International Journal of Computer Applications (0975 – 8887)
Volume 145 – No.6, July 2016

FriendFinder: A Lifestyle based Friend Recommender App for Smart Phone Users

Chinar Bhandari
ME (Computer Engineering),
Jayawantrao Sawant College of Engineering,
Hadapsar, Pune-28, Savitribai Phule Pune
University, Pune, India

M. D. Ingle
Asst. Prof (Computer Engineering),
Jayawantrao Sawant College of Engineering,
Hadapsar, Pune-28, Savitribai Phule Pune
University, Pune, India

ABSTRACT

Today's Social Networking services focuses towards suggesting you friends based on users social graph or Geo-location based, which neither take users life style into account or users liking ,disliking etc. Suggesting friends using the users' link analysis may not be the best preference of suggestion for the users. In this paper, we present FriendFinder, a reliable user relation based friend suggesting app which recommends friend list to app users based on their analysis of life style and daily curricular activities on mobile phone instead of social graphs. FriendFinder captures users data i.e. daily activities and work done through mobile, for ex: App Usage, App Frequency, Browser Activities etc. Then we create a user profile with all gathered data and find most relevant matching profiles of existing candidate friends matching our profile for similarity and suggesting the result out of similarity test to the user as a friend.

Keywords

Friend recommendation, mobile sensing, life style, social networks, app usage, app frequency, browseractivities, categories.

1. INTRODUCTION

Before smart phone even existed, people used to make friends based on people they interacted with in daily life, people who work with them or the one living in the neighbourhood. We name this method of friend making as Geo-Location friends. With the growth of technology, social networking sites started capturing users' social link and recommend friends based on the social link. Now days there are numerous social networking sites which keep your social graph into picture for recommending you friends. For Example: - Face book keeps a track of social link analysis for common friends among users and suggest them friends. But this method for friend suggestion may not be most effective according to the sociology findings. These studies suggest that people should be grouped based on following things:

1. Life Style/Daily Circular Activities.
2. Attitude.
3. Likings/Disliking.
4. Tastes.
5. Moral Values/Standards.
6. People they interact.

Usually Rule 4 or 6 are the point of focus for most of the social networking sites today. But as per the studies, Rule 1 plays an important role in finding perfect similar friend for the user. User's life style is difficult to capture, as life style information can be gathered from daily

circular activities or daily routines. Today people almost spend on an average 9-10 hours with their mobile. It is said that No one knows you better than your mobile. Therefore, we can use the high mobile computation powers to capture user's daily data using their daily activities data

via browsing activities and use these data to recommend friends for them. This recommendation system that we are suggesting can be used as a standalone app or can be used as an extension/framework for existing social networking sites like Face Book. In both cases FriendFinder can help to find them friends like them rather than some strange people based on common friends. Every day, we may do numerous work tasks, which may form a distinguishable category from n-number of activities. Using these categories we can point out likings, disliking of the user to that particular activity. Not necessary every activity a user does, have to be considered to be his liking. So for this reason we also try to capture the frequency at which the activity is done in daily routine. So, from the frequency and using the category ratio we can find the statistical data about the liking of the activity and find a friend to the user who has similar interest in that particular category of the activity. In this particular report, we will make more use of the technical words like activity in context of actions like Browsing, App Usage, App Frequency and refer these activities to categories like Technical, Gaming, Social, Art and Literature. Sports etc. To understand the life style model, we picture an analogy of user's life style, activities and categories as given below:

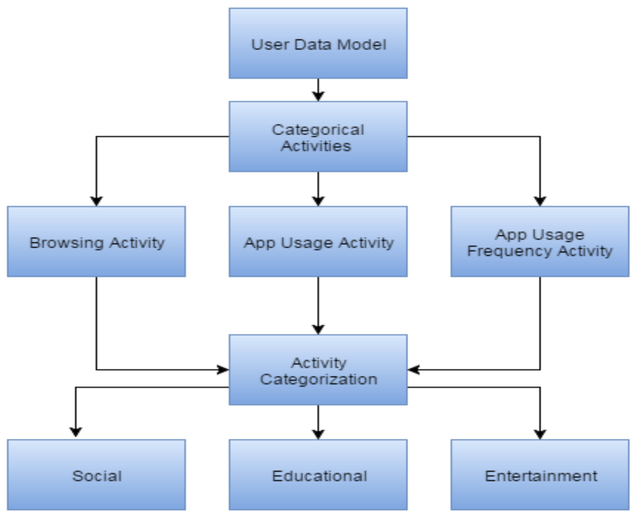


Fig 1: User Lifestyle Model

In the above figure, we try to gather users life style data via daily mobile activities which can be browsing activity and app usage activity, which can be categorized into Technical, Gaming, Social Networking, Literature etc. Our app is inspired by the recent growth and advancements in mobile technology which are acquired by great computational power, high processors etc. Using this sensing capabilities and also by gathering and extracting rich data content aware information, we can create a meaningful profile for the user. So this is the base perceptive for sensing daily circular activity of a user. In spite of great computational powers and sensing capabilities, there exist yet many difficulties for finding users daily data and using it for suggesting potential friends. The important aspect of this work can be categorized as follows: -

1) We capture user's daily mobile activities like browsing history and app usage, app frequency and store the most relevant data out of all noisy data.
2) We compare users profile with other candidate friends profile using category similarity match between profiles.
3) We give category the highest ranking system for profile matching system.
4) We set a timer for capturing user data on a daily basis.
5) We integrate a user's feedback mechanism through which we can improve our recommendation system more effectively.

2. LITERATURE REVIEW

In [1], the author used users work profession or daily activities like walking, shopping, sitting, typing etc. as life style activity. Gathering this data author tries to extract relevant data and using

pattern matching algorithms author recommends candidate friends to the user. But using only professional data may not be the best case to suggest friend. Later, recommender system these days [3], which tries to suggest products like (books, daily use goods, music etc.) to users have become a popular strategy now-a-days. For instance Flipkart recommends item to user based on previous browsing history or previous visited item. Soundwave helps you to discover your music liking based on your most played song types. Netflix [4] tries to suggest movies to user based on users rating system and watching habits. These all recommender system tries to use user's previous history with no frequency rate, to suggest item. This may not be the most useful product for the user, but still gets a recommendation just because he visited or purchased that item previously. Advancement in link analysis [2] networking systems, friend suggestion has received a lot of focus these days. For example: - Social networking sites like Face Book [5], Twitter, LinkedIn recommends friend to user base on social link analysis. But just on basis of common friends you cannot completely or efficient suggest friends to the user. Further the authors from [7] suggested a friend recommender system based on users social and physical context. In this paper author, however did not explain how actually you collect social and physical context data and use to suggest friends. Growing mobile technologies [3] like GPS was further used to collect Geo data. The authors from [8] suggested geographically based friends in link analysis networking by using and collecting GPS data and social analysis. Link recommendation based on weblogs and similar social networks also gained lots of attention these days. The Author in [9] analyzed that link suggestion issues in web link logs and link analysis networks, and suggested a method based on joint recommender making the use of link analysis and link network and data-based recommender using user interests. Activity serves the base purpose for users' life style model development. From high-level daily curricular activity [4], to low-level wireless rich sensor data, which has continuously used for data collection and has got a lot of attention these days. The writer of [6] used collected information using wearable sensor devices to recognize users daily activity based on HMM analysis model.

3. PROBLEM STATEMENT

Despite of having a lot of social networking sites in day to day life, which we use for making new friends but each of them fail to suggest a perfect similar match for the user based on the user's likings or dis-likings. Thus how can a user find a friend based on his life style modelling data?

4. IMPLEMENTATION DETAILS

4.1 Software Specifications

For implementation we have used Eclipse IDE and JDK 1.7 and the app is targeted on Android 4.0 and above.

4.2 Mathematical Model

Set Theory:
Let G be the Global Set of the App

Where G = {U; S; A; Fr; LDB; MR; AO}

1) Where, U is a dataset of all app users using the FriendFinder app and also server dataset and retrieval services from the app server.

And (u1, u2, u3,..un) belongs U where n=infinity.

2) S is the set of the app server dataset. This dataset is used for storing app user profile data like browsing activities, app usage, app frequency, category etc. which is gathered from daily mobile

activities. This data can be collected on go i.e. whenever user is accessing browser or he is active on an app, or he can also update the logs to the database. SDB is basically used for storing and retrieving the user profile logs for query matching.

And (s1, s2, s3,..sk) belongs S where k is not equal infinity.

3) A is the set of attribute used in the application.

And (a1, a2, a3,..ak) belongs A where k is not equal infinity.

4) Fr is the set of suggested friends.

And (Fr1, Fr2, Fr3,..Frn) belongs Fr where n=infinity.

5) LDB is a set of local app dataset that a specific user owns on his device. It consists of stored user credentials, so it can be matched against the entered credentials by the user to verify whether the user is authentic user or not.

And (ldb1, ldb2, ldb3,..ldbk) belongs LDB where k is not equal infinity.

6) MR is a set of retrieval or mining rules that are applied on the input dataset that is captured and stored in server database.

And (mr1, mr2, mr3 ...mrn) belongs MR where n=infinity.

7) AO is a set of associations that are extracted from the input and forms the output of the system.

Functionality or Morphism: -

LDB = Signup (userdetails);

Yes/No (Authentication) = Signin (username, password);

SP= MaintainLocalLog (attributes, details);

S= ExportLogToServer (sharedpreference);

Algorithm= TrainNaiveBayes (normalize-dataset);

Output= ClassifierData (currentuser);

4.3 Proposed System Architecture

In this paper, we use mobile activities to capture user's life style. Mobile activities like Browsing activity and App Usage Activity, App Frequency can be used as a key measure to collect relative data. This activity can be further categorized into low-level abstractions like Gaming, Technical, Social, Arts and Literature, Sports etc. Then we save this data on cloud. When the user seeks to suggest friends for him, we use this gathered data to match his profile with candidate friends profile to find the best similar matching profile based on category ranking. We use Nave Bayes Classifier algorithm for categorical classification .We will study this further using system architecture diagram given below:

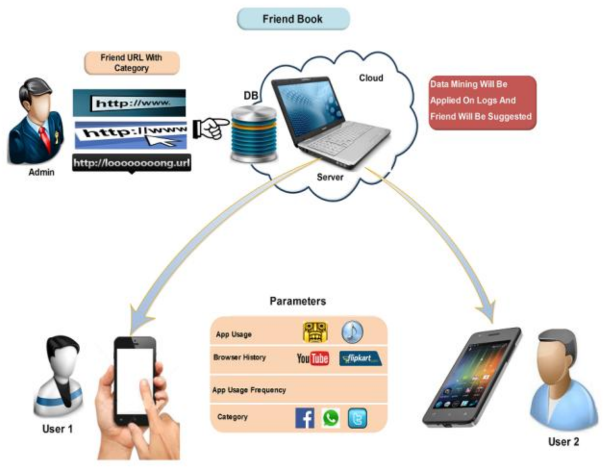


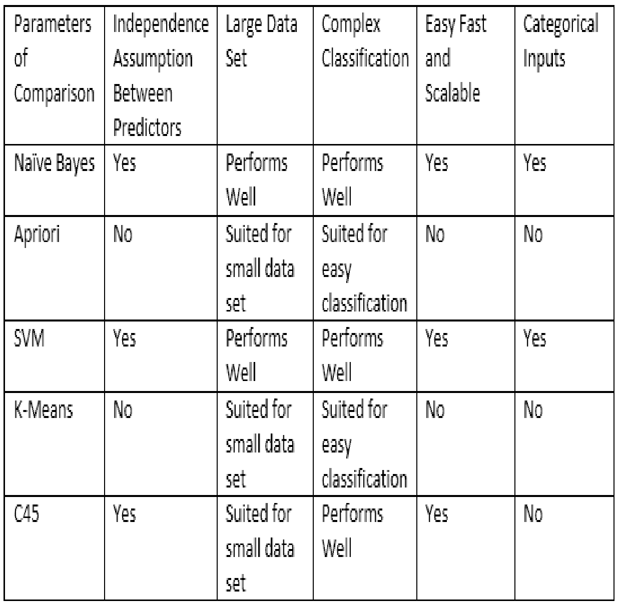
Fig 2: FriendFinder System Architecture

FriendFinder adopts a client-server architecture where each user carrying a mobile phone is a client and the server is cloud server. The above architecture can be explained as follows:

1) First a new user will register to FriendFinder.
2) Secondly he will login to the app with correct username and password.
3) Then the moment the user is logged in, the app will start collecting users mobile activity data like browsing activity and app usage with app usage frequency.
4) This data will be stored to the cloud against the user profile and will be continuously updated whenever user is logged in again.
5) Then the user will seek for friends, asking the app to suggest him some friends, the app will use his stored data which will be pulled from cloud and similarity algorithms will be applied against his profile and candidate friends profiles which are already registered in the app, and profile similarity matching will be done based on ranking system on category.
6) The most relevant matching candidate friends profile will be shown as output of step 5, and will be suggested as friends to the user.
7) The app will also contain discussion forum, wherein any two random users can chat with each other and share pictures or posts about any category. They can also comment on the post.
8) The comments posted in Step 7 is stored by our app and text mining will be applied to the comments regarding whether the user has a liking or a disliking about that category and can be used as a source of data collection.
9) We will also include a feed-back mechanism wherein the user can post a feed-back about friend recommendation and can be used as a measure to improve the algorithm efficiency for suggesting friend.

5. PROPOSED ALGORITHM

We are using Nave Bayes Classifier algorithm instead of other Data Mining algorithms (SVM, Apriori, etc.) for the use of categorical classifications for the following reasons: -



Parameters of Comparison	Independence Assumption Between Predictors	Large Data Set	Complex Classification	Easy Fast and Scalable	Categorical Inputs
Naive Bayes	Yes	Performs Well	Performs Well	Yes	Yes
Apriori	No	Suited for small data set	Suited for easy classification	No	No
SVM	Yes	Performs Well	Performs Well	Yes	Yes
K-Means	No	Suited for small data set	Suited for easy classification	No	No
C45	Yes	Suited for small data set	Performs Well	Yes	No

Fig 3: Naive Bayes vs. Other Algorithms

5.1 Performance Metrics

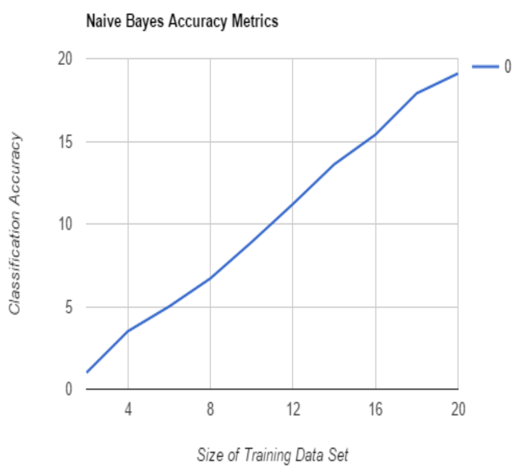


Fig 4: Naive Bayes Accuracy Metrics

5.2 Algorithmic Steps

Algorithm 1: Naive Bayes Classification Steps

Input: log files

Output: Friend Suggestion List

Process:

1. Log Files are gathered from server for current user and provided to algorithm as input.
2. Naive Bayes uses these log files and convert it into normalized form.
3. Conditional Probabilities are calculated out of normalized values and all user matching these conditional values and satisfying the threshold value will be suggested as friend to the user.

5.3 Normalization Techniques and Rules

Naive Bayes training data set parameters and its normalization techniques:

age (0-young, 1-middle age, 2-senior citizen).

gender (0-male, 1-female).

profession (0-student, 1-employee, 2- teacher, 3-businessman).

Further categories will be normalized into its usage Frequency such as (0-low, 1-medium, and 2-high).

Apps and URL categorization (Total=6)

- social.
- educational.
- financial.
- shopping.
- entertainment.
- games.

Rules: -

1) Calculate initial probability for all users.

For Ex: Initial probability u1=6/20.

2) Calculate individual probability for every attribute of current attribute value.

3) Calculate final probability for every user.

4) The highest probability among these all users represent the final Friend Suggestion.

5) Friend Suggestion = (Relevant Intersect Retrieved) / Retrieved.

5.4 Data Security

We are targeting soap based web services for data transfer from client side to server side. Soap based services follows WS-Security standards and is much secured web service. Also the data is stored in digital signature format instead of normal plain text format for additional security purpose.

5.5 Friend Ranking System

The ranking system for Naive Bayes Classifier totally depends on its conditional probability. For Ex: Say User 1 has a conditional probability for category social equal to 0.4 and conditional probability for category educational equal to 0.4 and 0.2 for gaming category, Thus a friend with highest matching conditional probability similar to user 1 and satisfying the threshold value will be suggested as friend to user 1.

5.6 Complexity Analysis

1) As our proposed algorithm lies in P-Complete Class, the app has been proven to lie in P-Complete Class.

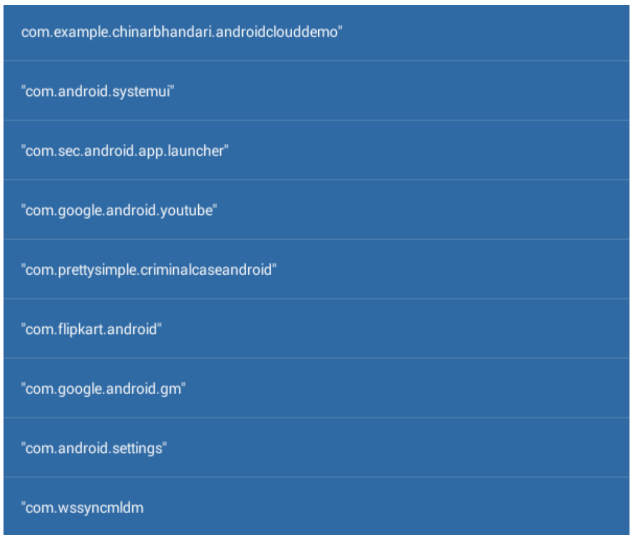
2) Naive Bayes Time Analysis: O(m*n)

6. PERMISSIONS REQUIRED

Some Permissions are required to read the browsing history in Android and to Capture Apps from Stack. Following permission request are made: -

1) Read History Bookmarks.
2) Get System Service.

Following Data is captured while performing test on a single user: -



```
com.example.chinarbhandari.androidclouddemo"
"com.android.systemui"
"com.sec.android.app.launcher"
"com.google.android.youtube"
"com.prettysimple.criminalcaseandroid"
"com.flipkart.android"
"com.google.android.gm"
"com.android.settings"
"com.wssyncmldm
```

Fig 5: User Data Captured

7. ANDROID SERVICES

Data is captured at a certain interval of time using the android services. The service is designed in such a way that the service process is called every 3 minutes and the check for data is been done. Following algorithm is applied for working: -

1) Service is called after every 3 minutes.

2) Data is stored in Shared preference first in case there is network unavailability.

3) Next time the service is called, first the old data and new data is checked for any changes, if found then only the changes are saved and the data is pushed to the server.

4) If no change is found then no action is taken.

7.1 App and Browsing Categorization Technique

App categorization is done at client side because the stack traces are not saved for a longer time at Android OS Level. So the app categorization is done immediately once the data is captured from the stack trace. Following algorithm is applied for categorization: -

1) Package Name is read from Stack Trace.
2) If the package name matches the saved list, normalization technique is used to find the uid for that category.
3) If the package name is not matching, the package is simply discarded.
4) Final Data is pushed to Server with category uid.

The browsing data is categorized at server side as the data is saved at the browser history for a long time. Following algorithm is applied for categorization: -

1) Browsing data is read in url format.
2) If the url matches the saved url's, the normalization technique is used to find the uid for that category.
3) If the url is not matching, the url is simply discarded.
4) Final Data is pushed to Server with category uid.

8. RESULTS AND DISCUSSIONS

8.1 Accuracy Table

Following table 1 shows the results obtained for accuracy of FriendFinder and compared with FriendBook

Table 1 FriendFinder vs. FriendBook

Friend Suggestion Apps	FriendBook	FriendFinder
Percentage Accuracy Ratio	89%	94%
Time Ratio	125ms	110ms

Following table 2 shows the results calculated for FriendFinder.

Table 2 Relevant vs. Non-Relevant

Factor	Formulae	Data
Relevant Suggestions	Relevant Retrieved/Retrieved	17/18=0.94
Non-Relevant Suggestions	Non-Relevant Retrieved/Retrieved	110ms

8.2 Test Data

It is planned to test the user profile against all pre-saved profiles in the server database which contains profiles of all categories, so it will be easy to test the efficiency and accuracy of the proposed algorithm.

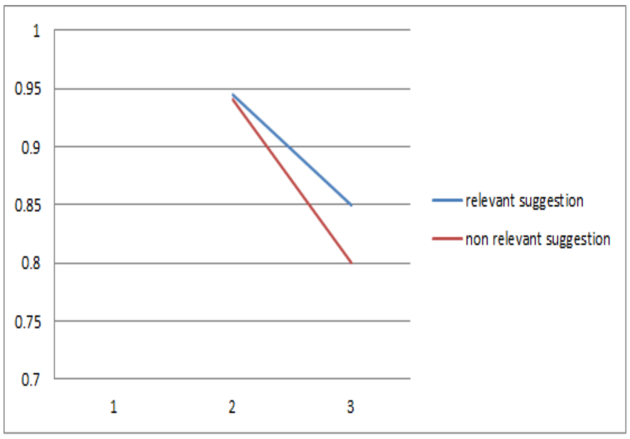


Fig 6: Relevant vs. Non-Relevant

8.3 Contributory Work

1) The app can be used as an add-on service in Facebook to capture user's data to suggest them friends.
2) It can be used as user wish list capturing service in online recommender systems like (Flipkart, Amazon) to capture users preference data to show their favourite products when they visit the app.

9. CONCLUSION AND FUTURE SCOPE

This paper helps you to get deep information on how link analysis and social content and recommender system has grown rapidly these years. Each previous method tries to use some link analysis to get users attention, which comes with some disadvantages like depending on social link analysis and geo-location in existing link network. Thus, we present you FriendFinder a novel semantic based friend finder, which uses users daily activities data gathered from mobile phone to recommend friends to the user in a more precise and accurate manner.

In future, we will also try to implement dynamic browsing activity records and dynamic app usage records so, the static dependency gets resolved.

10. REFERENCES

[1] Zhibo Wang, Jilong Liao, Qing Cao, Hairong Qi, and Zhi Wan, "Friendbook: A Semantic-based Friend Recommendation System for Social Networks", pages 1-14, 2015.

[2] Liang Hu, Guohang Song, Zhenzhen Xie, and Kuo Zhao, "Personalized Recommendation Algorithm Based on Preference Features", pages 293-299, 2014.

[3] Yongkun Li, and John C. S. Lui, "Friends or Foes: Distributed and Randomized Algorithms to Determine Dishonest Recommenders in Online Social Networks", pages 1695-1708, 2014.

[4] Malmaz Roshanaei, Shivakant Mishra,"An Analysis of Positivity and Negativity Attributes of Users in Twitter", pages 1-6, 2014.

[5] Shunmei Meng, Wanchun Dou, Xuyun Zhang and Jinn Chen,"KASR: A Keyword-Aware Service Recommendation Method on MapReduce for Big Data Applications", pages 1-11, 2013.

[6] Face book statistics. http://www.digitalbuzzblog.com/ facebook-statistics-stats-facts-2011/.

[7] L. Bian and H. Holtzman. Online friend recommendation through personality matching and collaborative filtering. Proc. of UBICOMM, pages 230-235, 2011.

[8] J. Kwon and S. Kim. Friend recommendation method using physical and social context. International Journal of Computer Science and Network Security, 10(11):116-120, 2010.

[9] X. Yu, A Pan, L.-A. Tang, Z. Li, and J. Han. Geo-friends recommendation in gps-based cyber-physical social network. Proc. Of ASONAM, pages 361-368, 2011. query and post query approaches for classifying deep-web forms to further improve the accuracy of the form classifier.

11. AUTHOR PROFILE

Mr. Chinar C. Bhandari, is currently pursuing M.E (Computer) from Department of Computer Engineering, Jayawantrao Sawant College of Engineering, Pune, India. Savitribai Phule Pune University, Pune, Maharashtra, India - 411007. He received his B.E (Computer) Degree from AISSMS IOIT, Pune, India. Savitribai Phule Pune University, Pune, Maharashtra, India -411007. His area of interest is mobile computing, web mining.

Prof. M.D Ingle, received his M Tech. (Computer) Degree from Dr. BabasahebAmbedkar Technological University, Lonere, Dist. Raigad-402 103, Maharashtra, India. He received his B.E (Computer) Degree from Govt college of Engineering, Aurangabad, Maharashtra, India.He is currently working as M.E coordinator and Asst Prof (Computer) at Department of Computer Engineering, JayawantraoSawant College of Engineering, Pune, India. SavitribaiPhule Pune University, Pune, Maharashtra, India -411007. His area of interest is network security and web&data mining.