ANDROID BASED RSS READER WITH AUTO TEXT SUMMARIZATION

Submitted in partial fulfillment of the requirements
of the degree of

BACHELOR OF ENGINEERING
in
COMPUTER ENGINEERING

by

NITHIN JOSE	08
FREDERICK FERNANDO	72
PAUL KOKKAT	77

Supervisor:
Ms. KIRAN ISRANI
(Prof., Computer Engineering Department, XIE)

Xavier Institute Of Engineering, Mahim
Computer Engineering Department
University of Mumbai
2016-2017

CERTIFICATE

This is to certify that the project entitled “**ANDROID BASED RSS READER WITH AUTO TEXT SUMMARIZATION”** is a bonafide work of

NITHIN JOSE	08
FREDERICK FERNANDO	72
PAUL KOKKAT	77

submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **“Bachelor of Engineering”** in **“Computer Engineering”**.

Supervisor/Guide
(Prof. Kiran Israni)

Head of Department
(Prof. Sushama Khanvilkar)

Principal
(Dr. Y.D. Venkatesh)

Project Report Approval for B.E.

This project report entitled **"ANDROID BASED RSS READER WITH AUTO TEXT SUMMARIZATION"** by

NITHIN JOSE	08
FREDERICK FERNANDO	72
PAUL KOKKAT	77

is approved for the degree of **Bachelor of Engineering** in **Computer Engineering** by the **University of Mumbai** during the academic year 2016-2017**.**

Examiner

1._______________________________________.

2._______________________________________.

Date:

Place:

Declaration

We declare that this written submission represents our ideas in our own words and where other’s ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the institute and can also evoke penalty action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Nithin Jose	(08)
Frederick Fernando	(72)
Paul Kokkat	(77)

Date:

Abstract

Today’s world is all about information, most of it available online. The World Wide Web contains billions of documents and is growing at an exponential pace. Tools that provide timely access to, and digest of, various sources are necessary in order to alleviate the information overload people are facing. These concerns have sparked interest in the development of automatic summarization systems. Such systems are designed to take a single article, a cluster of news articles, a broadcast news show, or an email thread as input, and produce a concise and fluent summary of the most important information. Recent years have seen the development of numerous summarization applications for news, email threads, lay and professional medical information, scientific articles, spontaneous dialogues, voicemail, broadcast news and videos, and meeting recordings. These systems, imperfect as they are, have already been shown to help users and to enhance other automatic applications and interfaces. So it is very essential that we create a system that could give the user more ease to navigate and read through all the important contents of the articles or news that they are interested in. Although we have many similar systems out there working on the same premise, there is still a lot of room for improvisation in this field.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1Problem Definition

In computing, a news aggregator, also termed a feed aggregator, feed reader, news reader, RSS reader or simply aggregator, is client software or a web application which aggregates syndicated web content such as online newspapers, blogs, podcasts, and video blogs(vlogs) in one location for easy viewing.

RSS feeds might find himself/herself spending too much of their time for catching up with their favorite news. And sometimes it may so happen that the articles themselves are too long. This may consume a lot of time for the end user.

1.2Aims and Objectives

To build an Android based RSS reader application that automatically summarizes the article and eliminates redundant and less-informative RSS news to save the time for the end user.

1.3Scope of the project

The project could be proven useful and very efficient to many people who would like to save time and be up to date with all their fields of interest, hence increase the productivity of our users. We wish to reach many people with our application and intend to make a change in the already existing crowded application options for RSS readers. We aim to make this application available in the Android Playstore.

1.4 Existing System

RSS was launched in 1999, in the beginning RSS was not a user-friendly gadget and it took some years to convert the technology of news feed commonly spread. Today RSS readers are available on the desktop and mobile front; they take the articles from a particular site using RSS feed. The RSS reader then aggregates these articles and presents them in a chronological order, the user can read a snippet of the article and then thereafter open the link provided in the RSS file to view the whole article.

Some RSS feed readers give a site-by-site categorization of the feed only, while some give a cumulative feed of all of the subscriptions a user might have. Various applications are available on the desktop and mobile front; namely Thunderbird, QuiteRSS, Bittorent clients have integrated RSS feed readers. Web services like feedly, feedreader, TheOldReader are most commonly used. The most popular one of them was Greader which was discontinued by Google in 2013.

Chapter 2

Review of Literature

2.1Study of RSS Technology

A tool used in industry and users for gathering worthy information from the Internet is RSS. RSS is being ferociously used in online news channels, wiki and blog. Using the RSS export of the website user can subscribe to the news. RSS is helpful for saving or retaining updated data on websites that user commonly visit. XML code is used by RSS that detect new information and update itself by feeding the information to the subscribers. RSS is also used to circulate the updated information. RSS feeds means user always gets updated and meaningful substance plethora of useless information. It avoids user to select/reject the substance according to their need. User can select/reject the updates before it is being sent to them. This helps to fill the unwanted updates at the initial level itself. RSS gives advantages to both users and web publishers:

• RSS keep posted the latest information. Latest information which is related to the new music, weather, local news, software upgrade.
• RSS provide subscription to individual user.
• RSS save the surfing time.
• RSS summaries the related article.
• RSS does not use email address to send updates. It makes it spam free and maintain user privacy.

• Easy subscription and unsubscription. To unsubscribe the RSS feed, simply fill up the reason and click on unsubscribe.
• RSS can be used as good marketing /advertising tool.

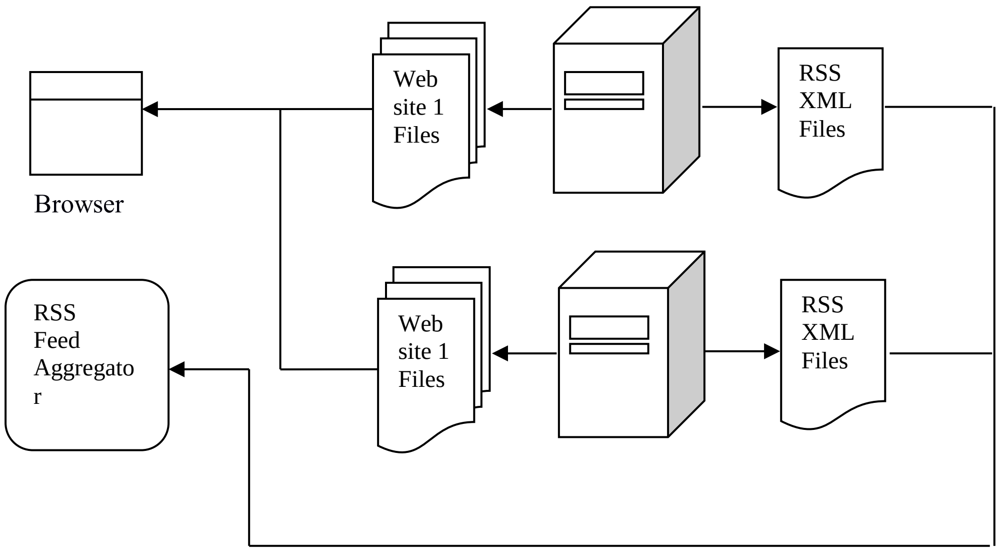

Fig 2.1 Working of an RSS

In the fore mentioned picture the working of an RSS reader is given; nstead of the user to navigate through multiple websites to absorb the data, the RSS reader retrieves this data from multiple sites and serves it to the user in a timely manner. The user has an option to receive timely notifications for each new feed. The feed data is either updated manually by the user or set to auto-retrieve data depending upon the user requirements.

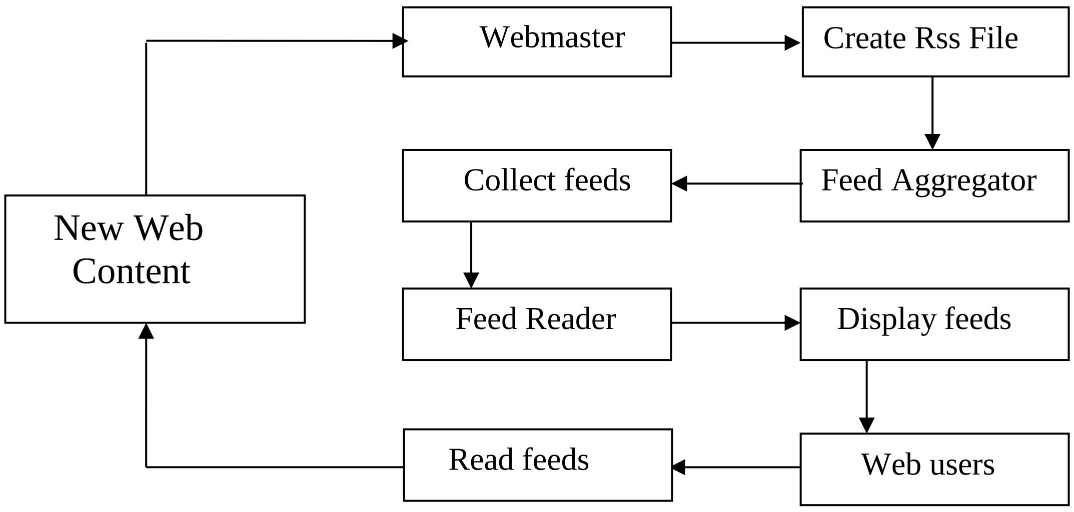

Fig 2.2 Overview of RSS

Web syndication is a form of syndication in which content is made available from one website to other sites. Most commonly, web feeds are made available to provide either summaries or full renditions of a website's recently added content. Web syndication involves a website providing content to an arbitrary number of subscribing websites that redistribute it. For the subscribing sites, syndication is an effective way of adding greater depth and immediacy of information to their pages, making them more attractive to users. For the providing site, syndication increases exposure. This generates new traffic for the transmitting site-making syndication an easy and relatively cheap, or even free, form of advertisement. The prevalence of web syndication is also of note to online marketers, since web surfers are becoming increasingly wary of providing personal information for marketing materials (such as signing up for a newsletter) and expect the ability to subscribe to a feed instead. Although the format could be anything transported over HTTP, such as HTML or JavaScript, it is more commonly XML. Web syndication formats include RSS and Atom.

This screen-grab of one such RSS application for the Windows Desktop Environment is given above. Current RSS application cannot display the whole article within the application. The RSS file could only display a part of the article, hence the user needs to click on the link provided and redirect to the source-site to read the rest of the news. This happens in app or else it fires up the browser to successfully redirect the user to the website. This process is cumbersome and time consuming. This project aims to remove this hurdle faced by the user by extracting the data from the articles by using web-scraping techniques. Thus the data which is needed is readily available at the user's disposal.

2.2 Summarization

Today the internet contains vast amount of electronic collections that often contain high quality information. To summarize the article, the article must be extracted from the website. This can be done using a technique called Web-Scraping.

Web scraping (web harvesting or web data extraction) is a computer software technique of extracting information from websites. This is accomplished by either directly implementing the Hypertext Transfer Protocol (on which the Web is based), or embedding a web browser.

Web scraping is closely related to web indexing, which indexes information on the web using a bot or web crawler and is a universal technique adopted by most search engines. In contrast, web scraping focuses more on the transformation of unstructured data on the web, typically in HTML format, into structured data that can be stored and analyzed in a central local database or spreadsheet. Web scraping is also related to web automation, which simulates human browsing using computer software. Uses of web scraping include online price comparison, contact scraping, weather data monitoring, website change detection, research, web mashup and web data integration.

After the text is extracted from the website, we have to summarize the text using text summarization algorithms. Text summarization is one of the applications of information retrieval, which is the method of condensing the input text into a shorter version, preserving its information content and overall meaning. Most of the previous methods on the sentence extraction-based text summarization task use the graph based algorithm to calculate importance of each sentence in document and most important sentences are extracted to generate document summary. These extraction based text summarization methods give an indexing weight to the document terms to compute the similarity values between sentences. Document features like term frequency, text length are used to assign indexing weight to terms. Therefore document indexing weight remains independent on context in which document term appears. We are considering the problem of context independent document indexing using Lexical association (semantic association). Main motivation behind using Lexical association is the central assumption that the context in which word appears gives important information about its meaning. NLP library is a machine learning based toolkit for the processing of natural language text. It supports the most common Text preprocessing tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and co reference resolution. Context sensitive document indexing is implemented using Text-Rank algorithm to compute how informative each of the document term is.

We get the meaningful tokenized words as output of text preprocessing. Once grammatical aspect of word is clear we use Lexical association (semantic association) to find out association between document words from our offline wordnet dictionary. It is a dictionary in which we have statically stored semantic relationship between all the words in online dictionary and more that are not in it. Next step document graph build using document words and lexical association. In graph word appear as node or vertex and lexical association between two words appear as an edge between those two words. TextRank algorithms applied on this graph to calculate indexing weights of each document word. From these indexing weights we get the similarity between two nodes or words but not in sentences. Our next task is to find out sentence similarity using these indexing weights. Cosine similarity measure is used to construct similarity matrix. It is one of the similarity measure in which similarity between sentences is found using vectors of that sentences. Our method takes advantage of this similarity measure.

In vector representation of sentence words are simply replaced by its calculated indexing weight. Then this computed sentence similarity measure has been used with the graph-based ranking method to build document graph and scored sentences. Then topmost scored sentences are included in summary of document.

The TextRank Algorithm:

$$S(V_i) = (1-d) + d * \sum_{j \in In(V_i)} 1/|Out(V_j)|\, S(V_j)$$

First, the words are assigned parts of speech, so that only nouns and adjectives (or some other combination for different applications) are considered. Then a graph of words is created. The words are the nodes/vertices (denoted V in the paper [1]). Each word is connected to other words that are close to it in the text. In the graph, this is represented by the connections on the graph (denoted E in the paper for edges [2]).

The algorithm is then run on the graph. Each node is given a weight of 1. Then the algorithm goes through the list of nodes and collects the influence of each of its inbound connections. The influence is usually just the value of the connected vertex (initially 1, but it varies) and then summed up to determine the new score for the node. Then these scores are normalized, the highest score becomes 1, and the rest are scaled from 0 to 1 based on that value. Each time through the algorithm gets closer to the actual "value" for each node, and it repeats until the values stop changing.

In post-processing, the algorithm takes the top scored words that have been identified as important and outputs them as key/important words. They can also be combined if they are used together often.

[1] V denotes the collection of all vertices, lowercase v denotes a specific vertex.

[2] E denotes all edges/connections, lowercase e denotes a specific edge. It is often given subscripts to indicate the "number" of the vertex that is being connected from and the vertex that is ring connected to.

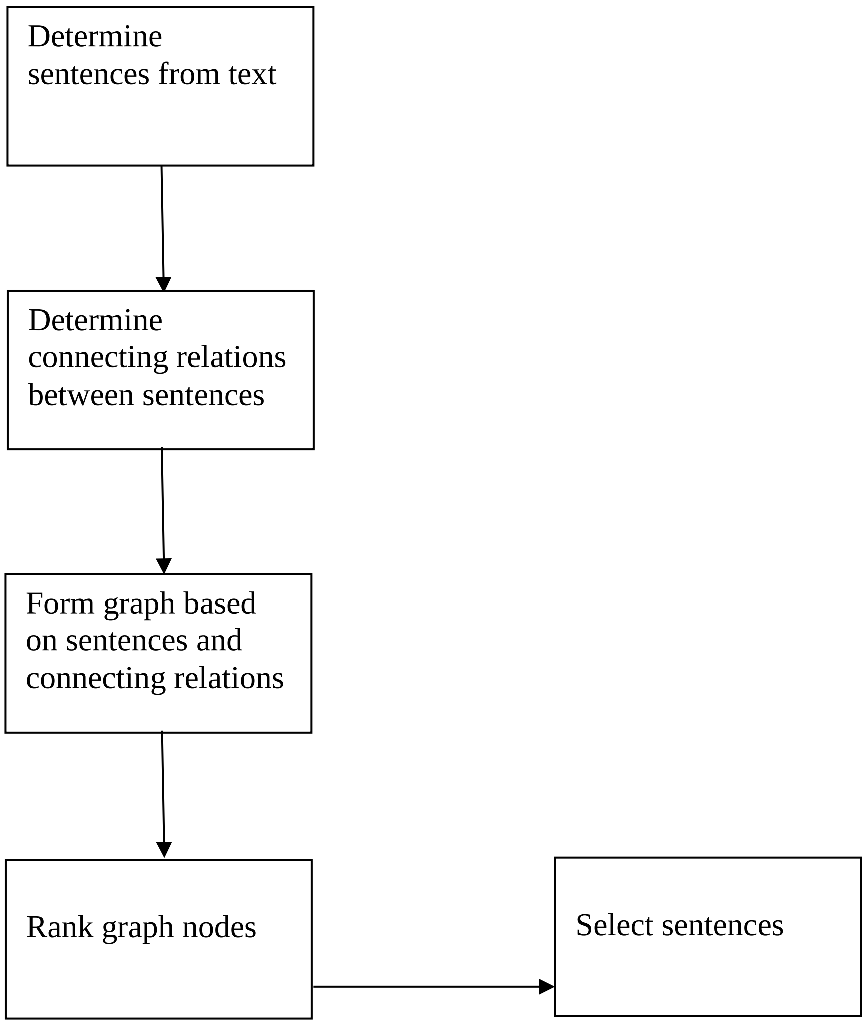
Determine sentences from text
Determine connecting relations between sentences
Form graph based on sentences and connecting relations
Rank graph nodes
Select sentences

Fig 2.3 Text summarization Flowchart

In information retrieval, tf–idf, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus[1]:8 It is often used as a weighting factor in information retrieval and text mining. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

Variations of the tf–idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query. tf–idf can be successfully used for stop-words filtering in various subject fields including text summarization and classification. One of the simplest ranking functions is computed by summing the tf–idf for each query term; many more sophisticated ranking functions are variants of this simple model.

2.3Similarity and a Fuzzy Equivalence Relation

Computing the similarity between documents has been extensively studied as an essential tool for applications such as text document searching, document clustering, copy or plagiarism detection, text document retrieval filtering, and categorization. Automatically determining whether two documents are similar and to what extent they are similar is a non-trivial problem. We first filtered the content in the “title” and “text” tags of each Wikipedia article.

The filtering process requires two steps:

(i) Removing Stopwords

(ii) Stemming remaining words

Stopwords, which are words that are very common, such as prepositions and demonstrative, interrogative, and indefinite pronouns, do not provide useful information to distinguish the contents of different documents Stopword removal is accomplished by verifying if a word is contained in the stopwords hash table, which is constructed by using several widely used stopword lists. Once the stopwords are removed, we proceed to stem all the remaining words using the Porter algorithm.

The correlation factors of different keywords computed by using the keyword-connection method follow the conjuncture that “The more documents in which two keywords occur, the more they relate to each other.” The co-occurrence correlation factor not only considers the number of documents in a collection where both words w1 and w2 appear, but it also considers the frequency of co-occurrence of both w1 and w2 in a document.

Even though the co-occurrence frequency matrix considers the co-occurrence frequency of any two words, it does not consider how close any two words are as they appear in a document. The distance correlation factor between any two words w1 and w2 considers the frequency of occurrence, as well as the “distance,” that is measured by the number of words, between w1 and w2 within a document as an additional factor to calculate the correlation factor of w1 and w2. In a document that discusses computer architecture, the two words “computer” and “architecture” are likely to appear closely together most of the times, and their correlation factor computed by using their distances would be higher if their positions are considered, along with their frequency of co-occurrence.

Once a word-to-word correlation matrix is calculated, we can define a fuzzy set association of each word and a sentence, paragraph, or document itself. Since a news article in RSS news feeds includes only a brief summary, i.e., the 3-4 line summary, in the Title and Description sections, we treat the Title and Description sections as the content of an RSS news article such that the degree of similarity between any two RSS news articles is determined by the correlation factors among the words in the respective Title and Description sections of the two articles. We counted

(i)The number of pairs of RSS news entries that were considered redundant or less-informative when in fact they are not (i.e. false positives)

(ii) The number of pairs that were considered different but are in fact redundant or less informative (i.e., false negatives)

Hereafter, we proceed to eliminate less-informative RSS news articles. This task can be accomplished by first generating clusters of all the RSS news articles that maintain certain degree of similarity.

R is a fuzzy equivalence relation if it is reflexive, symmetric, and max-min transitive,

$R(x, x) = 1, \forall x \in R$ (9)

$R(x, y) = R(y, x), \forall x, y \in R$ (10)

$R(x, z) \geq \max_{y \in Y} \min\{R(x, y), R(y, z)\}$ (11)

Chapter 3

Description

3.1 Analysis

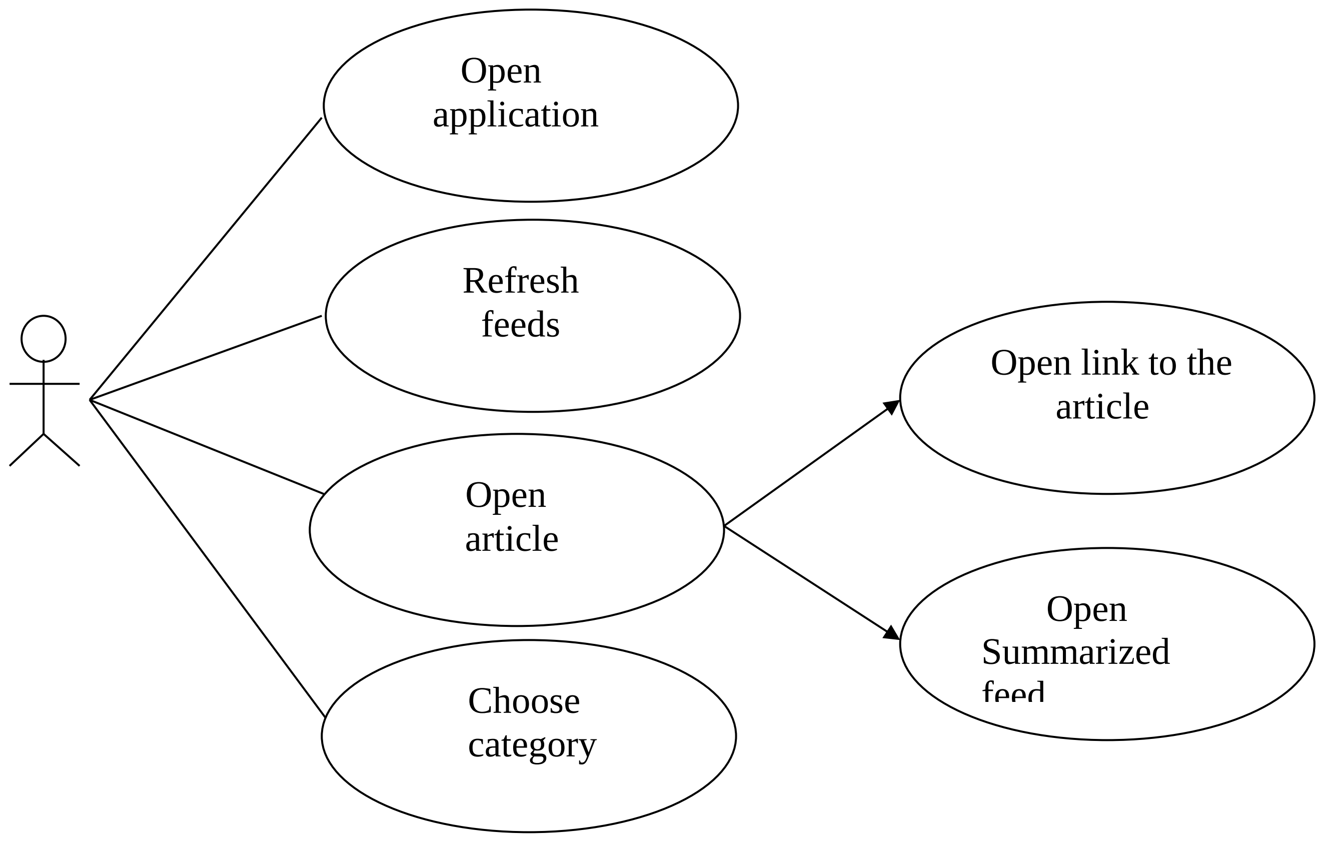

Fig.3.1.1 Use-Case Diagram

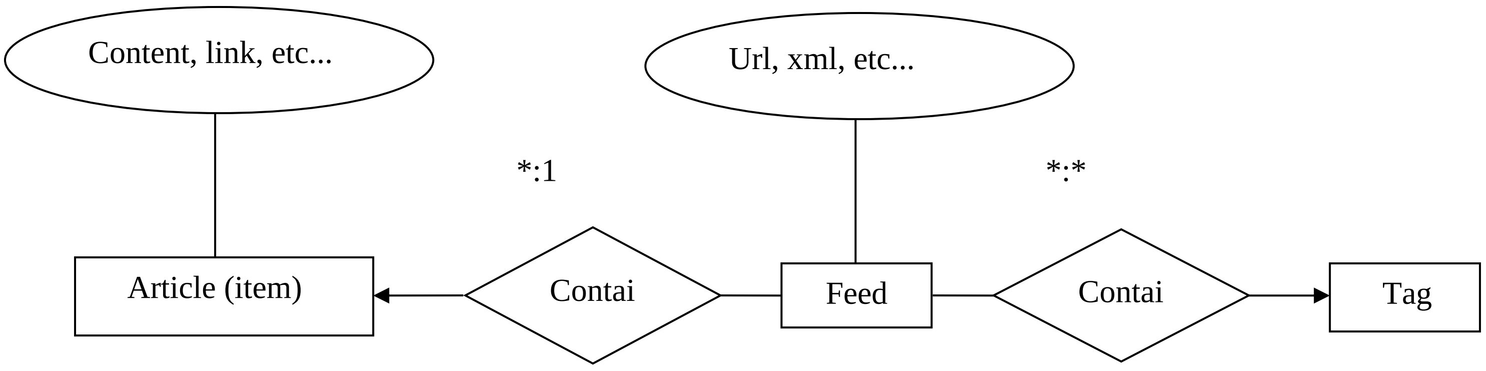

Fig. 3.1.2 E-R Diagram

3.2Design

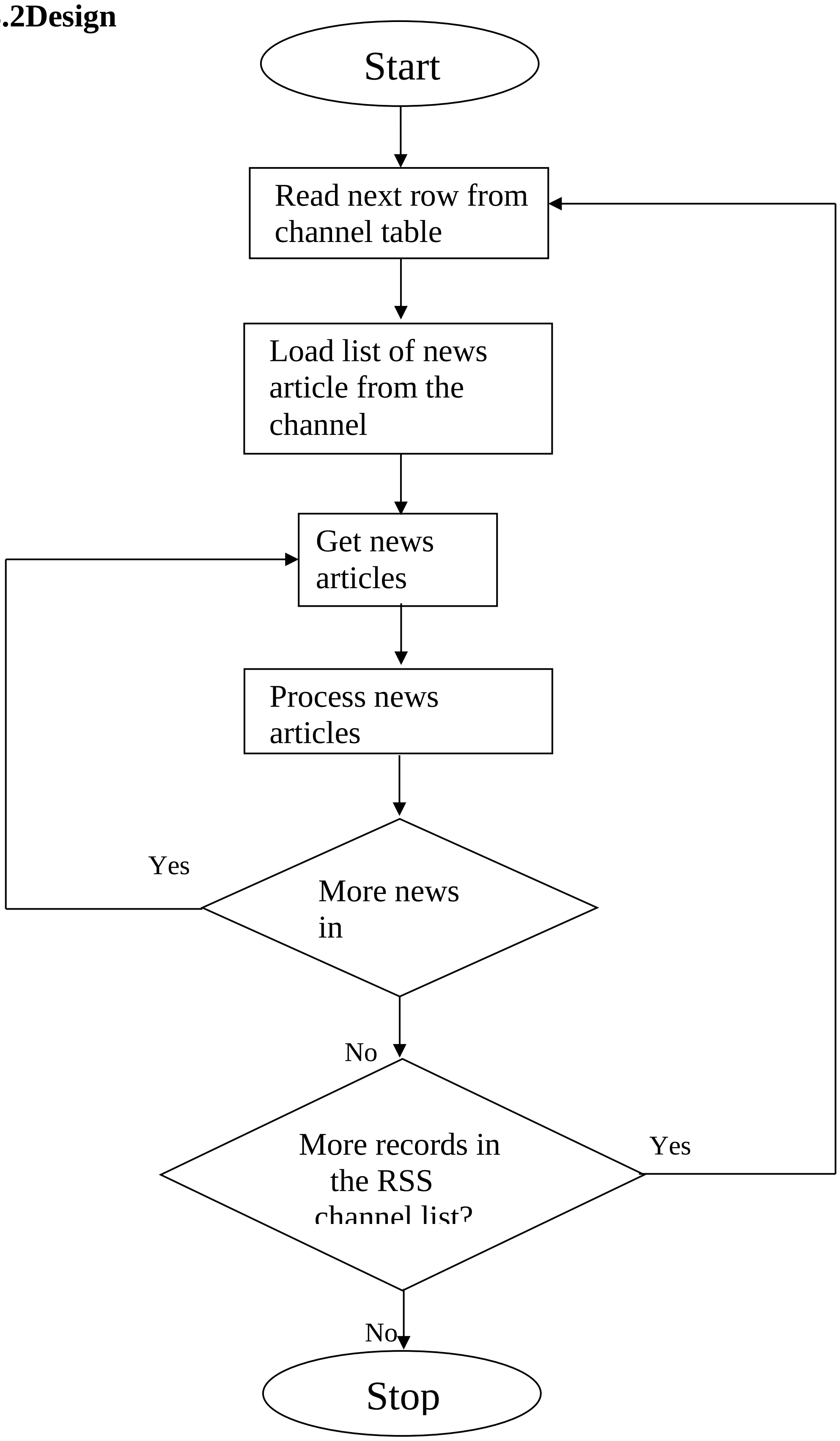
Start
Read next row from channel table
Load list of news article from the channel
Get news articles
Process news articles
Yes
More news in
No
More records in the RSS channel list?
Yes
No
Stop

Fig.3.2.1 RSS Flowchart

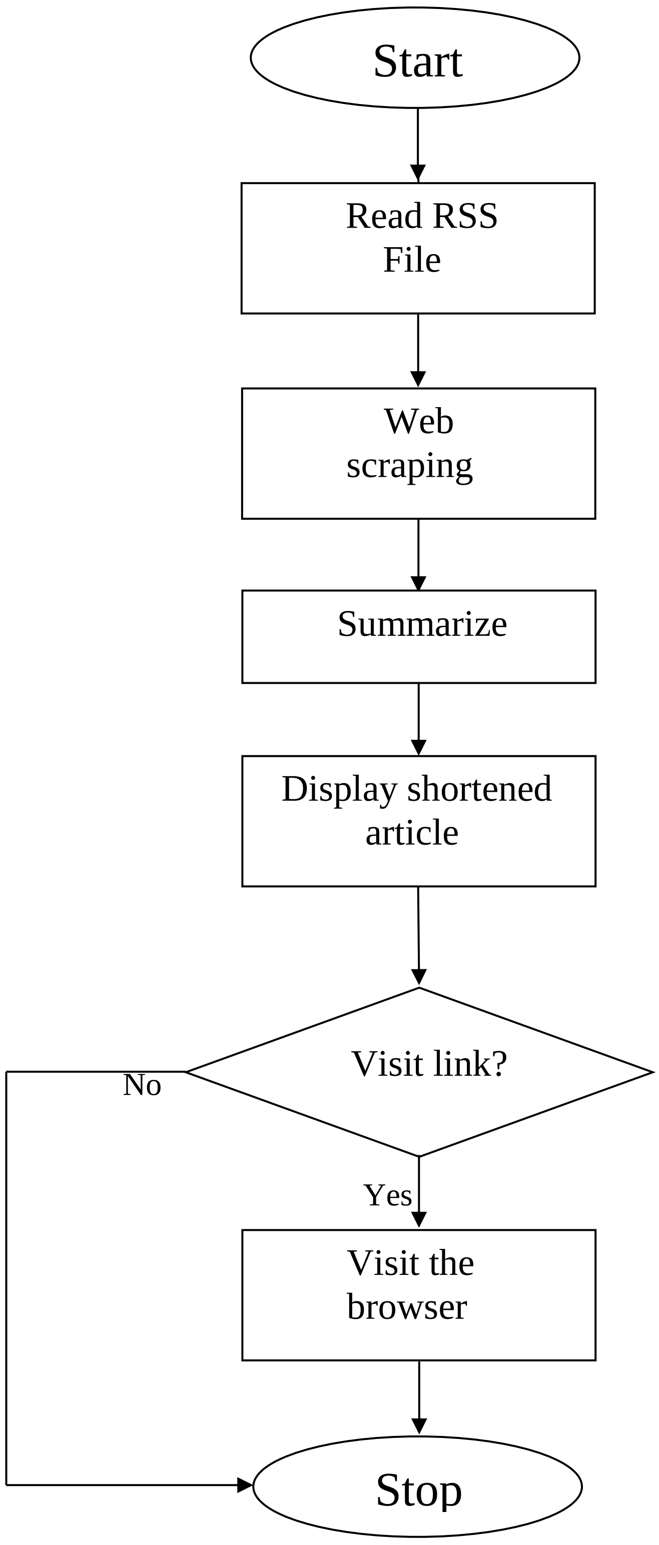

Fig. 3.2.2 RSS Service Flowchart

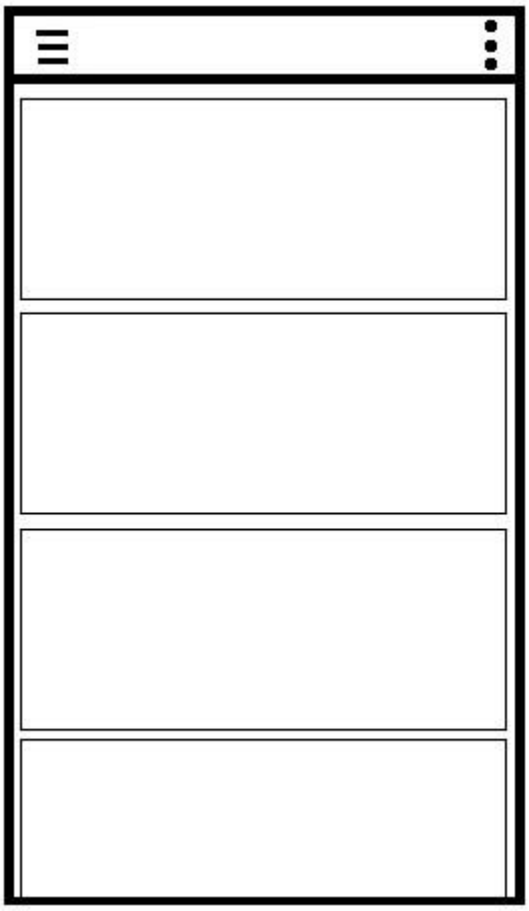
Fig 3.2.3 GUI- Article Feed

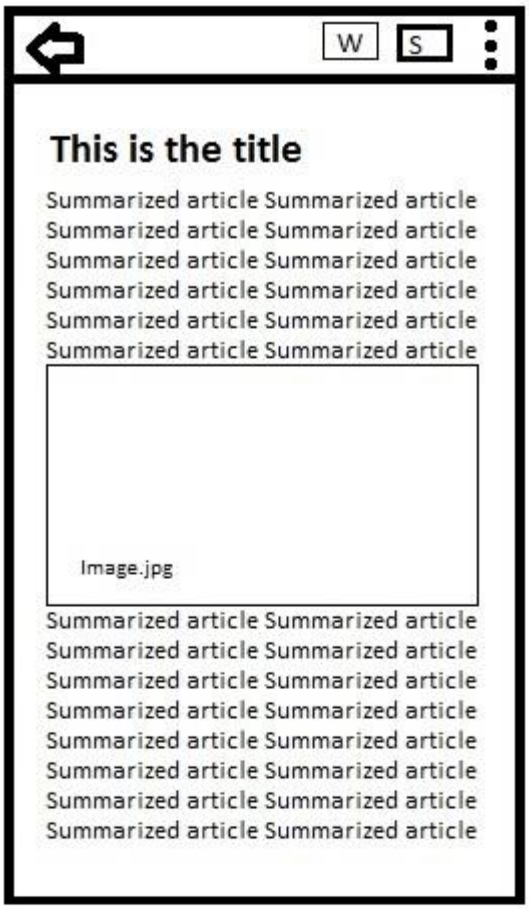
W
S
This is the title
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Image.jpg
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article
Summarized article Summarized article

Fig 3.2.4 GUI Summarized View

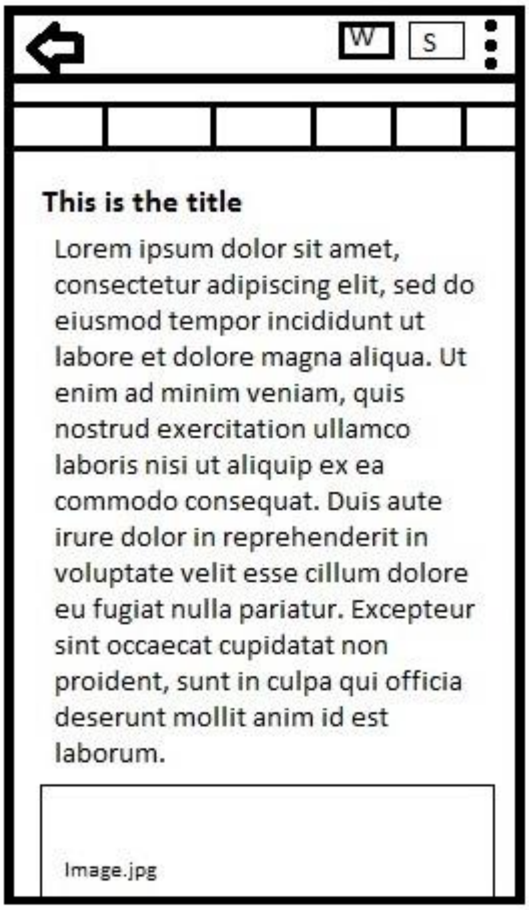
W
S
This is the title
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Image.jpg

Fig 3.2.5 GUI – Web View

3.3 Details of Hardware & Software

Hardware:

- PC/Laptop capable of running Java.
- Mobile running Android OS v4.0 and above

Software:

- NTLK libraries for Natural Language Processing
- Windows/Linux Operating system
- Android Studio
- Git and Bitbucket for version control
- Slack for communication
- Teamviewer

References

[1] Geetanjali Singh *et al*, “Really Simple Syndication (RSS) Technology Tools”, 2015 IEEE International Conference on Computational Intelligence & Communication Technology.

[2] Dipti.D.Pawar *et al, “*Text Rank: A Novel Concept for Extraction Based Text Summarization”, International Journal of Computer Science and Information Technologies, Vol. 5 (3) , 2014, 3301 – 3304.

[3] Ian Garcia *et al, “*Eliminating Redundant and Less-Informative RSS News Articles Based on Word Similarity and a Fuzzy Equivalence Relation”, Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence.

Acknowledgement

We sincerely thank Ms. Kiran Israni our project guide for her continuous support, supervision, motivation and guidance throughout the project in spite of her hectic schedule and her experience gave us the light in handling research project and helped us in clarifying the concepts, handling critical situations and in understanding the objective of our work.

We also wish to express our gratitude to the other staff members of Computer department who rendered their help during the period of project work.