parent site if stack overflow.

```
Select TOP(50) Reputation, Id, Displayname from Users
Group by Id, Reputation, Displayname
order by Reputation desc
```

Where users is the table containing details of all the users, from which we have extracted display name, Id and reputation score of top 50 users of the community. The table containing top 50 users of site is attached in appendix B. A sample of this table is given in table 4.1

2. For each user in top-50 list, top-20 posts are extracted where user has answered in the particular post. Posts are ordered by votes that the user has achieved on an answer provided by him in a post. Eg. User A has post P1 as top post where his answer has got 500 votes and post P2 achieved 2nd position with user A's answer on post P2 has achieved 480 votes and so on. Stack overflow's database contains a table known as 'posts' which contains details of all the posts of the community.

	Reputation	Id	DisplayName
2	953794	22656	Jon Skeet
3	738904	29407	Darin Dimitrov
4	733062	157882	BalusC
5	699810	17034	Hans Passant
6	687662	6309	VonC
7	672890	23354	Marc Gravell
8	650254	115145	CommonsWare
9	584007	34397	SLaks
10	561827	100297	Martijn Pieters
11	557888	893	Greg Hewgill
12	553741	1144035	Gordon Linoff
13	544953	157247	T.J. Crowder
14	537093	19068	Quentin
15	532757	14860	paxdiablo
16	520960	95810	Alex Martelli
17	508538	335858	dasblinkenlight
18	495769	5445	CMS
19	490861	13302	marc_s
20	489630	61974	Mark Byers
21	488873	23283	JaredPar
22	485112	20862	Ignacio Vazquez-Abrams
23	481567	69083	Guffa
24	478879	15168	Jonathan Leffler
25	465322	13249	Nick Craver

Table 4.1 Top-50 users of Stack overflow

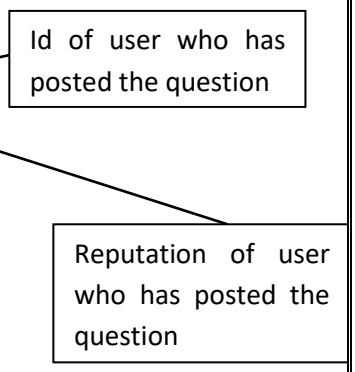
We have extracted post Id, post score, reputation of asker, answer count for each post. Eg. Jon skeet is the top user of stack overflow. To extract top 20 posts of Jon, where he has answered in the post, his Id is used, that is extracted in the previous table.

User name : Jon Skeet

Id : 22656

```
select postid as [Post Link],postscore,users.id as ownerid,
users.reputation as ownerreputation
FROM
(select posts.id as postid, posts.score as postscore, posts.owneruserid as owner
FROM
(SELECT TOP(21) Id as [Post Link], posts.owneruserid, parentId as par, score
from Posts where posts.owneruserid = 22656 order by score desc) as jon JOIN
Posts
ON posts.id = jon.par) as jon1 JOIN Users
ON Users.id = jon1.owner
order by postscore desc
```

Table containing details of top-20 posts belonging to Jon is:



	postid	postscore	answers	ownerid	ownerrep
2	6841333	5149	8	342235	62130
3	7074	4313	51	571	15612
4	176264	3084	31	2041950	20874
5	8881291	2408	14	953140	12755
6	285793	2036	21	33203	23880
7	247621	1654	9	22656	950709
8	285177	1450	13	33203	23880
9	541487	1442	37	65374	7449
10	886955	1290	28	15108	8799
11	519520	1262	33	2695	26989
12	263400	1028	15	4227	20858
13	232535	673	8	30280	27622
14	7325278	603	6	916075	3398
15	221925	596	10	45	57574
16	489258	594	16	13913	75624
17	799446	565	21	5975	19448
18	4317479	514	3	520692	4994
19	2483659	511	9	194562	3470
20	853526	471	14	1816	15990

Table 4.2 Details of top-20 posts belonging to Jon

Similarly, Details of Top-20 posts of all the top-50 users of stack overflow are extracted.

3. Now, for each user two ranks will be calculated according to a variation of pagerank algorithm in which One is based on the post score and another on the reputation of the asker.

3.1. On the basis of asker's Reputation : Let user A has asked a question and he has a reputation score of 5000.

According to pagerank, if a user's question is answered by 50 users, then his reputation is equally divided into all the 50 users irrespective of the content provided by each user.

Let a user X answered A's question, then contribution of A to the rank of X will be

Reputation of A / Total no. of answers to A's question

Here, contribution of A to the rank of X will be

5000/50 = 100 points.

3.2. On the basis of post's score : Let user A has answered a question Q and 1020 users has found that post useful and 20 users found it not useful , so the post score(upvotes - downvotes) will be 1000.

According to pagerank, if a question is answered by 50 users, then its score is equally divided into all the 50 users irrespective of the content provided by each user.

Let a user X answered a question Q, then contribution of Q to the rank of X will be

Post score of Q / Total no. of answers to question Q

Here, contribution of Q to the rank of X will be

1000/50 = 20 points.

Similarly, for a particular user, reputation earned by him in all the top 20 posts will be calculated.

Now for user X, Reputation earned by him in all of his top-20 posts is calculated and then two ranks will be determined using pagerank as follow:

1. $PR(X) = (1-d) + d(PR(Q_1)/Y + PR(Q_2)/Y + ...+PR(Q_N)/Y)$

Here ,

$PR(Q_i)$ is the score of the i_{th} question (score is the number of people who found this particular question helpful) to which A has provided an answer.

Y is the total number of answers which are accepted for this question.

&

2. $PR(X) = (1-d) + d(PR(U_1)/Y + PR(U_2)/Y + ...+PR(U_N)/Y)$

Here ,

$PR(U_i)$ is the reputation of the i_{th} user to whose question A has responded to.

Y is the total number of answers which are accepted for U_i 's question.

d is the probability factor used to evaluate pagerank (d= 0.85 as fixed by (8) as optimum probability)

For eg. User name : Jon Skeet

Here,

rep_score = postscore / answers

rep_ans = ownerrep / answers

2153.775 is the summation of all the rep_score

139610.6 is the summation of all the rep_ans

Rank on basis of post score = 0.15 + (0.85 * 2153.775) = 1830.859

Rank on basis of post score = 0.15 + (0.85 * 139610.6) = 118669.2

	postid	postscore	answers	ownerid	ownerrep	rep_score	rep_ans
2	6841333	5149	8	342235	62130	643.625	7766.25
3	7074	4313	51	571	15612	84.56863	306.1176
4	176264	3084	31	2041950	20874	99.48387	673.3548
5	8881291	2408	14	953140	12755	172	911.0714
6	285793	2036	21	33203	23880	96.95238	1137.143
7	247621	1654	9	22656	950709	183.7778	105634.3
8	285177	1450	13	33203	23880	111.5385	1836.923
9	541487	1442	37	65374	7449	38.97297	201.3243
10	886955	1290	28	15108	8799	46.07143	314.25
11	519520	1262	33	2695	26989	38.24242	817.8485
12	263400	1028	15	4227	20858	68.53333	1390.533
13	232535	673	8	30280	27622	84.125	3452.75
14	7325278	603	6	916075	3398	100.5	566.3333
15	221925	596	10	45	57574	59.6	5757.4
16	489258	594	16	13913	75624	37.125	4726.5
17	799446	565	21	5975	19448	26.90476	926.0952
18	4317479	514	3	520692	4994	171.3333	1664.667
19	2483659	511	9	194562	3470	56.77778	385.5556
20	853526	471	14	1816	15990	33.64286	1142.143
21						2153.775	139610.6
22						1830.859	118669.2

Table 4.3 Details of top-20 posts belonging to Jon along with rep_score and rep_ans

Now to bring these two ranks to the same scale, they are normalized to a scale of 0-100 by considering minimum value of post score = 0

minimum value of owner reputation = 0

maximum value of post score = 2100

maximum value of owner reputation = 220000

then ,

score of jon skeet on the basis of post score will be normalized as 1830.85 ⟶ 87

score of jon skeet on the basis of owner's reputation will be normalized as 118669.2 ⟶ 54

& Aggregate score of Jon (giving equal weightage to both the scores) = (0.5 * 87) + (0.5*54)

= 70.56

Similarly, aggregate scores for all the top-50 users is calculated.

4. Ranking of all the users according to our calculated aggregated scores: Top-50 users according to variant of pagerank is determined by putting the obtained aggregate score of all the users in descending order.

The table containing our top-50 users is attached in the appendix B and a sample of the same is:

Rank	User	Aggregate Score
1	Jon Skeet	70.56213744
2	T.J. Crowder	62.97963769
3	VonC	49.2521776
4	Hans Passant	47.63006023
5	Alex Martelli	41.18048366
6	BalusC	41.02895816
7	CMS	36.03027152
8	Eric Lippert	35.69
9	Charles Bailey	33.46
10	Felix Kling	32.82
11	JaredPar	31.61153823
12	Greg Hewgill	31.28896175
13	BoltClock	31.10995622
14	Ignacio Vazquez-A	30.81199561
15	Darin Dimitrov	30.27262948
16	unutbu	29.31658357
17	kennytm	29.13872459
18	Gumbo	28.56880032
19	Marc Gravell	27.90613291
20	Quentin	27.13556015
21	Mark Byers	26.81714192
22	Stephen C	26.28341067
23	bobince	26.25084452
24	marc_s	26.21283278

Table 4.4 Our top-50 users

5. Extraction of Top-3 tags of each user in Top-k list along with their scores for the respective tags and construction of Expert×Tag matrix. Due to non availability of structure in the API of stack Overflow that can relate each user to its top tags, tags and their respective scores have been extracted manually. To avoid the cumbersome task of extraction of data, top-30 users are selected to put in the Expert×Tag matrix and only top-3 tags of each user are extracted to showcase the essence of our proposed framework. Once top-3 tags for each user

is obtained, the matrix is completed by obtaining the score of all the tags that are in the list (top-3 tags of all the users) for each user.

The sample of the matrix is as follow :

		username										
2	tags	1	2	3	4	5	6	7	8	9	10	11
3	.net	65	13	1	25	1	32	1	12	1		1
4	algorithm	4	1		1		1		1		2	
5	android	4		1				71	1	1		1
6	android-intent		1		1	1	1	4		1	1	
7	angular	1						1	1	1		1
8	arrays	8	1	1		1		3	3	1	1	
9	asp.net mvc		41		1	1	1		3	1		1
10	asp.net mvc-6		32			1		1	1		1	
11	bash			1	1					1		1
12	c	1	1		2	1	1		1	1	4	1
13	C#	188	44		46		85	1	32		2	
14	c++			1	8	1				1	8	1
15	css	1	1	4			1		2			1
16	dart				1	1		1	1		1	
17	dataframe		1	1						1		1
18	delphi				1	1	1	1	1			
19	dictionary		1			1				6	1	1
20	django		1	1	1	1		1	1	2		1
21	dplyr						1			1	1	
22	eclipse	3		4		13		2	1			
23	ecmascript-6				1	1		1			1	1
24	firebase		1	1		1	1		1	1		
25	firebase-databse										1	

Table 4.5 Expert×Tag matrix

Here all the values are in thousands i.e. score of user 1 for .net is 65k.

Now all the values are normalized by putting a score of 1-5 for the intervals as follow :

1-25 ⟶ 1

26-50 ⟶ 2

51-75 ⟶ 3

76-100 ⟶ 4

>100 ⟶ 5

5. Applying pearson's correlation for each user, to find his correlation with all the other users in the list : Pearson correlation is used to find similarity between two given users. For each user in top-30, his correlation is determined with all the other users in the list.

eg. user name : Jon Skeet

In the output of pearson's correlation, a table is produced that contains all the users that are similar to Jon skeet in descending order of their similarity measure.

Code for pearson correlation is provided in appendix A.

users	similarity with Jon
Marc Gravell	0.68990064
CommonsWare	0.68990064
JB Nizet	0.68989986
Felix Kling	0.68989986
cletus	0.68989986
Gumbo	0.68989986
unutbu	0.68989986
Hans Passant	0.6898997
VonC	0.6898997
SLaks	0.6898997
Martijn Pieters	0.6898997
Greg Hewgill	0.6898997
Gordon Linoff	0.6898997
T.J. Crowder	0.6898997
Quentin	0.6898997
paxdiablo	0.6898997
Alex Martelli	0.6898997
dasblinkenlight	0.6898997
CMS	0.6898997
marc_s	0.6898997
Mark Byers	0.6898997
JaredPar	0.6898997
Ignacio Vazquez-Abrams	0.6898997
Guffa	0.6898997

Table 4.6 users that are similar to Jon skeet

6. Finding Top-10 neighbors of a particular User : When correlation of an user A with all the other users is arranged in descending order. Top-10 experts in this list will be neighbors of A . Select top-10 users from the above result table who, when indulged in any activity, Jon skeet will be notified about the same.Top-10 neighbors of Jon are in table 4.7 as below:

users
Marc Gravell
CommonsWare
JB Nizet
Felix Kling
cletus
Gumbo
unutbu
Hans Passant
VonC
SLaks

Chapter - 05

Result and Analysis

5.1 Top-50 users according to page rank and its correlation with the list of top-50 users from the community :

Correlation is calculated using spearman's rho which is a measure of the degree of agreement between two rankings. It is calculated with and without outliers in the data .

5.1.1 Results

A) With Outliers in data

When aggregated scores obtained by each user are arranged in descending order, we obtain rank for each user in the list which is represented by column "our rank". 'Rank Given ' is the rank obtained from stack overflow.

To calculate correlation between the two ranks, spearman's correlation is used.

$$r_s = \left[1 - \frac{6\sum D^2}{N^3 - N} \right]$$

score	users	Rank given	our rank	D	D^2
117.9022875	Johannes Schaub - litb	42	1	41	1681
74.32000569	Eric Lippert	26	2	24	576
73.61017071	Felix Kling	28	3	25	625
70.56213744	Jon Skeet	1	4	-3	9
62.97963769	T.J. Crowder	12	5	7	49
52.50019011	Charles Bailey	38	6	32	1024
50.18009129	deceze	48	7	41	1681
49.2521776	VonC	5	8	-3	9
47.63006023	Hans Passant	4	9	-5	25
43.37476852	Konrad Rudolph	46	10	36	1296
43.09217572	cletus	29	11	18	324
41.18048366	Alex Martelli	15	12	3	9
41.02895816	BalusC	3	13	-10	100
39.38509465	jfriend00	49	14	35	1225
36.03027152	CMS	17	15	2	4
34.53524605	Jerry Coffin	47	16	31	961
31.61153823	JaredPar	20	17	3	9
31.28896175	Greg Hewgill	10	18	-8	64
31.10995622	BoltClock	35	19	16	256
30.81199561	Ignacio Vazquez-Abrams	21	20	1	1
30.27262948	Darin Dimitrov	2	21	-19	361
29.31658357	unutbu	31	22	9	81
29.13872459	kennytm	45	23	22	484
28.56880032	Gumbo	30	24	6	36

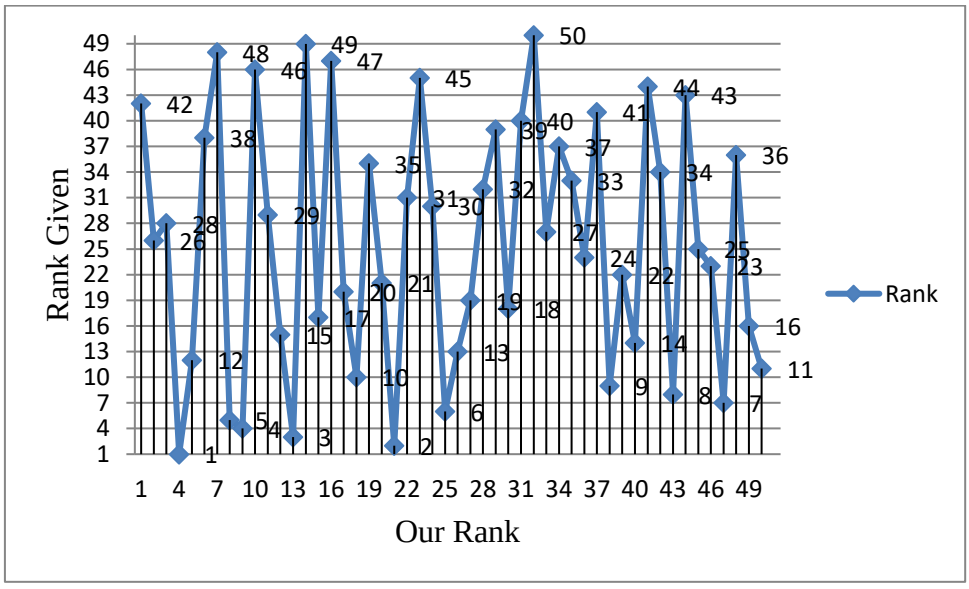
27.90613291	Marc Gravell	6	25	-19	361
27.13556015	Quentin	13	26	-13	169
26.81714192	Mark Byers	19	27	-8	64
26.28341067	Stephen C	32	28	4	16
26.25084452	bobince	39	29	10	100
26.21283278	marc_s	18	30	-12	144
25.36556133	Peter Lawrey	40	31	9	81
22.90690345	Pekka ì›f	50	32	18	324
20.9662829	JB Nizet	27	33	-6	36
20.7552193	Bozho	37	34	3	9
20.49485811	Pascal Thivent	33	35	-2	4
20.47427134	Nick Craver	24	36	-12	144
20.31859284	tvanfosson	41	37	4	16
19.69411416	Martijn Pieters	9	38	-29	841
19.67791548	Guffa	22	39	-17	289
19.18424199	paxdiablo	14	40	-26	676
18.6648592	Daniel Roseman	44	41	3	9
18.0915541	anubhava	34	42	-8	64
17.75524563	SLaks	8	43	-35	1225
16.45195812	Oded	43	44	-1	1
16.14001329	David Heffernan	25	45	-20	400
15.61586034	Jonathan Leffler	23	46	-23	529
15.00619454	CommonsWare	7	47	-40	1600
13.68640598	Reed Copsey	36	48	-12	144
13.0558257	dasblinkenlight	16	49	-33	1089

Table 5.1 Correlation with Outliers

Here D and D^2 are calculated in the table itself & $\sum D^2 = 20746$ & $N = 50$

so, $N^3 - N = 124950$

Correlation(rank given, our Rank) = 1 -(6*20476) / 124950 = 0.003794



Graph 5.1 With Outliers in data

B) Without Outliers in data

To calculate correlation between the two ranks, spearman's correlation is used.

$$r_s = \left[1 - \frac{6 \sum D^2}{N^3 - N} \right]$$

Here D and D^2 are calculated in the table itself.

$\sum D^2 = 11529$

$N = 50$

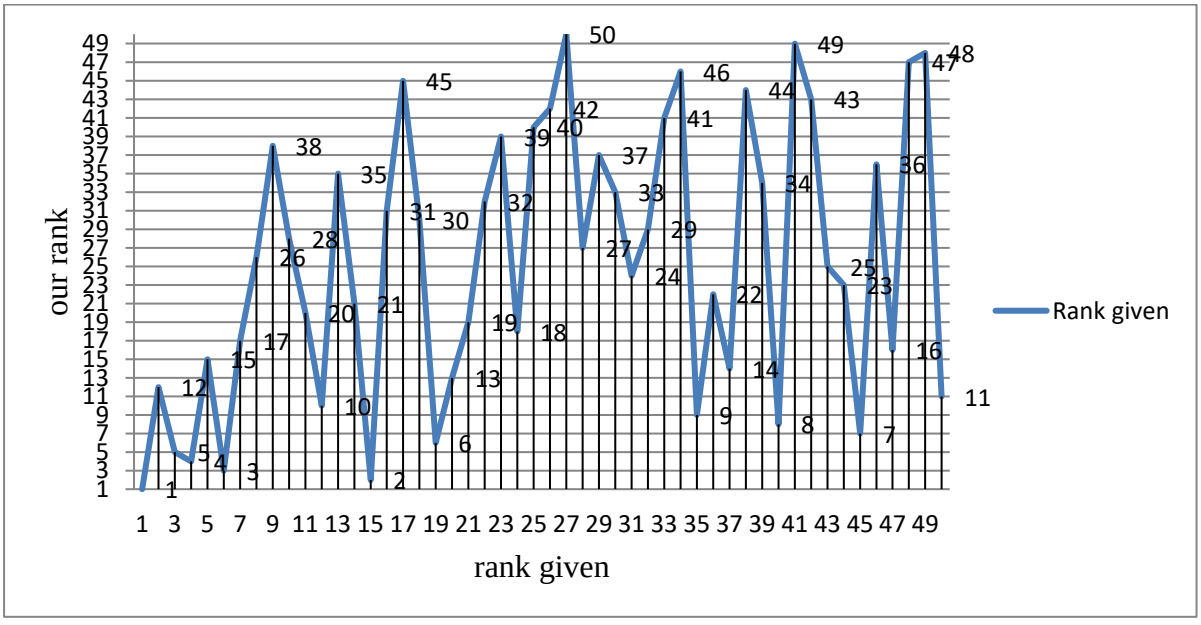
$N^3 - N = 124950$

Correlation(rank given, our Rank) = 1 -(6*11529) / 124950 = 0.45

User	Aggregate Score	rank given	our rank	D	D^2
Jon Skeet	70.56213744	1	1	0	0
T.J. Crowder	62.97963769	12	2	10	100
VonC	49.2521776	5	3	2	4
Hans Passant	47.63006023	4	4	0	0
Alex Martelli	41.18048366	15	5	10	100
BalusC	41.02895816	3	6	-3	9
CMS	36.03027152	17	7	10	100
Eric Lippert	35.69	26	8	18	324
Charles Bailey	33.46	38	9	29	841
Felix Kling	32.82	28	10	18	324
JaredPar	31.61153823	20	11	9	81
Greg Hewgill	31.28896175	10	12	-2	4
BoltClock	31.10995622	35	13	22	484
Ignacio Vazquez-A	30.81199561	21	14	7	49
Darin Dimitrov	30.27262948	2	15	-13	169
unutbu	29.31658357	31	16	15	225
kennytm	29.13872459	45	17	28	784
Gumbo	28.56880032	30	18	12	144
Marc Gravell	27.90613291	6	19	-13	169
Quentin	27.13556015	13	20	-7	49
Mark Byers	26.81714192	19	21	-2	4
Stephen C	26.28341067	32	22	10	100
bobince	26.25084452	39	23	16	256
marc_s	26.21283278	18	24	-6	36

Peter Lawrey	25.36556133	40	25	15	225
Johannes Schaub	23.44	42	26	16	256
Pekka ì›ƒ	22.90690345	50	27	23	529
JB Nizet	20.9662829	27	28	-1	1
Bozho	20.7552193	37	29	8	64
Pascal Thivent	20.49485811	33	30	3	9
Nick Craver	20.47427134	24	31	-7	49
cletus	20.47	29	32	-3	9
tvanfosson	20.31859284	41	33	8	64
Konrad Rudolph	20.26	46	34	12	144
Martijn Pieters	19.69411416	9	35	-26	676
Guffa	19.67791548	22	36	-14	196
paxdiablo	19.18424199	14	37	-23	529
Daniel Roseman	18.6648592	44	38	6	36
anubhava	18.0915541	34	39	-5	25
SLaks	17.75524563	8	40	-32	1024
jfriend00	17.72	49	41	8	64
Oded	16.45195812	43	42	1	1
David Heffernan	16.14001329	25	43	-18	324
Jonathan Leffler	15.61586034	23	44	-21	441
CommonsWare	15.00619454	7	45	-38	1444
Reed Copsey	13.68640598	36	46	-10	100
dasblinkenlight	13.0558257	16	47	-31	961
Jerry Coffin	12.2	47	48	-1	1
deceze	8.23	48	49	-1	1

Table 5.2 Correlation without Outliers in data



Graph 5.2 Without outliers in data

5.1.2 Analysis

1) We have observed that there are outliers in the data which are affecting our results to a great extent. For eg. if a user X has answered questions of other users with an average reputation of 100 but in one post he has answered a user whose reputation is 10000, then this particular entry will be considered as an outlier as it will affect the aggregate score of user X.
Due to presence of outliers in entries of every user, the correlation between the two ranks (one provided by stack overflow and other that is determined by us using variant of pagerank.) is extremely low.
After calculating the correlation with the presence of outliers, we have calculated the correlation by removing the extreme outliers, which have improved our results.

2) As this is the limitation of social network analysis that it divides the reputation of a particular user among all of his helpers, irrespective of the contribution made by each helper. This limitation has also affected our results. For eg. If a post has score of 5000 and it is answered by 100 users but only top 10 users has answered very well as compared to rest of the users, so it will be unfair to divide the post score equally in all the users who have contributed in the particular post. Due to rules of social network analysis, we are restricted to divide the post score equally. This also has affected our results and reduced the correlation between two lists of top-50 (one provided by stack overflow and other that is determined by us using variant of pagerank.) users.

5.2 Top-10 neighbors of a particular user.

Using pearson's correlation, correlation of each user A is determined with every other user and then out of them top-10 neighbors will be selected that are most similar to user A. when these users will indulge in any activity, A will be notified about the same.

5.2.1 Results

A sample result for the top most user in our list is included below:

users	similarity with Jon
Marc Gravell	0.68990064
CommonsWare	0.68990064
JB Nizet	0.68989986
Felix Kling	0.68989986
cletus	0.68989986
Gumbo	0.68989986
unutbu	0.68989986
Hans Passant	0.6898997
VonC	0.6898997
SLaks	0.6898997
Martijn Pieters	0.6898997
Greg Hewgill	0.6898997
Gordon Linoff	0.6898997
T.J. Crowder	0.6898997
Quentin	0.6898997
paxdiablo	0.6898997
Alex Martelli	0.6898997
dasblinkenlight	0.6898997
CMS	0.6898997
marc_s	0.6898997
Mark Byers	0.6898997
JaredPar	0.6898997
Ignacio Vazquez-Abrams	0.6898997
Guffa	0.6898997

Table 5.3 Similarity measure of a particular user with all the other users

When correlation of an user A with all the other users is arranged in descending order. Top-10 experts in this list will be neighbors of A . Select top-10 users from the above result table who, when indulged in any activity, Jon skeet will be notified about the same.

Whenever a user who belongs to table 5.4 Performs an activity that is post a question, post an answer or comment or participate in various ongoing competitions, Jon will be informed about the same and he will be given chances to earn more reputation score by indulging in discussion with people having same level of expertise as him.

Top-10 neighbors of Jon are

users
Marc Gravell
CommonsWare
JB Nizet
Felix Kling
cletus
Gumbo
unutbu
Hans Passant
VonC
SLaks

Table 5.4 Top-z neighbors of a particular user

5.2.1 Analysis

For the sake of simplicity, we have mapped the whole range of scores into five scores only, due to this reason, we are getting same score for many tags. Due to same ratings of several tags with multiple users, User A has many similar users to him having same similarity measure. If crisp values will be considered rather than a single value for an intervals, it will make the calculations cumbersome but it will make the results more accurate

Chapter - 06

Conclusion

6.1 Conclusion

The recommendation system we have proposed is called **Expert Recommendation system** which incorporates two important features of recommender systems that are trust and similarity achieved by using a well known global trust metric Page rank and collaborative filtering respectively. Online Q&A communities are perfect example of platforms where people participate to seek expertise on their interested topics. People do not look for personal advices but expert views on such platforms therefore, expert finding is an integral part of these communities. In order to trust someone's opinion who is not known in person by the users of the community, it is necessary to state credibility of such experts by setting some parameters.

The utility of this system lies in the fact that these communities always face the problem of information abundance and to get right person indulged in right threads of questions and answers is one of the biggest challenges till date but this system makes sure that an expert will be provided with relevant questions of his fields of interest and due to contribution of people of same level of expertise and same kind of interests, the threads will be information rich which is beneficial for all the users of the community

We have taken a very small dataset for the sake of simplicity, but results are clearly showing the utility of the proposed work. Our next step involves testing the framework on a larger dataset and on a real-time system.

6.2 Limitations

Few are the limitations of the system which are also the limitations of underlying techniques used.

1) Outliers : We have observed that there are outliers in the data which are affecting our results to a great extent. For eg. if a user X has answered questions of other users with an average reputation of 100 but in one post he has answered a user whose reputation is 10000, then this particular entry will be considered as an outlier as it will affect the aggregate score of user X.

Due to presence of outliers in entries of every user, the correlation between the two ranks is affecting adversely.

2) Equal weightage assigned to all the participating users : As this is the limitation of social network analysis that it divides the reputation of a particular user among all of his helpers, irrespective of the contribution made by each helper. This limitation has also affected our results. For eg. If a post has score of 5000 and it is answered by 100 users but only top 10 users has answered very well as compared to rest of the users, so it will be unfair to divide the post score equally in all the users who have contributed in the particular post.

3) Reputation is not at all depending on the concept involved in questions and answers : It does not include the similarity between the concept involved in the question and that in answer and reputation is purely determined by the asker's reputation and post score.

6.3 Future work

1. Concept map: A weightage should be assigned to the similarity measure between the concept involved in questions and answers It starts with extraction of concept of user's post and using it find the expertise level of the user.

- First of all each user's posted content is analyzed in order to create a data structure containing concept and keyword related to each post to which a particular user is associated.
- Calculating distance between the concepts : the concept in a question is mapped to that of an answer associated with it. The output of this stage is a two-dimensional matrix that holds distance between concepts.
- It calculates the similarity between the concept given in question and the concept in the respective answer.

2. Weightage should be assigned to each answerer on the basis of votes he has achieved for his answer on a particular post and Reputation of asker or post score should be divided among the helpers on the basis of basis of weightage given to each of them.

3. Tag specific Recommendation : A user should be recommended with post in which his neighbors are indulging and also that post should involve user's top tags. This will make the recommender system more precise and each user's expertise will be used to its level best and he will be provided with opportunities to sharpen his own knowledge about the topics.

References

[1] Resnick, P., & Varian, H. R. (1997). Recommender systems. *Communications of the ACM*, *40*(3), 56-58.

[2] Ricci, F., Rokach, L., & Shapira, B. (2011). *Introduction to recommender systems handbook* (pp. 1-35). springer US.

[3] Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, *2009*, 4.

[4] Herlocker, J. L., Konstan, J. A., & Riedl, J. (2000, December). Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work* (pp. 241-250). ACM.

[5] Shi, Y., Larson, M., & Hanjalic, A. (2014). Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)*, *47*(1), 3.

[6] Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender systems handbook* (pp. 73-105). Springer US.

[7] Massa, P., & Avesani, P. (2007, October). Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems* (pp. 17-24). ACM.

[8] Massa, P., & Avesani, P. (2004, October). Trust-aware collaborative filtering for recommender systems. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* (pp. 492-508). Springer Berlin Heidelberg.

[9] O'Donovan, J., & Smyth, B. (2005, January). Trust in recommender systems. In *Proceedings of the 10th international conference on Intelligent user interfaces* (pp. 167-174). ACM.

[10] Kumar A., Bala I. (2017, June).A Comprehensive study of TARS: Definition, Metrics and Advancements. Accepted in UPCON 2017

[11] Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The PageRank citation ranking: Bringing order to the web*. Stanford InfoLab.

[12] Chakrabarti, S. (2002). *Mining the Web: Discovering knowledge from hypertext data*. Elsevier.

[13] Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., ... & Wiener, J. (2000). Graph structure in the web. *Computer networks*, *33*(1), 309-320.

[14] Zhang, J., Ackerman, M. S., & Adamic, L. (2007, May). Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th international conference on World Wide Web* (pp. 221-230). ACM.

[15] Rafiei, M., & Kardan, A. A. (2015). A novel method for expert finding in online communities based on concept map and PageRank. *Human-centric computing and information sciences*, *5*(1), 10.

[16] Kardan, A., Omidvar, A., & Behzadi, M. (2012). Context based expert finding in online communities using social network analysis. *International J of Computer Science Research and Application*, *2*(1), 79-88.

[17] Wang, G. A., Jiao, J., Abrahams, A. S., Fan, W., & Zhang, Z. (2013). ExpertRank: A topic-aware expert finding algorithm for online knowledge communities. *Decision Support Systems*, *54*(3), 1442-1451.

[18] Zhao, Z., Wei, F., Zhou, M., & Ng, W. (2015, April). Cold-start expert finding in community question answering via graph regularization. In *International Conference on Database Systems for Advanced Applications* (pp. 21-38). Springer International Publishing.

[19] Zhao, Z., Yang, Q., Cai, D., He, X., & Zhuang, Y. (2016, July). Expert Finding for Community-Based Question Answering via Ranking Metric Network Learning. In *IJCAI* (pp. 3000-3006).

[20] Cheng, X., Zhu, S., Chen, G., & Su, S. (2015, November). Exploiting User Feedback for Expert Finding in Community Question Answering. In *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on* (pp. 295-302). IEEE.

[21] El-Korany, A. (2013). Integrated expert recommendation model for online communities. *arXiv preprint arXiv:1311.3394*.

[22] Bozzon, A., Brambilla, M., Ceri, S., Silvestri, M., & Vesci, G. (2013, March). Choosing the right crowd: expert finding in social networks. In *Proceedings of the 16th International Conference on Extending Database Technology* (pp. 637-648). ACM.

[23] Aslay, Ç., O'Hare, N., Aiello, L. M., & Jaimes, A. (2013, July). Competition-based networks for expert finding. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 1033-1036). ACM.

[24] "Stack Overflow API," [online], Available: https://data.stackexchange.com/stackoverflow

[25] "Stack Overflow Documentation,"[online], Available: https://stackoverflow.com/documentation/documentation/topics

APPENDIX A

Code Snippets

```
import java.io.BufferedReader
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Scanner;
import java.util.StringTokenizer;
import jxl.Cell;
import jxl.CellType;
import jxl.Sheet;
import jxl.Workbook;
import jxl.read.biff.BiffException;
import jxl.write.Label;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import jxl.write.WriteException;
import jxl.write.biff.RowsExceededException;
public class recommender
{
  private String inputFile;
  public void setInputFile(String inputFile)
                {
                    this.inputFile = inputFile;
                }
                public void writeexcel(float[] neighbors)
                {
                     int len = neighbors.length;
                     File f = new File("F:/output.xls");
```

```
WritableWorkbook myexcel = null;
        try {
                myexcel = Workbook.createWorkbook(f);
            }
        catch (IOException e1)
            {
                // TODO Auto-generated catch block
                e1.printStackTrace();
             }
 WritableSheet sheet = myexcel.createSheet("output", 0);
          try {
                for(int i = 0; i< len; i++)
             {
                  Label l = new Label(0,i,""+neighbors[i]);
                  sheet.addCell(l);
              }
             }
          catch (RowsExceededException e)
             {
                // TODO Auto-generated catch block
                 e.printStackTrace();
             }
           catch (WriteException e)
             {
               // TODO Auto-generated catch block
               e.printStackTrace();
             }
           finally
             {
          try {
                myexcel.write();
                myexcel.close();
             }
          catch (WriteException e)
```

```
                {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                 }
              catch (IOException e)
                 {
                     // TODO Auto-generated catch block
                      e.printStackTrace();
                 }
                 }
          }
public int[][] read(int matrix[][]) throws IOException
{
       File inputWorkbook = new File(inputFile);
       Workbook w;
       try {
          w = Workbook.getWorkbook(inputWorkbook);
          // Get the first sheet
          Sheet sheet = w.getSheet(0);
          // Loop over first 10 column and lines
          for (int i = 0; i < sheet.getRows(); i++)
            {
             for (int j = 0; j < sheet.getColumns(); j++)
                 {
                   Cell cell = sheet.getCell(j,i);
                   CellType type = cell.getType();
                   if(cell != null){
                   if (type == CellType.NUMBER) {
                   matrix[i][j] = Integer.parseInt(cell.getContents());
                }
           }
       }
      }
      }
```

```
                catch (BiffException e)
                   {
                       e.printStackTrace();
                   }
    return matrix;
   }
public static void main(String[] args)
{
    // TODO Auto-generated method stub
    recommender excel = new recommender();
    excel.setInputFile("F:/user.xls");
    int[][] matrix = new int[31][58];
    System.out.println("Expert Recommendation System!!!");
     try    {
          matrix = excel.read(matrix) ;
          float[] neighbors = new float[29];
          System.out.println("User Pearson");
          Scanner sc = new Scanner(System.in);
          System.out.print("enter the user");
          int user = sc.nextInt();
           neighbors = PearsonSimilarityUser(matrix,user);
          PrintWriter writer = new PrintWriter("result1.csv", "UTF-8");
          for (int i = 0; i < neighbors.length; ++i)
          {
                   System.out.println(neighbors[i]);
          }
           writer.close()
           excel.writeexcel(neighbors);
          }
       catch (IOException e)
       {
          // TODO Auto-generated catch block
           e.printStackTrace();
} }
```

```
public static float[] PearsonSimilarityUser(int[][] matrix, int user)throws
FileNotFoundException, IOException
{
        int lenexpert = matrix.length;
        int lentag = matrix[0].length;
        float[] neighbors = new float[31];
        int expert = 0;
        int tag = 0;
            float correlation = 0;
            for (int j = 0; j < lenexpert; ++j)
            {
                int userthis = 0;
                int userother = 0;
                int count = 0;
                for (int k = 0; k < lentag; ++k)
                {
                    if ((matrix[user][k] != -1) && (matrix[j][k] != -1))
                            {
                                userthis += matrix[user][k];
                                userother += matrix[j][k];
                                count++;
                            }
                 }
          if (count>0)
          float thisavg = (float)userthis/(float)count;
          float otheravg = (float)userother/(float)count;
          float num = 0
          float denom1 = 0;
          float denom2 = 0;
          for (int k = 0; k < lentag; ++k)
                {
                  if ((matrix[0][k] != -1) && (matrix[j][k] != -1))
                   {
                     num += (matrix[0][k] - thisavg)*(matrix[j][k] - otheravg);
```

```
            denom1 += (matrix[0][k] - thisavg)*(matrix[0][k] - thisavg);
            denom2 += (matrix[j][k] - otheravg)*(matrix[j][k] - otheravg);
           }
           }
           if (num > 0)
           {
                 if (denom1*denom2>0)
                    {
                     correlation = (float) (num/(Math.sqrt(denom1*denom2)));
                    }
           }
           }
          neighbors[j]=correlation;
           }
         return neighbors;
}
}
```

APPENDIX B

Top-50 users extracted from Stack Overflow are:

Rank	User name
1	Jon Skeet
2	Darin Dimitrov
3	BalusC
4	Hans Passant
5	VonC
6	Marc Gravell
7	CommonsWare
8	SLaks
9	Martijn Pieters
10	Greg Hewgill
11	Gordon Linoff
12	T.J. Crowder
13	Quentin
14	paxdiablo
15	Alex Martelli
16	dasblinkenlight
17	CMS
18	marc_s
19	Mark Byers
20	JaredPar
21	Ignacio Vazquez-Abrams
22	Guffa
23	Jonathan Leffler
24	Nick Craver
25	David Heffernan
26	Eric Lippert
27	JB Nizet
28	Felix Kling
29	cletus
30	Gumbo
31	unutbu
32	Stephen C

33	Pascal Thivent
34	anubhava
35	BoltClock
36	Reed Copsey
37	Bozho
38	Charles Bailey
39	bobince
40	Peter Lawrey
41	tvanfosson
42	Johannes Schaub - litb
43	Oded
44	Daniel Roseman
45	kennytm
46	Konrad Rudolph
47	Jerry Coffin
48	deceze
49	jfriend00
50	Pekka ì›ƒ

Top-50 users calculated by our Proposed Framework

Rank	User Name
1	Jon Skeet
2	T.J. Crowder
3	VonC
4	Hans Passant
5	Alex Martelli
6	BalusC
7	CMS
8	Eric Lippert
9	Charles Bailey
10	Felix Kling
11	JaredPar
12	Greg Hewgill
13	BoltClock
14	Ignacio Vazquez-
15	Darin Dimitrov
16	unutbu
17	kennytm
18	Gumbo
19	Marc Gravell
20	Quentin

21	Mark Byers
22	Stephen C
23	bobince
24	marc_s
25	Peter Lawrey
26	Johannes Schaub
27	Pekka ì›ƒ
28	JB Nizet
29	Bozho
30	Pascal Thivent
31	Nick Craver
32	cletus
33	tvanfosson
34	Konrad Rudolph
35	Martijn Pieters
36	Guffa
37	paxdiablo
38	Daniel Roseman
39	anubhava
40	SLaks
41	jfriend00
42	Oded
43	David Heffernan
44	Jonathan Leffler
45	CommonsWare
46	Reed Copsey
47	dasblinkenlight
48	Jerry Coffin
49	deceze
50	Gordon Linoff

APPENDIX C

List of Publications

Accepted

1. Kumar A., Bala I. (2017, June).A Comprehensive study of TARS: Definition, Metrics and Advancements. Accepted in UPCON 2017

Communicated

1. Kumar A., Bala I. (2017, July).Expert Recommender System: Using trust metric and collaborative filtering, International Journal of Intelligent Systems and Applications.