CLASSIFICATION OF MESSAGES USING ASSOCIATION RULE MINING FOR DYNAMIC MESSAGE APP IN ANDROID ENVIRONMENT

Dr. S. Sabeen
Department of Computer Science,
SRM Institute of Science and Technology, Chennai, India.

Dr. R. Arunadevi
Head & Associate Professor, Department of Computer Science,
Vidhya Sagar Women's College Chengalpettu,
Tamilnadu, India

Abstract

"Dynamic Message App" is an application created dynamically which is used to share notices, messages and schedules of meetings and various other messages which can be shared among users who are using these applications. It will act as a co0mmon screen for users who can share and read messages of all who are using this application. The Dynamic Message App system (DMAS) is developed from the scratch and therefore there is no existing system and no data migration is done. In this paper a dynamic message app system developed which displays the information on any kind of display (as notice board) when any authorized user sends the notice message from his mobile phone. A person can use this application by registering and the details will be stored in the back end for future references. The authorized users who are using this application can use this notice board for writing notices and thus other users can read it and come to know about the happenings. The main objective of this system is to send messages to common screen which has to be displayed drastically and dynamically after classification of messages. This system contains three main parts: First, creation of a dynamic notice board in which the user can register him/her by providing the required credentials. Then the registered user with his unique user ID and password can login, and can send notices/ messages to the screen. In this paper a new method is proposed for classifying text using the association rule mining using digraph model introduced in chapter 3. Association rule mining technique is used to derive feature set from pre-classified text documents. Naïve Bayesian classifier is then used on derived features for final classification. Finally the messages will be displayed dynamically and can be visualized only by other authenticated users.

Key words: Dynamic Message App, android, data mining, directed graph, path systems, frequent item set, association rule, classification.

1. Introduction

Dynamic Message App has been developed in Android environment keeping the requirements in today's world. Nowadays, most of the mobile applications are developed using Android. Dynamic Message App is a common notice board used to share notices, messages and schedules of meeting and various other messages which can be shared among users who are using these applications. Android is an Operating System for mobile devices and is built over a Linux Kernel. It was initially developed by Android INC and later acquired by Google. Currently Android is held by OHA (Open Handset Alliance). Android allows developers to write their own application using various API's and Libraries. The Android SDK provides the tools and API's necessary to begin developing applications on the Android platform using the Java Programming Language.

2. **Creation of Dynamic Message App using Android Environment**

Android is an Open Source software platform for mobile phone application development. It provides the opportunity to create mobile phone applications and the user-interface. It is built on an open source framework, and feature powerful SDK libraries. Android provides several options to save persistent application data and is a software stack for mobile devices that includes an operating system, middleware and key applications. Android will ship with a set of core applications including an email client, SMS program, calendar, maps, browser, contacts, and others.

2.1 Hardware Specification

We conducted the experiments on the system having the following hardware configuration and software configuration

Processor	:	200MHz Online Processor
RAM	:	64 MB RAM
HDD	:	Minimum/80 GB
Web server	:	The local network of Wi-Fi connection
Database Server	:	SQL Plus

2.2 Software Specification

Operating System for Web server	:	Windows, Linux 32/64-bit (x86) etc.
Operating System for Database Server	:	Windows, Linux 32/64-bit (x86) etc.
IDE	:	Eclipse IDE with ADT
DBMS	:	Android DBMS 0.9b version, 0.8b also

Third Party S/W : NA

2.3 Front End – Android

Android is an Open source software platform for mobile phone application development. It provides the opportunity to create mobile phone applications and the user-interface provided by it is better when compared to other technologies. It is built on an open source framework, and features powerful SDK libraries. Android applications are written in the Java programming language. The Android SDK tools compile the code—along with any data and resource files—into an Android package, an archive file with an .apk suffix. All the code in a single .apk file is considered to be one application and is the file that Android-powered devices use to install the application. Android provides several options for you to save persistent application data. The solution you choose depends on your specific needs, such as whether the data should be private to your application or accessible to other applications (and the user) and how much space your data requires. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. Android will ship with a set of core applications including an email client, SMS program, calendar, maps, browser, contacts, and others. All the applications are developed using the java programming language. By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more. The design representation used for designing the front-end of the dynamic notice board is given in the figure 1.

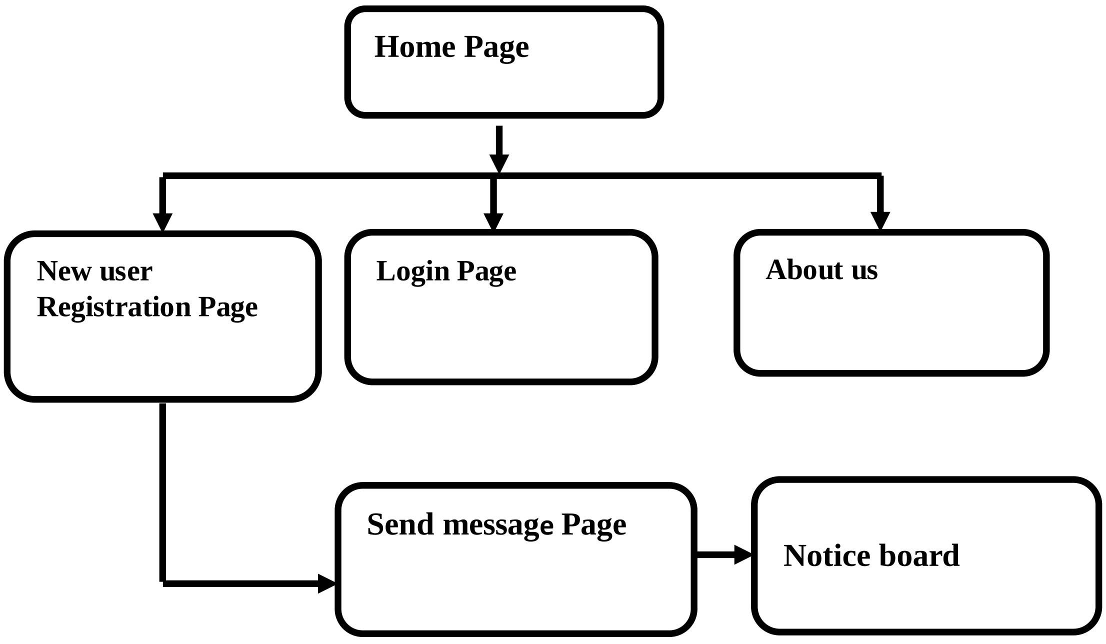

Figure1 Front-end design for Dynamic Message App

2.4 Back End - SQLite Database

As there is no separate mechanism to create and manage the databases in Android, we have to create database by writing our own code.

SQLite is an Open Source lightweight transactional database embedded into Android. It supports SQL syntax. One of the features of SQLite is that, it requires only little memory at runtime (approximately 250 kilobytes), so it's a perfect choice for creating databases on many mobile operating systems. SQLite supports the data types like TEXT, INTEGER and REAL which are similar to String, long and double respectively in Java. For using SQLite, no configuration of database is required. Each SQLite database is stored within a single file on disk. Android SDK provides two classes to work with databases.

- SQLiteOpenHelper-is used for creating, opening and upgrading databases
- SQLiteDatabase –is used for communicating data changes

SQLiteOpenHelper class provides two methods:

- onCreate(SQLiteDatabasedb): Invoked when database is created.
- onUpgrade(SQLiteDatabasedb, intoldVersion, intnewVersion): Invoked when database is modified

Important Features of SQLite Database are stated below

- Free to develop, deploy, and distribute.
- Lightweight database as there is no client server concept.
- Multilingual support is there.
- No limitation on data size.
- Performance is good as there is single database file that is managed across the database.
- Can be used on laptops as well as PDA devices.

Limitations of SQLite Database are given below.

- Only LEFT OUTER JOIN is supported, but no support for RIGHT OUTER JOIN or FULL OUTER JOIN
- There is one data file for the whole database that is the cause of following limitations
- Single point of access implicitly decrees a single point of failure and a single point of performance degradation.

- Achieving concurrency is not possible.
- In case of data file corruption, the recovery is very difficult. FOREIGN KEY constraints are parsed but are not enforced

2.5 Creation of SQLite Database

Database design is required to manage the larger bodies of information. The management of data involves both the definition of structure of the storage of information and provisions of mechanism for the manipulation of information. The ContentValues stores set of values that content Resolver can process. The Context which is the abstract implementation of Android services stores the global information about the application environment. It allows access of application- specific resources and classes, as well as up-calls for application level operations such as launching activities, broadcasting and receiving intents etc., the cursor exposes a result of a query on SQLite database. The SQLException as Nicolas (2009) is the exception that indicates that there was an error with SQL parsing or execution. The methods are included to manage SQL database to create, delete, and execute database management tasks. SQLiteOpenHelper class is used to manage database creation and version management and the tables are maintained to store information in the system for the future reference and manipulation. In Dynamic Notice Board, table1 is used for registration to store the user name, password, email Id, location of the user and provide them authorization to use the dynamic notice board. The table1 is maintained to store information in the system for the future reference and manipulation.

Table.1: Registration Table

S. No.	**Field Name**	**Description**	**Size**	**Type**	**Optional/ Mandatory**	**Default Values**
1	UserName	Name to be entered by use	25	Varchar2	Mandatory	User name
2	Password	Unique key for authentication	25	Varchar2	Mandatory	Password
3	Email id	Email id of the user	50	Varchar2	Mandatory	Email id
4	Location	Current Residential address of user	75	Varchar2	Mandatory	Location

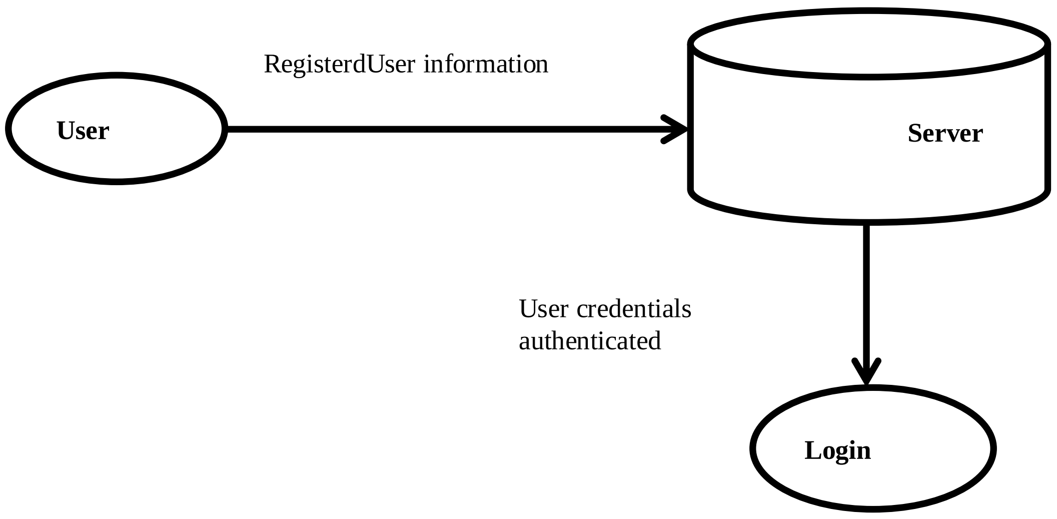

Figure2: Back End for Dynamic Message App

3 DESIGN AND CONSTRUCTION OF DYNAMIC MESSAGE APP

Design is the first step in moving from problem domain to solution domain. Design is essentially the bridge between requirements specification and final solution. The goal of the design process is to produce a model or representation of a system, which can be used later to build that system. The produced model is called the “Design of the Dynamic Message App” given in figure 3. It is a representation of the architecture of dynamic notice board system.

The design of dynamic notice board is the most creative and challenging phase in system development life cycle, which helps the coding this system. It provides understanding and procedural details necessary for implementing this system. A number of elements of dynamic message system are to be identified which constitutes the whole system.

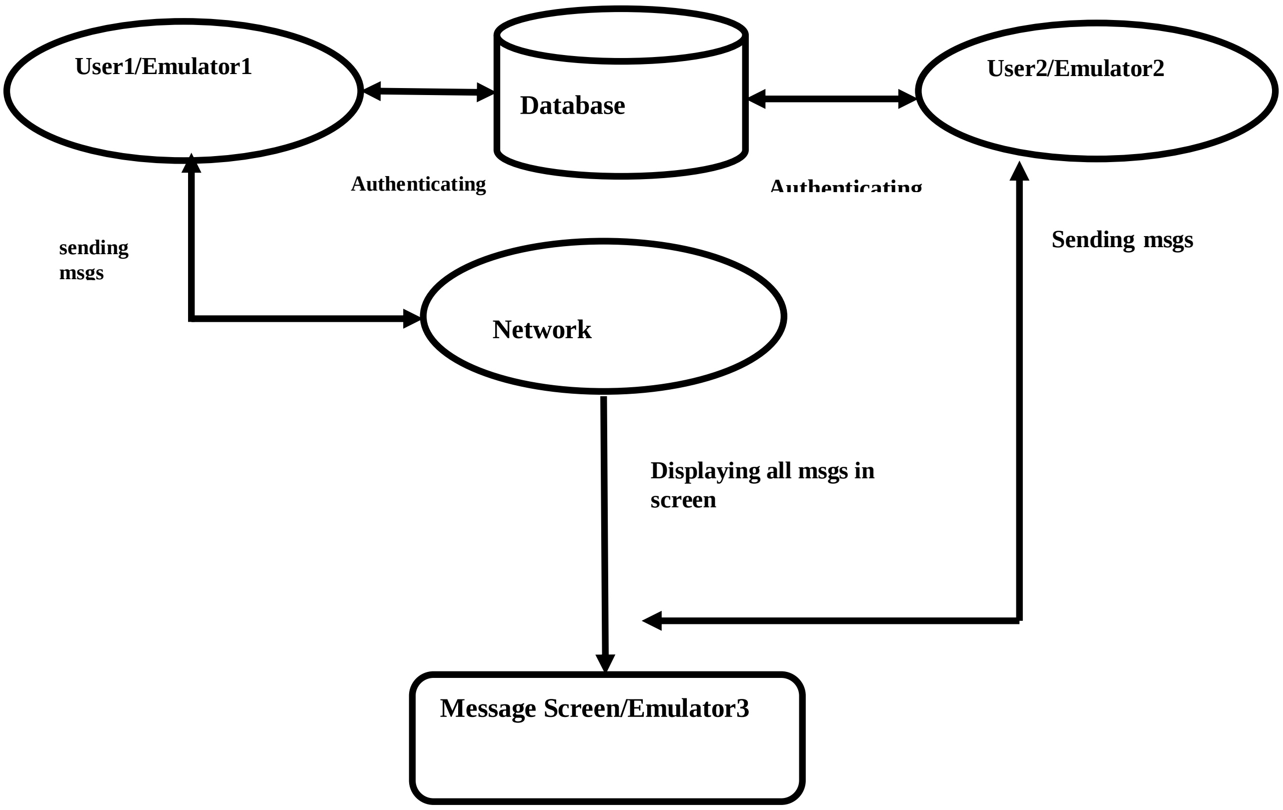

Figure.3: Architecture of Dynamic Message App

4 DYNAMIC MESSAGE APP

The main objective of this system is to dynamic message app which displays the information on any kind of display (as notice board) when the authorized user send the notice message from his mobile phone. The message app is designed to accept messages from multiple users. The users must be authorized users. Authorization is provided to the users by registering themselves to the database through the registration form of the application. Unauthorized users are not allowed to send notices to the notice board. Validations are done the user details while registering to the application. After successful registration the users are allowed to login to the application to view the notice board or to send messages to the notice board.

A message app is a surface intended for the posting of public messages, for example, to advertise items wanted or for sale, announce events, scheduling of meetings or provide information. Message boards are sometimes referred to as notice boards which help people to know about various messages which are in their interests. As we are in the era of technology mobile phones, tablets have become an important device to all. So we are making this research project using Android technology to bring people closer and share the news which they come to know. With Notice board people can send notices which can be viewed by other users. Today in this busy-fast world we all are so busy that often we forget things which are important or we will be unknown about so many things which we should know. So there is a better option of showing all this things in a hand held device. Moreover the news which we know can be shared with other just by using the mobile phone. This all is possible and easy by using a Notice board application which enables users to perform all this at ease. The notices sent by all users in this application are displayed in one common screen which is called the Dynamic Notice board system. Notice board serves as a display medium where people can view all events and notices posted by other users.

There are six components designed and used in our dynamic notice board system. They are Home, Login, Registration/Sign up, Send Notice, Notice board module, About Us module.

4.1Home Page for Dynamic Message App

The home page contains the link for existing users to login and registration page for new users and also it contains the link stating the description about this application in the form of About Us. It also contains an exit link to exit from the application. Figure4 shows the home page of dynamic message app

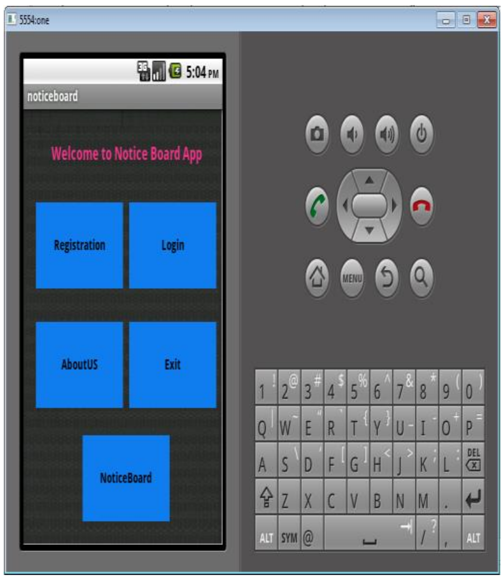

Figure 4 Home page for dynamic message app

5 CLASSIFICATIONS OF MESSAGES USING ASSOCIATION RULE MINING

As the amount of online text increases, the demand for text classification to aid the analysis and management of text is increasing. Text is cheap, but information, in the form of knowing what classes a text belongs to, is expensive. Automatic classification of text can provide this information at low cost. Classification is one of the most important tasks in data mining. There are many classification approaches for extracting knowledge from data such as statistical John (1995), divide-and-conquer Furnkranz, J. (1996) and covering Cendrowska, J. (1987) approaches.

Text databases are databases that contain word descriptions for objects. These word descriptions are usually not simple keywords but rather long sentences or paragraphs, such as product specifications, error or bug reports, warning messages, summary reports, notes, or other documents. The widely used and well-known data mining functionalities are Characterization and Discrimination, content based analysis as in Hayes(1990), Association Analysis, Classification and Prediction adopted in literature Han,(2001), Cluster Analysis Lewis (1990), Outlier Analysis, Evolution Analysis. For our text classification purpose we have used Association Analysis for generating associative word sets.

In this paper we use our data structure introduced in our previous work Arumugam, S., **Sabeen**,S(2013) consisting of a directed graph *D* and a multiset of directed paths in *D* to represent a database of transactions. Let us consider a set of transaction where each document is considered as a transaction. We introduced an algorithm which generates the directed graph *D* and which also simultaneously computes several other measures such as in-degree, out-degree, total number of arcs, length of a largest transaction, frequency of occurrence of various nodes and the number of occurrences of each arc in *D*. The second algorithm generates all the frequent patterns of word set in transaction data base using the directed graph model of the text database We proposed an algorithm, *DGTDB* to construct the directed graph representing a TDB in chapter3. This algorithm scans the data exactly once, dynamically constructs the digraph *D* and simultaneously computes several parameters such as frequency of occurrence of each node, number of loops at each node, number of occurrence of each arc *uv*, total number of arcs in *D*, the maximum length of a transaction, in-degree and out-degree of each node.

The *DGTDB* algorithm first creates all nodes of *D*, one node for each item, with support count 0. Then each transaction is scanned and the directed path in D representing the transaction is constructed. If $(i_1, i_2, \ldots, i_k)$ is a transaction representing a document, the arc $(ij\ ,\ ij+1)$ is represented as a linked list. The header of this list has two fields. One field is used to store the list of vertices $(i_1, i_2, \ldots, i_{j+1})$ which is called the label of the arc (i, j) and the other field is used to store the frequency of occurrence of the arc (i, j).

In this section we use an algorithm Arumugam, S., Sabeen,S(2013) for extracting the set *L* of all frequent word set from *DGTDB*. For each node *i*, the algorithm generates all frequent word set L_1 with *i* as the last word of L_1. If In-Edge (*i*) = 0, then $\{i\}$ is the only pattern with *i* as the last item. Otherwise for each edge e_c with *i* as head we consider all subsets *X* of *Label*(e_c, *i*) such that $|X| \geq 2$ and $i \in X$. Then the frequency of *X* is the sum of the frequencies of all the edges e_c with *i* as head for which $X \subseteq Label(e_c, i)$. If this frequency is greater than or equal to the minimum support threshold, *s* then *X* is added to the set of frequent patterns. The algorithm for extracting frequent pattern for a DGTDB is XoFP is explained in our previous work Arumugam, S., **Sabeen**,S(2013)

Association rule mining is one of the most important tasks in data mining. It is considered a strong tool for market basket analysis that aims to investigate the shopping behavior of customers in hoping to find regularities Agrawal (1993). In finding association rules, one tries to find group of items that are frequently sold together in order to infer items from the presence of other items in the customer’s shopping cart. For instance, an association rule may state that “ 80% of customers who buy diaper and ice also buy cereal”. This kind of information may be beneficial and can be used for strategic decisions like items shelving, target marketing, sale promotions and discount strategies. Association rules is a valuable tool that has been used widely in various industries like supermarkets, mail ordering, telemarketing, insurance fraud, and many other applications where finding regularities are targeted.

For the transactional database *D*, the association rule problem is to find all rules that have a support and confidence greater than certain user. Specified thresholds, denoted by *minsupp* and *minconf*, respectively. The problem of generating all association rules from a transactional database can be decomposed into two sub problems as in Agrawal (1993).
1. The generation of all word set with support greater than the *minsupp*. These word sets are called *frequent* wordsets. All other word sets are called *infrequent*.
2. For each frequent word set generated in step1, generate all rules that pass *minconf* threshold. For example if itemset {X,Y,Z} is frequent, then we might evaluate the confidence of rules $XY \Rightarrow Z$, $XZ \Rightarrow Y$ and $YZ \Rightarrow X$.

For clarity, consider for example the database shown below in Table 2, and let *minsupp* and *minconf* be 0.70 and 1.0, respectively. The frequent itemsets in Table 1 are {bread}, {milk}, {juice}, {bread, milk} and {bread, juice}. The association rules that pass *minconf* among these frequent itemsets are $juice \Rightarrow bread$ and $milk \Rightarrow bread$.

Table 2 a TDB from reliance shop

Tid	item
1	bread, milk, juice
2	bread, juice, milk
3	milk, ice, bread, juice
4	bread, eggs, milk
5	ice, basket, bread, juice

The second step of association rule discovery that involves generation of the rules is considerably a straightforward problem given that the frequent word set and their *support* are known, adopted in literature Hipp (2000). We can use the algorithm, GEAR for generating associative word set.

The first step of finding frequent itemsets is relatively a resource consuming problem that requires extensive computation and large resource capacity especially if the size of the database and the itemsets are large as in Blackmore (2003). Generally, for a number of distinct items *m* in a customer transaction database *D*, there are 2^m possible number of itemsets. Consider for example a Reliance fresh shop that contains 2500 different distinct items. Then there are 2^{2500} possible different combinations of potential frequent itemsets, known by candidate itemsets, in which some of them do not appear even once in the database, and thus usually only a small subset of this large number of candidate itemsets is frequent.

6 IMPLEMENTATION OF ASSOCIATION RULE MINING ON TEXT DATA

Let us consider a set of transaction where each document is considered as a transaction as follows:

1. algorithm, network, graph, multicast, processor, system, parallel
2. cluster, network, design, message, processor, system, framework
3. algorithm, software, graph, method, session, analysis, parallel
4. switch, load, design, power, path, system, timing
5. cable, load, energy, power, current, motor, signal
After implementation of the Association rule (considering minimum support as 0.4 & confidence 1) we will get,
a. {algorithm, graph} $\Rightarrow$ **{parallel}** from 1 , 3
b. {network, processor} $\Rightarrow$ **{system}** from 1, 2
c. {design} $\Rightarrow$ **{system}** from 2 , 4
d. {load} $\Rightarrow$ **{power}** from 4 , 5

Abstracts from different thesis, research papers are considered as training document for developing a model for classifying new documents of unknown class. Most of the papers are collected from World Wide Web. Three categories of papers from Computer Science, Electrical and Electronic Engineering and Mechanical Engineering are considered as training documents.

6.1 Implementation of Association Rule Mining on Text Data

Let us consider a set of transaction where each document is considered as a transaction as follows:
1. algorithm, network, graph, multicast, processor, system, parallel
2. cluster, network, design, message, processor, system, framework
3. algorithm, software, graph, method, session, analysis, parallel
4. switch, load, design, power, path, system, timing
5. cable, load, energy, power, current, motor, signal

After implementation of the Association rule (considering minimum support as 0.4 & confidence 1) we will get,
a. {algorithm, graph} $\Rightarrow$ {parallel} from 1 , 3
b. {network, processor} $\Rightarrow$ {system} from 1, 2
c. {design} $\Rightarrow$ {system} from 2 , 4
d. {load} $\Rightarrow${power} from 4 , 5

Abstracts from different thesis, research papers are considered as training document for developing a model for classifying new documents of unknown class. Most of the papers are collected from World Wide Web. Three categories of papers from Computer Science, Electrical and Electronic Engineering and Mechanical Engineering are considered as training documents.

7. Applying Naïve Bayes Theorem in Classification

Before classifying a new document of the text data (abstract), target class of which is to be determined, it is again preprocessed by a process similar to that applied to training data. The steps for preprocessing and classifying a new document can be summarized as follows:

1. Remove periods, commas, punctuation, stop words. Collect words that have occurrence frequency more than once in the document.
2. View the frequent words as word sets.
3. Search for matching word set(s) or its subset (containing items more than one) in the list of word sets collected from training data with that of subset(s) (containing items more than one) of frequent word set of new document.
4. Collect the corresponding probability values of matched word set(s) for each target class.
5. Calculate the probability values for each target class from Naïve Bayes classification theorem.

Following the steps mentioned above, we can determine the target class of a new document. We give an example to illustrate the whole process. Consider the following text (abstract) which can be any one of the categories of Computer Science, Electrical and Electronic Engineering or Mechanical Engineering, adopter in the literature Chowdhury (2010).

"This paper discusses feedback control problems like regularization, non interaction and linearization, for affine nonlinear singular systems. First, based on the constrained dynamic algorithm in affine nonlinear systems, an algorithm is introduced. By using such an algorithm, sufficient and necessary conditions are derived for the solvability of regularization problem. Then, another algorithm is proposed, based on which a sequence of integers can be defined for the system. It is shown that under some mild conditions, the dynamic part of singular systems can be linearized by using a regular feedback. Finally, an example is provided to illustrate the main results".

After preprocessing the above text we have found the following frequent words:
{feedback, problem, regularization, affine, nonlinear, singular, system, based, dynamic, algorithm, using, condition }

Now search for word set(s) or its subset(s) from the list of word sets in Table: 5 matching with subset of frequent word set of new document. The following probability values in different categories are found accordingly.

Table 3 Probability values in different category

Matched Word Set from Training data	**S**	**E**	**E**
{problem, graph, algorithm }	.027	.0067	.0067
{irregular, multicast, algorithm, system }	.02	.0067	.0067
{algorithm, message-passing, multicast, system }	.02	.0067	.0067
{dynamic, system, interaction}	.0065	.019	.006
{multidestination, based, multicast, system	.02	.0067	.0067
{using, parameter, system }	.0065	.019	.0065
{ condition, algorithm }	.02	.0067	.0067

Table 4: Word set with occurrence frequency

Large Word set found	**Number of Occurrence in Document**		
	CS	**EE**	**ME**
Graph, algorithm	5		
Technology, processor, system	4		
Design, system	4		
Message-passing, system	4		
Oscillation, system, power, model		3	
Distribution, load, feeder, system		3	
Multicast, message-passing, system	3		
Destination, multicast, approach	3		
System, result, model		3	
Power, control, system		3	
Problem, graph, algorithm	3		
Message, communication, system	3		
Stability, system, power		3	
Multidestination, message-passing, system	3		
Customer, feeder		3	
Instability, experiment			3
Virtual, routing	3		

Device, power		3	
Block, power		3	
Voltage, power		3	
Shear, stress			3
Generator, test		3	
Current, signal		3	
Stability, control, system, power, model, strategy, device, oscillation		2	
Change, distribution, system, load, customer, temperature, feeder		2	
Pinout, framework, processor, technology, system, design	2		
Approach, message-passing, multicast, destination, system	2		
Broadcast, message, multicast, approach, destination	2		
Distribution, power, system, load, feeder		2	
Multidestination, communication, message, system, message-passing	2		
Power, damping, model, oscillation, system		2	
Irregular, multicast, algorithm, system	2		
Algorithm, message-passing, multicast, system	2		
Effect, system, power, load		2	
Multicast, network, message, algorithm	2		

Matching subsets from frequent words of new document to be considered for probability calculation are:

1.{algorithm, problem}	2.{algorithm,system}
3.{dynamic, system}	4.{system, based}
5.{system, using}	6.{algorithm, condition}

Table 5: Word set with probability value

Large WordSet	**S**	**E**	**E**
Graph, algorithm	.04	.0067	.0067
Technology, processor, system	.033	.0067	.0067
Design, system	.033	.0067	.0067
Message-passing, system	.033	.0067	.0067
Oscillation, system, power, model	.0065	.026	.0065
Distribution, load, feeder, system	.0065	.026	.0065
Multicast, message-passing, system	.027	.0067	.0067
Destination, multicast, approach	.027	.0067	.0067
System, result, model	.0065	.026	.0065
Power, control, system	.0065	.026	.0065
Problem, graph, algorithm	.027	.0067	.0067
Message, communication, system	.027	.0067	.0067
Stability, system, power	.0065	.026	.0065
Multidestination, message-passing, system	.027	.0067	.0067
Customer, feeder	.0065	.026	.0065
Instability, experiment	.0079	.0079	.031
Virtual, routing	.027	.0067	.0067
Device, power	.0065	.026	.0065
Block, power	.0065	.026	.0065
Voltage, power	.0065	.026	.0065
Shear, stress	.0079	.0079	.031
Generator, test	.0065	.026	.0065
Current, signal	.0065	.026	.0065
Stability, control, system, power, model, strategy, device,	.0065	.019	.0065
Oscillation	.0065	.019	.0065
Change, distribution, system, load, customer, temperature, feeder	.02	.0067	.0067
Pinout, framework, processor, technology, system, design	.02	.0067	.0067
Approach, message-passing, multicast, destination, system	.02	.0067	.0067
Broadcast, message, multicast, approach, destination	.0065	.019	.0065

The prior probability and probability values of word sets calculated using Naïve Bayes equation are:

Prior probability ***P***(CS) = **0.40**, ***P***(EE) = **0.44**, ***P***(ME) = **0.16**
P({algorithm,problem}|CS)=**0.027**, ***P***({algorithm,problem}|EE)=**0.0067**,
P({algorithm,problem} |ME)=**0.0067,** ***P***({algorithm,condition}|CS)=**0.02**, ***P***({algorithm,condition}|EE)=**0.0067**,
P({algorithm,condition} |ME)=**0.0067**
P({algorithm,system}|CS)=**0.02**, ***P***({algorithm,system}|EE)=**0.0067**,
P({algorithm,system}|ME)=**0.0067** ***P***({dynamic,system}|CS)=**0.0065**,
P({dynamic,system}|EE)=**0.019**, ***P***({dynamic,system |ME)=**0.0065**
P({based,system}|CS)=**0.02**, ***P***({based,system}|EE)=**0.0067**,
P({based,system} |ME)=**0.0067** ***P***({using,system}|CS)=**0.0065**,
P({using,system}|EE)=**0.019**, ***P***({using,system}|ME)=**0.0065**

***For* Computer Science** = 0.4X0.027X0.027X0.02X0.0065X0.02X0.0065
= **0.00000000000492804**
***For* Electrical & Electronic** = 0.44X0.0067X0.0067X0.0067X0.019X0.0067X0.019
= 0.000000000000320080405964
***For* Mechanical** = 0.16X0.0067X0.0067X0.0067X0.0065X0.0067X0.0065
= 0.0000000000000013622157796

From the above result we found the document classified as **Computer Science**.

7 .1 Receiving the Text and Displaying the Messages in Dynamic Message App.

To listen for incoming SMS messages, BroadcastReceiver class is created. The Broadcast Receiver class enables the application to receive intents sent by other applications using the sendBroadcast() method. Essentially, it enables the application to handle events raised by other applications. When an intent is received, the OnReceive() method is called; hence, we need to override this. When an incoming SMS message is received, the onReceive() method is fired. The SMS message is contained in the Intent object via a Bundle object. The messages are srored in an Object array in the PDU format Coder's(2008). To extract each message, we use the static createFromPdu() method from the SmsMessage class. The SMS message is then displayed using the Toast class. The phone number of the sender is obtained via the getOriginatingAddress() method, Darcey (2012), so if it is needed to send an auto reply to the sender, this is the method to obtain the sender's phone number. One interesting characteristic of the BroadcastReceiver is that we can continue to listen for incoming SMS messages even if the application is not running; as long as the application is installed on the device, any incoming SMS messages will be received by the application.

7.2 Updating an Activity from a Broadcast Receiver

The previous section described how Broadcast Receiver class is used to listen for the incoming SMS messages and then use the Toast class to display the received SMS message. Often, we want to send the SMS message back to the main activity of the application. For example here it is to display the message in a Text View. First, a text View is added to the activity so that it can be used to display the received SMS messages. Next, SMS Receiver class is modified so that when it receives an SMS message, it will broadcast another Intent object. Any application listening for this intent can be notified. The SMS received is also sent out via intent. A broadcast Receiver object, Meier (2007) is used to listen for broadcast intents. When a broadcast intent is received, the SMS message in the TextView is updated. The Intent Filter object is created so that we can listen to the particular intent. In this case, the intent is "SMS_RECEIVED_ACTION". Finally BroadcastReceiver is registered in the activity's onResume () event and unregister it in the onPause () event. This means that the TextView will display the SMS message only when the message is received. If the SMS message is received when the activity is not in the foreground, the TextView will not be updated.

8 CONCLUSION

The application "Dynamic Message App" using Android technology which builds a platform for users to share and view the notices, events, schedules etc. As the world today is much closer and narrow by the grace of mobile devices, we are deploying this application in mobile through which the user can use Message board and get to know about various news and events going around. The results of our explored are promising and free of errors. The dynamic Message App system is full-fledged and user-friendly.

9. FUTURE ENHANCEMENT

1. The future directions can include exploring the possibilities for enhancing the dynamic message app system in many ways. In the application, additional facilities like deleting the messages from notice board and filtering the messages sent by users etc. can be done to enhance the application facility in a more secure manner.
2. Multi language Support: First phase of the system rollout will be in English only. The next phase can be rolled out based on the requirement different language.
3. Messages posted by users can be stored in a centralized database system like MySQL and notices can be retrieved and displayed to users using Web services.
4. Messages posted by users can be filtered by admin.

References

[1] R. Agrawal, T. Imelinski and A. Swami, Mining association rules b/w sets of items in large database, In Proc. of the ACM SIGMOD International Conference on Management of Data (ACM SIGMOD 93), Washington, USA, May 1993.
[2] G. Chartrand and L. Lesniak, Graphs and Digraphs, Chapman and Hall, CRC, 4th edition, 2005.
[3] R. Agrawal and R. Srikant, Fast Algorithm formining association rules, In Proc. of the 20th International Conference on Very Large Database (VLDB' 94), Santiago, Chile, June 1994.
[4] J. Han, J. Pei and Y. Yin, Mining Frequent Patterns without Candidate Generation, In Proc. of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, Texas, USA, May 2000.
[5] S. Brin, R. Motwani, J.D. Ullman and S. Tsur, Dynamic itemset counting and implications rules for masket basket data, In Proc. of the ACM SIGMOD International Conference on Management Data, 1997.
[6] M. Hontsma and A. Swami, Set oriented mining for association rules in relatrend database, The technical report RJ9567, IBM Almaden Research Centre, San Jose, California, October 1993.
[7] J. Han and M. Kamber, Data minining, Concepts and Applications, Elsevier Inc., (2006).
[8] Mark L.Murphy, Beginning Android2, Apress Publisher, Newyork, 2010.
[9] Nicolas Gramlich Android Programming with Tutorials from the anddev.org-Community, Harper Collins Publisher, UK, 2009.
[10] Reto Meier, Professional Android application development, Second Edition, GoogleOHA Publisher, US, 2007.
[11] Unlocking Android : A Developer's Guide, First Edition , by Necrommonger, 2009
[12] The Busy Coder's Guide to Android Development, Version 1.0, by Mark Murphy, US, 2008
[13] Teach Yourself Android Application Development in 24 Hours, by Darcy/Conder, Second Edition, 2012.
[14] David D. Lewis, 1992. "Feature Selection and Feature Extraction for Text Categorization, appeared in Speech and Natural Language", Proceedings of a workshop held at Harriman, New 0York, February 23-26, 1992. Morgan Kaufmann, San Mateo, CA.
[15] Eibe Frank and Ian H. Witten, 2000. "Data Mining: Practical Machine Learning Larning Tools and Techniques with Java Implementation", Morgan Kaufmann Publisher: CA, 2000.
[16] Jiawei Han and Micheline Kamber, 2001. "Data Mining: Concepts and Techniques", Morgan Kaufmann Publisher: CA, 2001.
[17] Arumugam, S., Sabeen, S. "Association Rule Mining using Path Systems in Directed Graphs", INT J COMPUT COMMUN, ISSN 1841-9836, 8(6):791-799, December, 2013.