Text Reformulations using Graph Based TextRank Algorithm

Project report submitted in complete fulfilment of the requirement for the degree of Bachelor of Technology

In

Computer Science and Engineering

By

Shikher Aatrey (131243)
Rahul Arora (131231)

Under the supervision of

Dr. Pradeep Kumar Gupta

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this report entitled **"Text Reformulations using Graph Based Text-Rank Algorithm"** in complete fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from July 2016 to Dec 2016 under the supervision of **Dr. Pradeep Kumar Gupta** Assistant Professor, Department of Computer Science And Engineering. The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature)	(Student Signature)
Shikher Aatrey(131243)	Rahul Arora(131231)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)
Dr. Pradeep Kumar Gupta
Assistant Professor
Department of Computer Science And Engineering

ACKNOWLEDGEMENT

We are grateful and indebted to **Dr. Pradeep Kumar Gupta**, Assistant professor, Department of Computer Science And Engineering for his help and advice in completion of this project report. We also express our deep sense of gratitude and appreciation to our guide for his constant supervision, inspiration and encouragement right from the beginning of this project. We also want to thank our parents and friends for their immense support and confidence upon us. We deem it a pleasant duty to place on record our sincere and heartfelt gratitude to our project guide for his long sightedness, wisdom and co-operation which helped us in tackling crucial aspects of the project in a very logical and practical way.

Shikher Aatrey	Rahul Arora
(131243)	(131231)

Table of Contents

List of Abbreviations

1. **ML** - Machine Learning
2. **IEEE** - Institute of Electrical and Electronics Engineers
3. **ACM** - American Computer Machinery
4. **TF-IDF** - Term Frequency - Inverse Document Frequency
5. **NLP** - Natural Language Processing
6. **AWS** - Amazon Web Services

List of Figures

ABSTRACT

We comprehend individuals don't have sufficient energy to experience long news articles regular. So through this project, we examine the way toward diminishing a content archive with a PC program keeping in mind the end goal to make a synopsis that holds the most imperative purposes of the first record. In particular, similar to key-phrase extraction, text summarisation means to distinguish the embodiment of a content. The main genuine contrast is that now we are managing bigger content units—entire sentences rather than words and expressions.

Our techniques on the sentence extraction-based content summarisation errand utilise the chart based TextRank algorithm to figure significance of every sentence in record and most critical sentences are extracted to produce report rundown. These extraction based content summarisation techniques give an ordering weight to the record terms to register the likeness values between sentences.

1. INTRODUCTION

1.1 Text Reformulation

Automatic reformulation is the process of reducing a text document with a computer program in order to create a summary that retains the most important points of the original document. Technologies that can make a coherent summary take into account variables such as length, writing style and syntax.[1] Automatic data summarization is part of machine learning and data mining. The main idea of summarization is to find a representative subset of the data, which contains the *information* of the entire set. Summarization technologies are used in a large number of sectors in industry today. An example of the use of summarization technology is search engines such as Google. Other examples include document summarization, image collection summarization and video summarization. Document summarization, tries to automatically create a *representative summary* or *abstract* of the entire document, by finding the most *informative* sentences. Similarly, in image summarization the system finds the most representative and important (or salient) images. Similarly, in consumer videos one would want to remove the boring or repetitive scenes, and extract out a much shorter and concise version of the video. This is also important, say for surveillance videos, where one might want to extract only important events in the recorded video, since most part of the video may be uninteresting with nothing going on. As the problem of information overload grows, and as the amount of data increases, the interest in automatic summarization is also increasing.

Generally, there are two approaches to automatic summarization: *extraction* and *abstraction*. Extractive methods work by selecting a subset of existing words, phrases, or sentences in the original text to form the summary.[2] In contrast, abstractive methods build an internal semantic representation and then use natural language generation techniques to create a summary that is closer to what a human might generate. Such a summary might contain words not explicitly present in the original. Research into abstractive methods is an increasingly important and active research area, however due to complexity constraints, research to date has focused primarily on extractive methods. In some application domains, extractive summarization makes more sense. Examples of these include image collection summarization and video summarization.

1.2 TextRank Algorithm

TextRank is a broadly useful graph-based positioning calculation for NLP. Basically, it runs PageRank on a diagram uniquely intended for a specific NLP undertaking. For key-express extraction, it assembles a chart utilising some arrangement of content units as vertices. Edges depend on some measure of semantic or lexical comparability between the content unit vertices. Not at all like PageRank, the edges are ordinarily undirected and can be weighted to mirror a level of closeness.

Graph-based ranking algorithms are essentially a way of deciding the importance of a vertex within a graph, based on global information recursively drawn from the entire graph.

The basic idea implemented by a graph-based ranking model is that of “voting” or “recommendation”. When one vertex links to another one, it is basically casting a vote for that other vertex. The higher the number of votes that are cast for a vertex, the higher the importance of the vertex. Moreover, the importance of the vertex casting the vote determines how important the vote itself is, and this information is also taken into account by the ranking model. Hence, the score associated with a vertex is determined based on the votes that are cast for it, and the score of the vertices casting these votes.

1.3 Text as a Graph

To empower the use of graph-based positioning calculations to common dialect writings, we need to assemble a chart that speaks to the content, and interconnects words or other content substances with important relations.[6] Contingent upon the current application, content units of different sizes and attributes can be included as vertices in the chart, e.g. words, collocations, whole sentences, or others. Also, the application manages the sort of relations that are utilised to draw associations between any two such vertices, e.g. lexical or semantic relations, logical cover, and so forth. Despite the sort and attributes of the components added to the diagram, the use of chart based positioning calculations to common dialect writings comprises of the accompanying fundamental strides:

1. Identify text units that best define the task at hand, and add them as vertices in the graph.
2. Identify relations that connect such text units, and use these relations to draw edges between vertices in the graph. Edges can be directed or undirected, weighted or unweighted.
3. Iterate the graph-based ranking algorithm until convergence.
4. Sort vertices based on their final score. Use the values attached to each vertex for ranking/selection decisions.

The text when converted to a fully-connected graph looks similar to Figure - 1.1 below.

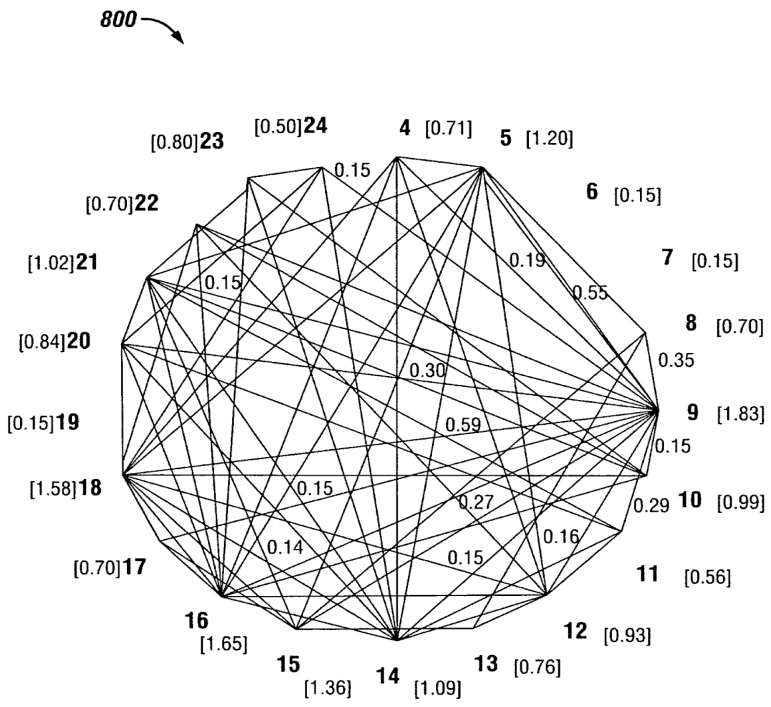


Figure - 1.1
Representing text as a fully-connected graph

1.4 Recommendation Engine

A recommendation engine (sometimes referred to as a recommender system) is a tool that lets algorithm developers predict what a user may or may not like among a list of given items. Recommendation engines are a pretty interesting alternative to search fields, as recommendation engines help users discover products or content that they may not come across otherwise. This makes recommendation engines a great part of web sites and services such as Facebook, YouTube, Amazon, and more. Figure -1.2 gives us an idea about how Recommendations actually work.

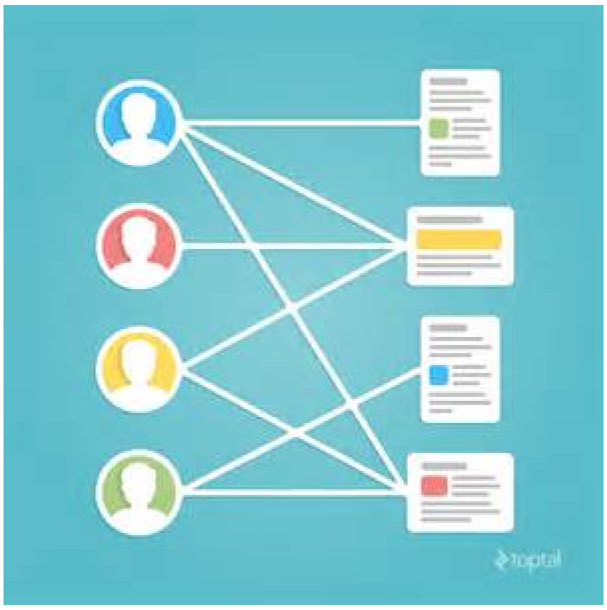


Figure - 1.2

Working of Recommendation systems

Recommendation engines work ideally in one of two ways. It can rely on the properties of the items that a user likes, which are analyzed to determine what else the user may like; or, it can rely on the likes and dislikes of other users, which the recommendation engine then uses to compute a similarity index between users and recommend items to them accordingly. It is also possible to combine both these methods to build a much more robust recommendation engine. However, like all other information related problems, it is essential to pick an algorithm that is suitable for the problem being addressed.

1.5 Problem Statement

Today web contains incomprehensible measure of electronic accumulations that regularly contain excellent data. Nonetheless, more often than not the Internet gives more data than is required. Client needs to choose best gathering of information for specific data require in least conceivable time Text summarisation is one of the utilisations of data recovery, which is the strategy for consolidating the info content into a shorter variant, protecting its data substance and general significance. There has been an enormous measure of work on inquiry particular summarisation of records utilising closeness measure. This venture concentrates on sentence extraction based single report summarisation. The vast majority of the past techniques give an ordering weight to the archive terms to process the likeness values between sentences Document highlights like term recurrence, content length are utilised to dole out ordering weight to terms. Accordingly archive ordering weight stays free on setting in which report term shows up. The ordering strategies utilised as a part of existing models can't recognise terms reflected in sentence closeness values. Next to no work has been accomplished for the issue of setting free archive ordering for the content summarisation undertaking. This venture utilises the calculation which utilises an ordering structure as a part of which list is based on the premise of setting of the record as opposed to on the terms premise.

1.6 Aims And Objectives

The overall aim is to analyse and introduce the TextRank graph-based ranking model for graphs extracted from natural language texts.

1.7 Methodology

In this project, summarisation is finished by applying content positioning over the worldwide reasonable lumps in the report. The sentences whose weight is over the limit is picked and revamped in the request in which the sentences showed up in the first record.

2. LITERATURE SURVEY

In this day and age time is restricted yet the data accessible online is in overabundance. With the approach of individual portable registering gadgets, we are being given a blast of data consistently. This makes it progressively vital to expend however much data as could reasonably be expected at all measure of time, while killing immaterial and repetitive information. News is another space which falls prey to this data over burden. With the rise of exceptionally quick news conveyance through web-based social networking administrations, for example, Twitter, Facebook well as different News Sources, It is a desperate need of the day to spare time and handle simply enough data that is required about current occasions. This issue can be tackled utilising Automatic Summarisation of News Articles, making it conceivable to expend News rapidly and in nibble measured scraps.

Text Summarisation is an approach that can be utilised to pack substantial content articles into littler and more concise rundowns. This fundamentally utilises Natural Language Processing standards and calculations to comprehend articles and produce littler and effective Summaries. Innovations that can make an intelligent synopsis consider factors, for example, length, composing style and linguistic structure. It gives a truly necessary arrangement during a time of progressively limited capacity to focus and the TL;DR (Too Long; Didn't Read) era.

Programmed summarisation is the way toward lessening a content report with a PC program keeping in mind the end goal to make synopsis that holds the most critical purposes of the first record. Innovations that can make a sound rundown consider factors, for example, length, composing style and language structure. Programmed information summarisation is an imperative range inside machine learning and information mining. Summarisation advancements are utilised today, in countless in industry today. A case of the utilisation of summarisation innovation is web crawlers, for example, Google. Different illustrations incorporate report summarisation, picture accumulation summarisation and video summarisation.

The fundamental thought of summarisation is to locate a delegate subset of the information, which contains the data of the whole set. Document summarisation, tries to automatically create a representative summary or abstract of the entire document, by finding the most informative sentences.

Characteristic dialect handling (NLP) is a field of software engineering, computerised reasoning, and computational etymology worried with the connections amongst PCs and human (regular) dialects. All things considered, NLP is identified with the region of human–computer communication. Many difficulties in NLP include common dialect understanding, that is, empowering PCs to get importance from human or normal dialect information, and others include characteristic dialect era. The significant assignments in NLP which are identified with Text Summarisation incorporate Parsing, Part-of-Speech Tagging, Tokenisation, Stemming, Coreference determination and so on.

Chart based positioning calculations, for example, Kleinberg's HITS calculation (Kleinberg, 1999) or Google's PageRank (Brin and Page, 1998), have been generally and effectively utilized as a part of reference investigation, informal organizations, and the examination of the connection structure of the World Wide Web. To put it plainly, a diagram based positioning calculation is a method for settling on the significance of a vertex inside a chart, by considering worldwide data recursively registered from the whole diagram, as opposed to depending just on nearby vertex-particular data. A comparative line of intuition can be connected to lexical or semantic charts separated from normal dialect records, bringing about a diagram based positioning model called TextRank (Mihalcea and Tarau, 2004), which can be utilized for an assortment of common dialect handling applications where learning drawn from a whole content is utilized as a part of making neighborhood positioning/choice choices. Such content situated positioning strategies can be connected to assignments running from robotized extraction of key expressions, to extractive summarisation and word sense disambiguation (Mihalcea et al., 2004). In this venture, we examine a scope of chart based positioning calculations, and assess their application to programmed unsupervised sentence extraction in the setting of a content summarisation errand. We demonstrate that the outcomes got with this new unsupervised strategy are focused with beforehand created best in class frameworks.

Recommender systems are an important part of the information and e-commerce ecosystem. They represent a powerful method for enabling users to filter through large information and product spaces. Nearly two decades of research on collaborative filtering have led to a varied set of algorithms and a rich collection of tools for evaluating their performance. Research in the field is moving in the direction of a richer understanding of how recommender technology may be embedded in specific domains. The differing personalities exhibited by different recommender algorithms show that recommendation is not a one-size- fits-all problem. Specific tasks, information needs, and item domains represent unique problems for recommenders, and design and evaluation of recommenders needs to be done based on the user tasks to be supported. Effective deployments must begin with careful analysis of prospective users and their goals. Based on this analysis, system designers have a host of options for the choice of algorithm and for its embedding in the surrounding user experience. This paper discusses a wide variety of the choices available and their implications, aiming to provide both practicioners and researchers with an introduction to the important issues underlying recommenders and current best practices for addressing these issues.

Collaborative filtering (CF) is a popular recommendation algorithm that bases its predictions and recommendations on the ratings or behavior of other users in the system. The fundamental assumption behind this method is that other users' opinions can be selected and aggregated in such a way as to provide a reasonable prediction of the active user's preference.[3] Intuitively, they assume that, if users agree about the quality or relevance of some items, then they will likely agree about other items — if a group of users likes the same things as Mary, then Mary is likely to like the things they like which she hasn't yet seen. There are other methods for performing recommendation, such as finding items similar to the items liked by a user using textual similarity in metadata (content-based filtering or CBF). The focus of this survey is on collaborative filtering methods, although content-based filtering will enter our discussion at times when it is relevant to overcoming a particular recommender system difficulty. The majority of collaborative filtering algorithms in service today, including all algorithms detailed in this section, operate by first generating predictions of the user's preference and then produce their recommendations by ranking candidate items by predicted preferences. Often this prediction is in the same scale as the ratings provided by

88 2.1 Baseline Predictors 89 users, but occasionally the prediction is on a different scale and is meaningful only for candidate ranking. This strategy is analagous to the common information retrieval method of producing relevance scores for each document in a corpus with respect to a particular query and presenting the top-scored items. Indeed, the recommend task can be viewed as an information retrieval problem in which the domain of items (the corpus) is queried with the user's preference profile. Therefore, this section is primarily concerned with how various algorithms predict user preference. In later sections we will discuss recommendation strategies that diverge from this structure, but in actual implementation they frequently start with a preference-ranked list of items and adjust the final recommendation list based on additional criteria.

2.1 Types of Summarisation

The Summary of a document is a reduced, though precise, representation of the text which seeks to render the exact idea of its contents. Its principal objective is to give information about and provide privileged access to the source documents. Summarisation is automatic when it is generated by software or an algorithm. The main types of of automatic summarisation include extraction-based and abstraction-based, maximum entropy-based.[7]

2.1.1 Extraction-Based Summarisation

Extraction consists of selecting units of text (sentences, segments of sentences, paragraphs or passages), deemed to contain a document's essential information, and of assembling these units in an adequate way.

2.1.2 Abstraction-Based Summarisation

Extraction strategies simply duplicate the data regarded most vital by the framework to the outline (for instance, key provisions, sentences or passages), while reflection includes rewording areas of the source archive. When all is said in done, reflection can gather a content more firmly than extraction, however the projects that can do this are harder to create as they require the utilisation of regular dialect era innovation, which itself is a developing field. Frameworks that create synopses by reflection depend on content comprehension and try to produce a syntactically right, brief and reasonable content. Not very many conceptual summarisation frameworks have been made. Slob is one such framework and was one of the first to utilise semantic understanding for English writings to create their outlines.

2.2 Graph-Based Ranking Algorithms

Graph-based ranking algorithms are essentially a way of deciding the importance of a vertex within a graph, based on information drawn from the graph structure. In this section, we present three graph-based ranking algorithms – previously found to be successful on a range of ranking problems. We also show how these algorithms can be adapted to undirected or weighted graphs, which are particularly useful in the context of text-based ranking applications. Let G = (V, E) be a directed graph with the set of vertices V and set of edges E, where E is a subset of V × V . For a given vertex Vi, let In(Vi) be the set of vertices that point to it (predecessors), and let Out(Vi) be the set of vertices that vertex Vi points to (successors).

2.2.1 HITS

HITS (Hyperlinked Induced Topic Search) (Kleinberg, 1999) is an iterative calculation that was intended for positioning Web pages as per their level of "power". The HITS calculation makes a refinement between "powers" (pages with an expansive number of approaching connections) and "centre points" (pages with an extensive number of active connections). For every vertex, HITS produces two arrangements of scores – a "power" score, and a "centre point" score:[9] Look at Figure - 2.1 for the formula.

$$HITS_A(V_i) = \sum_{V_j \in In(V_i)} HITS_H(V_j)$$

$$HITS_H(V_i) = \sum_{V_j \in Out(V_i)} HITS_A(V_j)$$

Figure - 2.1

Formula for calculating HITS

2.2.2 Positional Power Function

Introduced by (Herings et al., 2001), the positional power function is a ranking algorithm that determines the score of a vertex as a function that combines both the number of its successors, and the score of its successors.[9] The formula for the same is given in Figure - 2.2

$$POS_P(V_i) = \frac{1}{|V|} \sum_{V_j \in Out(V_i)} (1 + POS_P(V_j))$$

Figure - 2.2

Formula for calculating POS_p

The counterpart of the positional power function is the positional weakness function, defined as Figure 2.3:

$$POS_W(V_i) = \frac{1}{|V|} \sum_{V_j \in In(V_i)} (1 + POS_W(V_j))$$

Figure - 2.3

Formula for calculating POS_W

2.2.3 PageRank

PageRank (Brin and Page, 1998) is maybe a standout amongst the most prominent positioning calculations, and was planned as a strategy for Web connect examination. Dissimilar to other positioning calculations, PageRank coordinates the effect of both approaching and active connections into one single model, and in this manner it delivers just a single arrangement of scores:

$$PR(V_i) = (1 - d) + d * \sum_{V_j \in In(V_i)} \frac{PR(V_j)}{|Out(V_j)|}$$

Figure - 2.4

Formula for calculating PageRank

Have a look at Figure - 2.4 for the detailed formula. Here d is a parameter that is set between 0 and 1. For each of these algorithms, starting from arbitrary values assigned to each node in the graph, the computation iterates until convergence below a given threshold is achieved. After running the algorithm, a score is associated with each vertex, which represents the "importance" or "power" of that vertex within the graph. Notice that the final values are not affected by the choice of the initial value, only the number of iterations to convergence may be different.

2.2.4 Undirected Graphs

Albeit generally connected on coordinated diagrams, re-cursive chart based positioning calculations can be additionally connected to undirected diagrams, in which case the out-level of a vertex is equivalent to the in-level of the vertex. For approximately associated charts, with the quantity of edges relative with the quantity of vertices, undirected diagrams have a tendency to have more continuous meeting bends. As the availability of the chart increments (i.e. bigger number of edges), union is normally accomplished after less cycles, and the union bends for coordinated and undirected charts basically cover.

2.2.5 Weighted Graphs

With regards to Web surfing or reference examination, it is abnormal for a vertex to incorporate numerous or incomplete connections to another vertex, and consequently the first definition for chart based positioning calculations is accepting unweighted diagrams.

Notwithstanding, in our TextRank demonstrate the charts are work from normal dialect messages, and may incorporate various or fractional connections between the units (vertices) that are separated from content. It might be subsequently helpful to show and consolidate into the model the "quality" of the association between two vertices Vi and Vj as a weight wij added to the relating edge that interfaces the two vertices.

Therefore, we present new formulae for diagram based positioning that consider edge weights when figuring the score connected with a vertex in the chart.[7]

$$HITS_A^W(V_i) = \sum_{V_j \in In(V_i)} w_{ji} HITS_H^W(V_j)$$

$$HITS_H^W(V_i) = \sum_{V_j \in Out(V_i)} w_{ij} HITS_A^W(V_j)$$

$$POS_P^W(V_i) = \frac{1}{|V|} \sum_{V_j \in Out(V_i)} (1 + w_{ij} POS_P^W(V_j))$$

$$POS_W^W(V_i) = \frac{1}{|V|} \sum_{V_j \in In(V_i)} (1 + w_{ji} POS_W^W(V_j))$$

$$PR^W(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} w_{ji} \frac{PR^W(V_j)}{\sum_{V_k \in Out(V_j)} w_{kj}}$$

Figure - 2.5

Formulae for calculating TextRank score

While the final vertex scores (and therefore rankings) for weighted graphs differ significantly as compared to their unweighted alternatives, the number of iterations to convergence and the shape of the convergence curves is almost identical for weighted and unweighted graphs. All the formulae for calculating TextRank are represented in Figure - 2.5

2.3 Sentence Extraction

To empower the use of graph based ranking algorithm to normal dialect writings, TextRank begins by building a chart that speaks to the content, and interconnects words or other content substances with significant relations. For the undertaking of sentence extraction, the objective is to rank whole sentences, and in this way a vertex is added to the diagram for every sentence in the content. To set up associations (edges) between sentences, we are characterising a "similitude" connection, where "likeness" is measured as a component of substance cover. Such a connection between two sentences can be viewed as a procedure of "suggestion": a sentence that addresses certain ideas in a content, gives the peruser a "proposal" to allude to different sentences in the content that address similar ideas, and in this way a connection can be drawn between any two such sentences that share normal substance.

The overlap of two sentences can be determined simply as the number of common tokens between the lexical representations of the two sentences, or it can be run through syntactic filters, which only count words of a certain syntactic category. Moreover, to avoid promoting long sentences, we are using a normalisation factor, and divide the content overlap of two sentences with the length of each sentence. Formally, given two sentences Si and Sj, with a sentence being represented by the set of Ni words that appear in the sentence:
Si = Wi1, Wi2 , ..., WiNi, the similarity of Si and Sj is given in Figure - 2.6:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Figure - 2.6

Formula for Jaccard's Similarity

The subsequent diagram is exceedingly associated, with a weight connected with every edge, demonstrating the quality of the associations between different sentence combines in the content. The content is along these lines spoke to as a weighted diagram, and therefore we are utilising the weighted chart based positioning formulae. The chart can be spoken to as: (a) basic undirected diagram; (b) coordinated weighted diagram with the introduction of edges set from a sentence to sentences that follow in the content (coordinated forward); or (c) coordinated weighted chart with the introduction of edges set from a sentence to past sentences in the content (coordinated in reverse). After the positioning calculation is keep running on the diagram, sentences are sorted in turned around request of their score, and the top positioned sentences are chosen for incorporation in the synopsis. The sentences with the most astounding rank are chosen for incorporation in theory.

2.4 Why TextRank Works

Why TextRank Works Intuitively, TextRank functions admirably in light of the fact that it doesn't just depend on the neighbourhood setting of a content unit (vertex), but instead it considers data recursively drawn from the whole content (diagram). Through the diagrams it expands on writings, TextRank recognises associations between different elements in a content, and actualises the idea of proposal.

A content unit prescribes other related content units, and the quality of the suggestion is recursively processed in light of the significance of the units making the proposal. For example, in the key-expression extraction application, co-happening words suggest each different as critical, and it is the regular setting that empowers the ID of associations between words in content. During the time spent distinguishing imperative sentences in a content, a sentence suggests another sentence that locations comparable ideas as being helpful for the general comprehension of the content.

The sentences that are exceedingly suggested by different sentences in the content are probably going to be more useful for the given content, and will be in this manner given a higher score. A similarity can be likewise drawn with PageRank's "irregular surfer display", where a client surfs the Web by taking after connections from any given Web page. With regards to content displaying, TextRank actualises what we allude to as "content surfing", which identifies with the idea of content attachment (Halliday and Hasan, 1976): from a specific idea in a content, we are probably going to "take after" connections to associated ideas – that is, ideas that have a connection with the present idea (be that a lexical or semantic connection).

This likewise identifies with the "weaving" marvel (Hobbs, 1974): truths connected with words are partaken in various parts of the talk, and such connections serve to "sew the talk together". Through its iterative system, TextRank goes past basic chart availability, and it can score content units construct additionally in light of the "significance" of other content units they connection to. The content units chose by TextRank for a given application are the ones most suggested by related content units in the content, with inclination given to the proposals made by most compelling ones, i.e. the ones that are thusly exceptionally prescribed by other related units. The basic speculation is that in a durable content section, related content units tend to shape an "Internet" of associations that approximates the model people work about a given setting during the time spent talk understanding.

2.5 Building the Recommendation Engine

Recommendation systems changed the way inanimate websites communicate with their users. Rather than providing a static experience in which users search for and potentially buy products, recommender systems increase interaction to provide a richer experience. Recommender systems identify recommendations autonomously for individual users based on past purchases and searches, and on other users' behavior.[3]

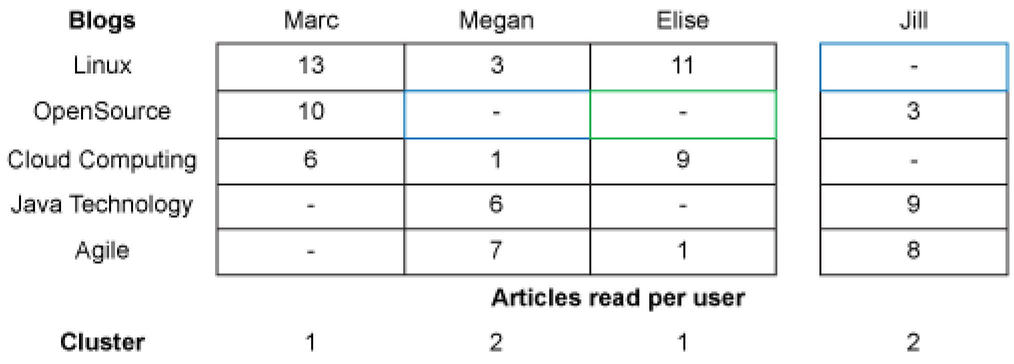
Most recommender systems take either of two basic approaches: collaborative filtering or content-based filtering. Other approaches (such as hybrid approaches) also exist.

2.5.1 Collaborative Filtering

Collaborative filtering arrives at a recommendation that's based on a model of prior user behavior. The model can be constructed solely from a single user's behavior or — more effectively — also from the behavior of other users who have similar traits. When it takes other users' behavior into account, collaborative filtering uses group knowledge to form a recommendation based on like users. In essence, recommendations are based on an automatic collaboration of multiple users and filtered on those who exhibit similar preferences or behaviors.

For example, suppose you're building a website to recommend blogs. By using the information from many users who subscribe to and read blogs, you can group those users based on their preferences. For example, you can group together users who read several of the same blogs. From this information, you identify the most popular blogs that are read by that group. Then — for a particular user in the group — you recommend the most popular blog that he or she neither reads nor subscribes to.

In the Figure - 2.7 below, a set of blogs forms the rows, and the columns define the users. The intersection of blog and user contains the number of articles read by that user of that blog. By clustering the users based on their reading habits (for example, by using a *nearest-neighbor* algorithm), you can see two clusters of two users each. Note the similarities in the reading habits of the members of each cluster: Marc and Elise, who both read several articles about Linux® and cloud computing, form Cluster 1. In Cluster 2 reside Megan and Jill, who both read several articles about Java™ and agile.



Blogs	Marc	Megan	Elise	Jill
Linux	13	3	11	-
OpenSource	10	-	-	3
Cloud Computing	6	1	9	-
Java Technology	-	6	-	9
Agile	-	7	1	8
		Articles read per user		
Cluster	1	2	1	2

Figure - 2.7

Sample table showing interests of users

Now you can identify some differences within each cluster and make meaningful recommendations. In Cluster 1, Marc read 10 open source blog articles, and Elise read none;

Elise read one agile blog, and Marc read none. In table, then, one recommendation for Elise is the open source blog. No recommendations can be made for Marc because the small difference between him and Elise in agile blog reads would likely be filtered away. In Cluster 2, Jill read three open source blogs, and Elise read none; Elise read 11 Linux blogs, and Jill read none. Cluster 2, then, carries a pair of recommendations: the Linux blog for Jill and the open source blog for Megan.

Another way to view these relationships is based on their similarities and differences, as illustrated in the Venn diagram in Figure - 2.7 The similarities define (based on the particular algorithm used) how to group users who have similar interests. The differences are opportunities that can be used for recommendation — applied through a filter of popularity, for example.

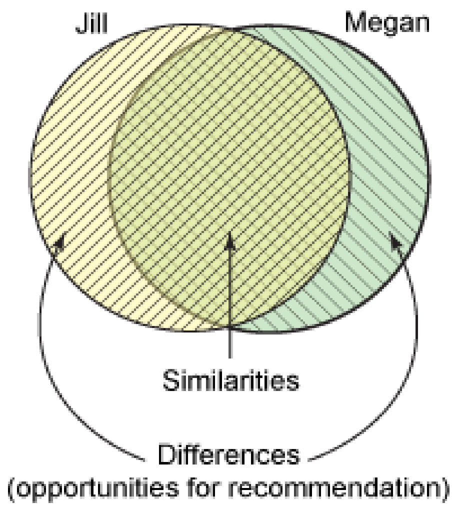


Figure - 2.7

Venn diagram of collaborative filtering

2.6 Discussion

TextRank prevails with regards to recognising the most critical sentences in a content in light of data only drawn from the content itself. Not at all like other managed frameworks, which endeavour to realise what makes a decent outline via preparing on accumulations of synopses worked for different articles, TextRank is completely unsupervised, and depends just on the offered content to infer an extractive synopsis, which speaks to a summarisation show nearer to what people are doing while creating a conceptual for a given record.

Notice that TextRank goes beyond the sentence “connectivity” in a text.

Another vital part of TextRank is that it gives a positioning over all sentences in a content – which implies that it can be effortlessly adjusted to separating short outlines (features comprising of one sentence), or longer more explicative rundowns, comprising of more than 100 words. We are likewise researching blends of key-expression and sentence extraction strategies as a technique for building short/long outlines. At long last, another favourable position of TextRank over already proposed techniques for building extractive rundowns is the way that it doesn't require preparing corpora, which makes it effectively versatile to different dialects or areas.

2.7 Existing Systems – Similar Applications

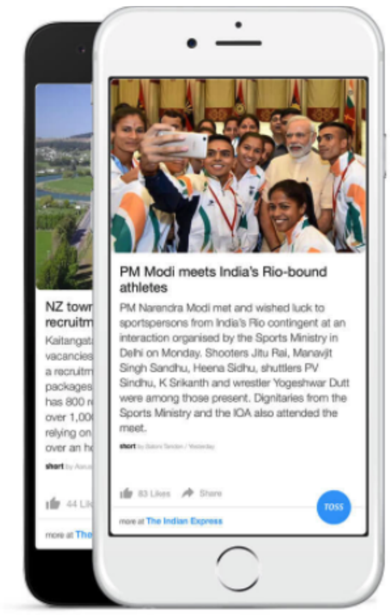
In this section applications that perform in some way sentence based text summarisation will be examined.

2.7.1 Inshorts

Inshorts (previously known as News In Shorts) is a Delhi-based company that is known for its content discovery and distribution application for Android and iOS. It aggregates news and other content such as videos, infographics, and blogs, and summaries them in 60 words or less. Inshorts pushes content to its app, consisting of news and other updates that are each

60 words or less. The company employs a team that manually curates and provides the 60 word summary for each feed that is divided into categories such as national, business, sports, and technology. The app is available for both Android and iOS operating systems.
Figure 2.8 - Inshorts Application

The weakness of this application where our implementation comes is that Inshorts does manual summarisation using their skilled writers and curators. We are planning on removing this middleware and automate the process.


PM Modi meets India's Rio-bound athletes

PM Narendra Modi met and wished luck to sportspersons from India's Rio contingent at an interaction organised by the Sports Ministry in Delhi on Monday. Shooters Jitu Rai, Manavjit Singh Sandhu, Heena Sidhu, shuttlers PV Sindhu, K Srikanth and wrestler Yogeshwar Dutt were among those present. Dignitaries from the Sports Ministry and the IOA also attended the meet.

Figure - 2.8

Screenshot from the app

2.7.2 SUMMARIST

SUMMARIST is an endeavour to create hearty extraction innovation to the extent it can go and afterward proceed with innovative work of methods to perform deliberation. This work confronts the profundity versus vigour tradeoff: either frameworks break down/translate the information profoundly enough to create great outlines (yet are constrained to little application areas), or they work powerfully over pretty much unhindered content (however can't dissect profoundly enough to combine the contribution to a genuine synopsis, and henceforth perform just subject extraction). Specifically, typical procedures, utilising parsers, sentence structures, and semantic representations, don't scale up to genuine size, while Information Retrieval and other factual strategies, being founded on word checking and word bunching, can't make genuine synopses since they work at the word (surface) level rather than at the idea level.

3. SYSTEM DEVELOPMENT

3.1 Design and implementation

a. The final product will be a web application which will be backed by Python(Flask) and Node.js.

b. The front-end of the application will be developed using HTML, CSS, JS, Jinja2 and Ionic.

c. For increasing reliability and reducing response time we'll be using tools like nginx (for load balancing and fast serving of static content) and memcached (for caching sql queries and reducing response time).

d. After completely developing the web application it'll be converted to an android app using ionic framework to provide ease of access to the user.

e. The server will be hosted on cloud like AWS or Openshift.

3.2 Data Flow Diagrams

3.2.1 DFD Level 0

Figure - 3.1

DFD Level 0

3.2.2 DFD Level 1

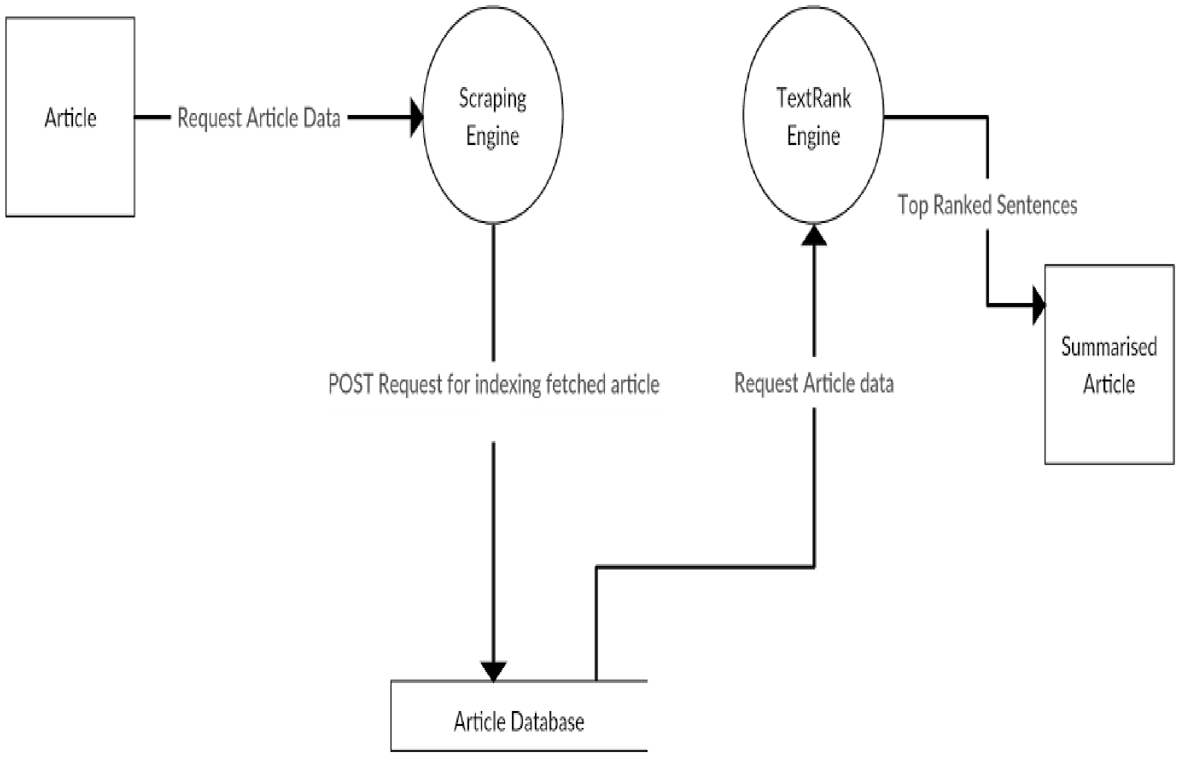


Figure - 3.2

DFD Level 1

3.2.3 DFD Level 2

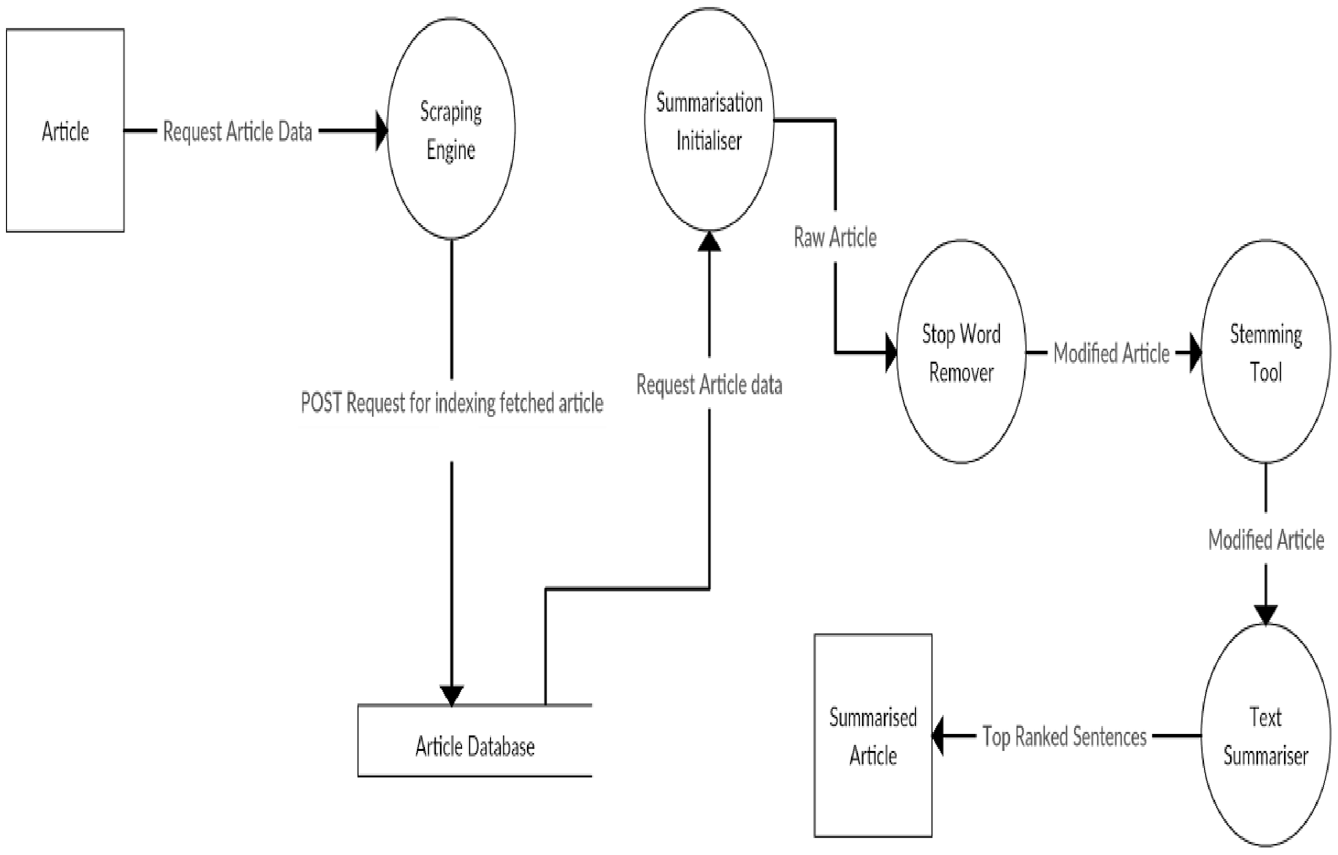


Figure - 3.3

DFD Level 2

3.2.4 DFD Level 3

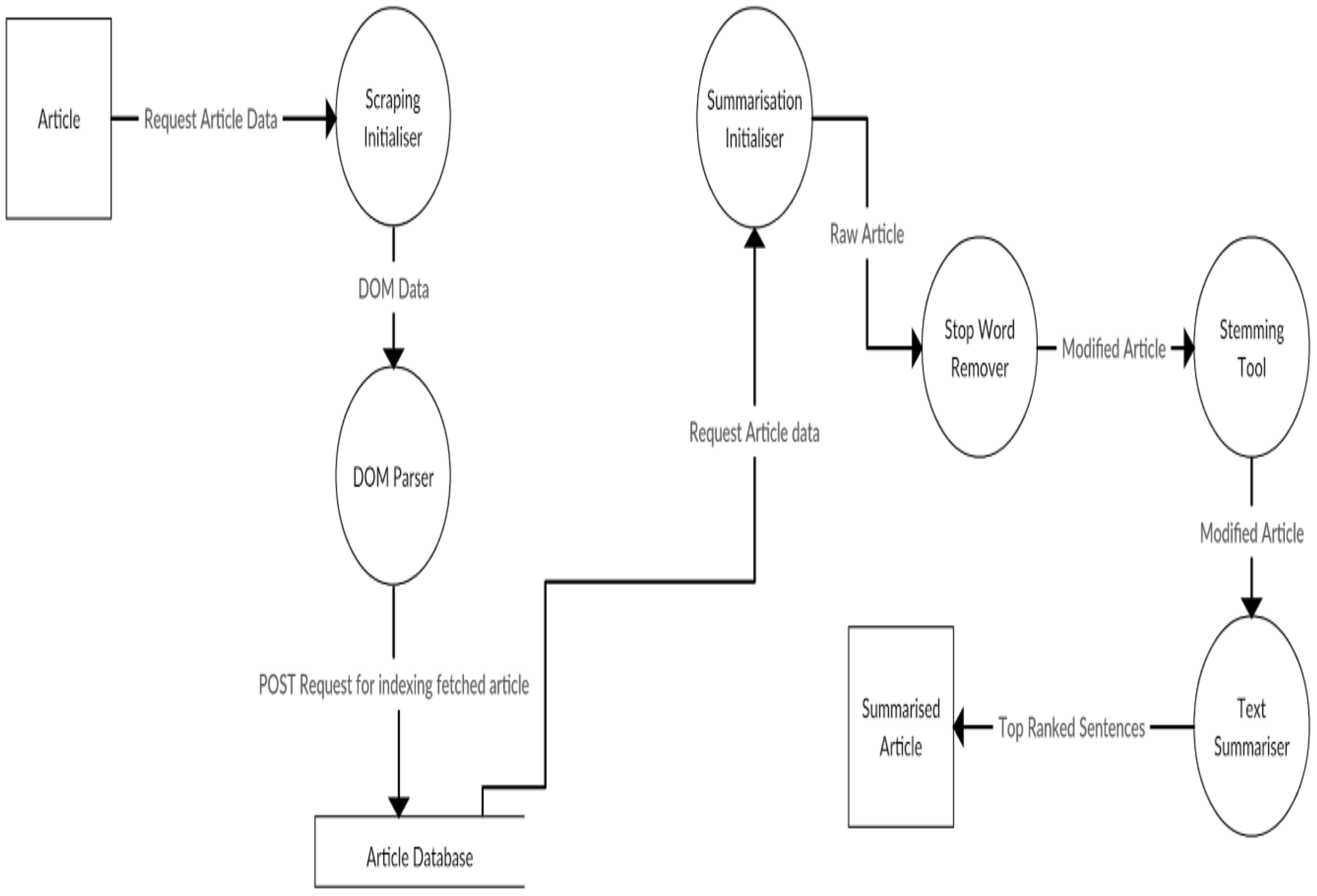


Figure - 3.4

DFD Level 3

3.3 Architecture of the Application and Algorithm

3.3.1 Pseudocode of the Algorithm

```
a. Data to be processed is fetched from the Database.
b. Removal of the Noise is done by following methods:
    (i) Stop-Words like is, are, I etc are removed from the text.
    (ii) The remaining text is passed through a Stemmer.
c. The noise free text is passed through the Text-Rank Algorithm.
d. Each sentence is considered as a node of a fully-connected graph.
e. The weights of the graph are calculated using Jaccard's Similarity index.
f. Top 30% sentences with highest weights in each paragraph are selected.
g. Adding all these selected sentences together makes the complete summary.
h. Generated Summary is stored back in Database.
```

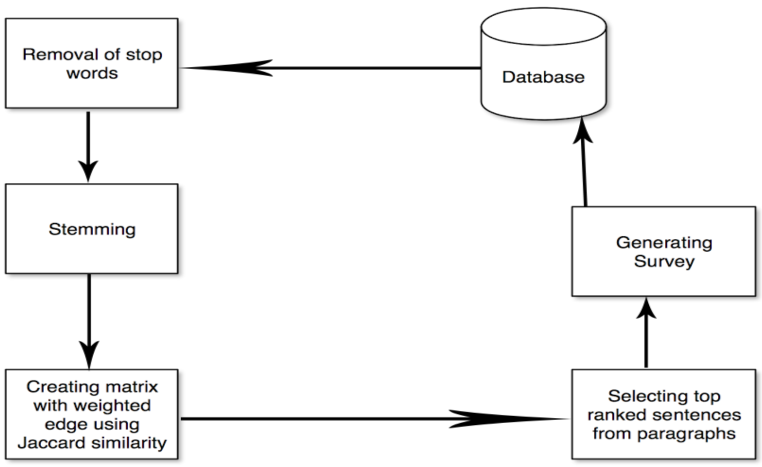

Removal of stop words
Database
Stemming
Generating Survey
Creating matrix with weighted edge using Jaccard similarity
Selecting top ranked sentences from paragraphs

Figure - 3.5

Pseudocode

3.3.2 Web Application Architecture

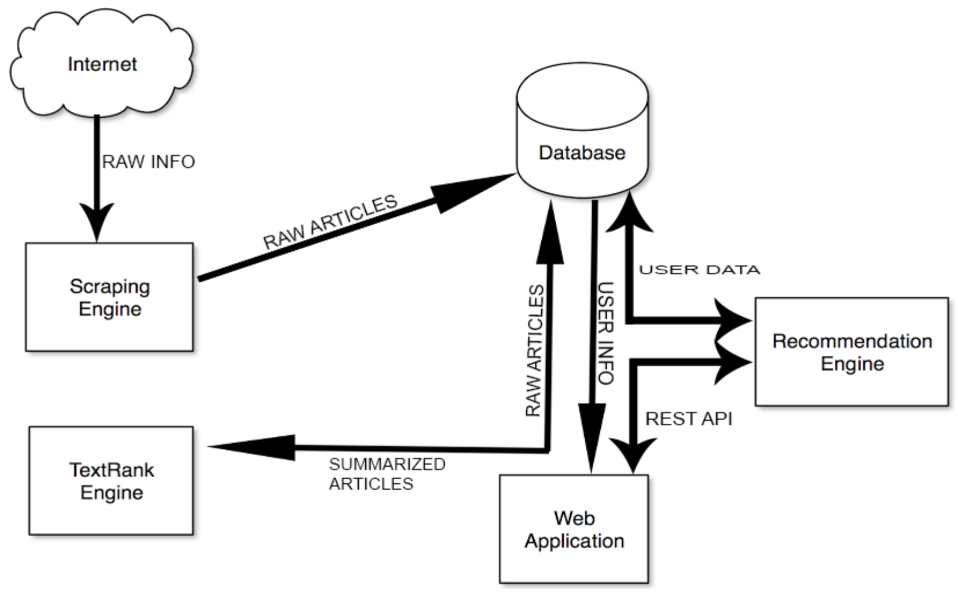
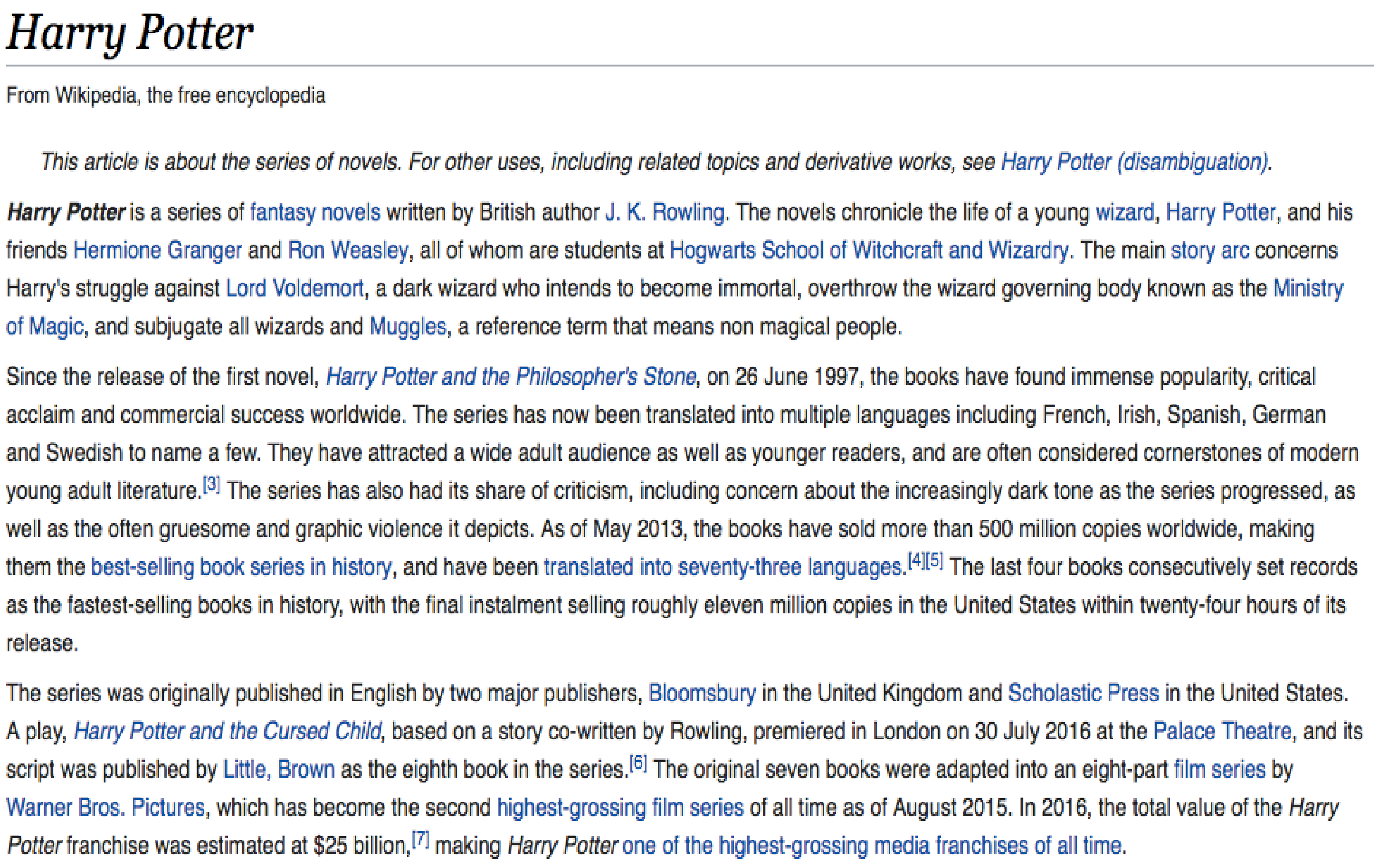


Figure - 3.6

Architecture of the web app

3.4 Sample Inputs and Outputs

3.4.1 Sample Input from Wikipedia



Harry Potter

From Wikipedia, the free encyclopedia

This article is about the series of novels. For other uses, including related topics and derivative works, see Harry Potter (disambiguation).

Harry Potter is a series of fantasy novels written by British author J. K. Rowling. The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermione Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry. The main story arc concerns Harry's struggle against Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and Muggles, a reference term that means non magical people.

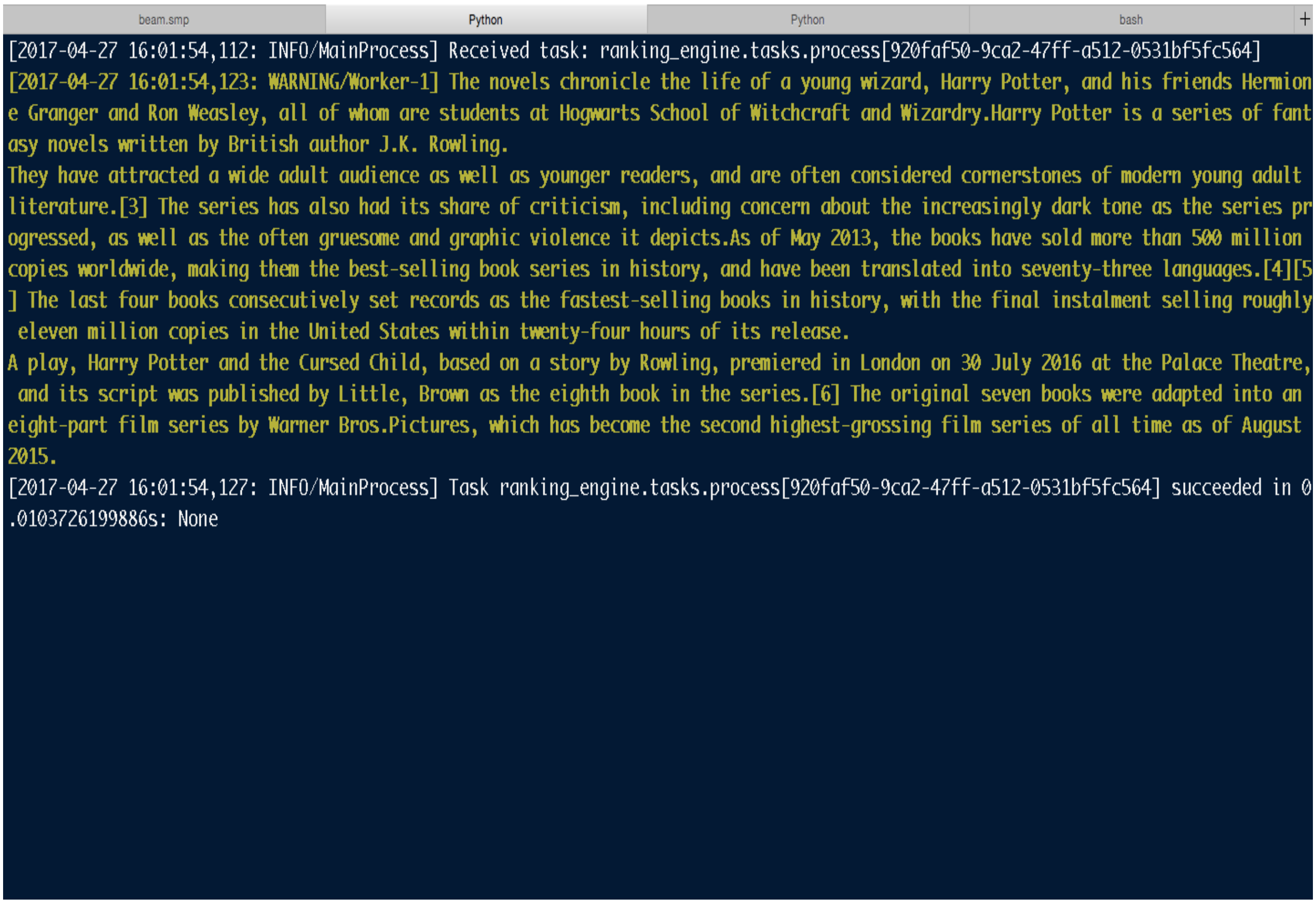
Since the release of the first novel, *Harry Potter and the Philosopher's Stone*, on 26 June 1997, the books have found immense popularity, critical acclaim and commercial success worldwide. The series has now been translated into multiple languages including French, Irish, Spanish, German and Swedish to name a few. They have attracted a wide adult audience as well as younger readers, and are often considered cornerstones of modern young adult literature.[3] The series has also had its share of criticism, including concern about the increasingly dark tone as the series progressed, as well as the often gruesome and graphic violence it depicts. As of May 2013, the books have sold more than 500 million copies worldwide, making them the best-selling book series in history, and have been translated into seventy-three languages.[4][5] The last four books consecutively set records as the fastest-selling books in history, with the final instalment selling roughly eleven million copies in the United States within twenty-four hours of its release.

The series was originally published in English by two major publishers, Bloomsbury in the United Kingdom and Scholastic Press in the United States. A play, *Harry Potter and the Cursed Child*, based on a story co-written by Rowling, premiered in London on 30 July 2016 at the Palace Theatre, and its script was published by Little, Brown as the eighth book in the series.[6] The original seven books were adapted into an eight-part film series by Warner Bros. Pictures, which has become the second highest-grossing film series of all time as of August 2015. In 2016, the total value of the *Harry Potter* franchise was estimated at $25 billion,[7] making *Harry Potter* one of the highest-grossing media franchises of all time.

Figure - 3.7

Sample Text input from Wikipedia

3.4.2 Sample Output generated by the software



```
[2017-04-27 16:01:54,112: INFO/MainProcess] Received task: ranking_engine.tasks.process[920faf50-9ca2-47ff-a512-0531bf5fc564]
[2017-04-27 16:01:54,123: WARNING/Worker-1] The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermion
e Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry.Harry Potter is a series of fant
asy novels written by British author J.K. Rowling.
They have attracted a wide adult audience as well as younger readers, and are often considered cornerstones of modern young adult
literature.[3] The series has also had its share of criticism, including concern about the increasingly dark tone as the series pr
ogressed, as well as the often gruesome and graphic violence it depicts.As of May 2013, the books have sold more than 500 million
copies worldwide, making them the best-selling book series in history, and have been translated into seventy-three languages.[4][5
] The last four books consecutively set records as the fastest-selling books in history, with the final instalment selling roughly
 eleven million copies in the United States within twenty-four hours of its release.
A play, Harry Potter and the Cursed Child, based on a story by Rowling, premiered in London on 30 July 2016 at the Palace Theatre,
 and its script was published by Little, Brown as the eighth book in the series.[6] The original seven books were adapted into an
eight-part film series by Warner Bros.Pictures, which has become the second highest-grossing film series of all time as of August
2015.
[2017-04-27 16:01:54,127: INFO/MainProcess] Task ranking_engine.tasks.process[920faf50-9ca2-47ff-a512-0531bf5fc564] succeeded in 0
.0103726199886s: None
```

Figure - 3.8

Sample Text output generated by the Software

4. PERFORMANCE ANALYSIS

We evaluate the TextRank sentence extraction algorithm on a single-document summarisation task, using 10 news articles taken from online blogs. For each article, TextRank generates a summary — the task undertaken by other systems participating in this single document summarisation task. For evaluation, we are using manually produced reference summaries.

We evaluate the summaries produced by TextRank using each of the three graph-based ranking algorithms described above and show the results obtained with each algorithm, when using graphs that are: undirected.

Algorithm	**Undirected Graph Score**
PageRank	0.4904
HITS	0.4912
POS	0.4878

Among all algorithms, the H I T S_A and PageRank algorithms provide the best performance.

This proves that graph-based ranking algorithms, previously found successful in Web link analysis, can be turned into a state-of-the-art tool for sentence extraction when applied to graphs extracted from texts.

5. CONCLUSIONS AND FUTURE WORK

As a rule, TextRank calculation accomplished preferable outcomes over an arbitrary procedure of choice. This affirms the speculation set forward toward the start of our paper. Despite the fact that TextRank delivered humble outcomes when it came to imitate the execution of human subjects at recognising the most solid proclamations, it holds extraordinary guarantee as an apparatus for supporting manual validity evaluation, since it gives an extractive outline which not just incorporate same of the most educational articulations, additionally chooses those that can be discarded because of human predisposition, and since test information has demonstrated that individuals tend to correspond significance with believability, the calculation may serve as a technique for beating this inclination while get ready information for manual appraisal. At the point when utilised for recognising believable and educational archives, TextRank fared moderately better in selecting the most useful reports. It was likewise equivalent to human evaluator when utilising significance rankings as a strategy for foreseeing mean sentence validity per a given report.

The future objective of this project is to present a proposal framework for the whole application which will fill in as a different API recommending the clients related articles which will utilise significant things like ML and Collaborative Filtering.

Another issue that we can simply work around is the fact that we are calculating new recommendations every time a user makes or changes their ratings for movies. Instead of doing recalculations on-the-fly in real time, we should queue these recommendation update requests for the users and perform them behind the scene - perhaps setting a timed refresh interval.

Besides these “technical” choices, there are also some strategic choices that can be made to improve the recommendations. As the number of items and users grow, it will become increasingly costly (in terms of time and system resources) to generate recommendations. It is possible to make this faster by choosing only a subset of users to generate recommendations from, instead of processing the entire database every time. For example, if this was a recommendation engine for restaurants, you could limit the similar user set to contain only those users that live in the same city or state.

Other improvements may involve taking a hybrid approach, where recommendations are generated based on both collaborative filtering and content-based filtering. This would be especially good with content such as movies, where the properties of the content is well defined. Netflix, for example, takes this route, recommending movies based on both other users' activities and the movies' attributes.

6. REFERENCES

[1] Rada Mihalcea, “Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarisation”, ACLdemo 2004

[2] Rada Mihalcea and Paul Tarau, “TextRank: Bringing Order into Texts” - 2004

[3] B. Sarwar, G. Karypis, J. Konstan and John Riedl, “Item-Based Collaborative Filtering Recommendation Algorithms”, Proceedings of the 10th international conference on World Wide Web 2001: 285-295.

[4] Joel Larocca Neto, Alex A. Freitas, Celso A. A. Kaestner , ”Automatic Text Summarisation using a Machine Learning Approach" ,

[5] Rada Mihalcea, “Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarisation”, ACLdemo 2004

[6] Salton, A. Singhal, M. Mitra, and C. Buckley. 1997. Automatic text structuring and summarisation. Information Processing and Management 2(32).

[7] Teufel and M. Moens. 1997. Sentence extraction as a classification task. In ACL/EACL workshop on ”Intelligent and scalable Text summarisation”, pages 58–65, Madrid, Spain.

[8] Marcu, D. Discourse trees are good indicators of importance in text. In Mani., I.; Maybury, M. (eds.). Adv. in Automatic Text Summarisation. The MIT Press (1999) 123-136

[9] Mitra, M.; Singhal, A.; Buckley, C. Automatic text summarisation by paragraph extraction. In Proceedings of the ACL’97/EACL’97 Workshop on Intelligent Scalable Text Summarisation. Madrid (1997)

[10] Nevill-Manning, C. G. ; Witten, I. H. Paynter, G. W. et al. KEA: Practical Automatic Keyphrase Extraction. ACM DL 1999 (1999) 254-255

[11] . Rath, G. J. ; Resnick A. ; Savvage R. The formation of abstracts by the selection of sentences. American Documentation 12 (2) (1961) 139-141

[12] Strzalkowski , T.; Stein, G.; Wang, J.; Wise, B. A Robust Practical Text Summarizer. In Mani, I.; Maybury, M. (eds.), Adv. in Autom. Text Summarisation. The MIT Press (1999)

[13] Teufel, S.; Moens, M. Argumentative classification of extracted sentences as a first step towards flexible abstracting. In Mani, I.; Maybury M. (eds.). Advances in automatic text summarisation. The MIT Press (1999)

[14] Brill, E. A simple rule-based part-of-speech tagger. In Proceedings of the Third Conference on Applied Comp. Linguistics. Assoc. for Computational Linguistics (1992)