DOCUMENT SUMMARIZATION USING LATENT SEMANTIC ANALYSIS

Divyarsha Koduri (DXK190007) Rani Pravalika Kandili (RXK190016)
Theja Kunam (TXK190012)

INTRODUCTION

Nowadays, Machine Learning and Natural Language Processing Algorithms are widely used for predicting and analyzing the behavior of various datasets. Such datasets and analysis help us a lot to understand the behavior of a customer and thereby provide a customized experience to every individual. Based on this, we are working on a Kaggle dataset to analyze the customer reviews of Amazon Fine Food. The aim of our project is to build a model that helps in summarizing huge number of customer reviews about a food and thereby provide an overview of it. The major tasks of our project includes data collection, implementing Latent Semantic Analysis (LSA) algorithm and TextRank Algorithm for text summarization. LSA is a technique used for creating vectorized representation of texts which are asserted to apprehend their semantic content. The main function of LSA is to calculate the resemblance of text pairs by semantics. TextRank is a graph based ranking model for text processing which can be used to find the most appropriate sentences in text and also to find keywords.

ABSTRACT

Latent semantic analysis (LSA) is a technique in natural language processing, in particular distributional semantics, of analyzing relationships between a set of documents and the terms they contain by producing a set of concepts related to the documents and terms. Latent semantic analysis (originally Called Latent Semantic Indexing) was implemented for the mission of Information Retrieval. It is picking from huge database of documents complement the given search query. query. Latent sematic Analysis broadens the vector-based method by using Singular Value Decomposition to restructure data. The information about this process is explained below. The assumption is that there is a set of latent variables which have the meanings that can be expressed in some language. The above-mentioned variables are independent. Singular Value Decomposition (SVD) is a technique used in vector space of matrix algebra which essentially re-structures and gives ranks to the dimensions in the vector space. As the measurements in a vector space calculated by Singular Value Decomposition are structured as highest to lowest significant, the minimized representation is testified to be the best possible for that dimensionality. In Latent semantic analysis, the typical presupposition is that only the top 350 dimensions out of many available dimension are required for obtaining the meaning of texts. By establishing the representations on a decreased number of dimensions, words are assigned to similar vectors based on the context they occur and thereby getting a high similarity rating. The dimensions that are ignored are presumed to random associations, noise, or any other not so crucial factor. This analysis performed on documents for Information retrieval through LSA is much better than the available contemporary methods. It is very surprising to notice that this analysis models the behavior of human on a heterogeneity of linguistic deeds. Available most routine steps in LSA are described below. Collection of huge data/text and classify them into documents. In many applications, every paragraph in a document is assumed to be an individual document on the basis of the assumption that the information in the various paragraphs of the same document will have some relation. After that, a co-concurrence matrix of terms and documents is created. A term means a word that can appear more than one time in more than one document and various morphological analysis or no stemming will be executed to merge the various forms of the word. Each term y has the information about the number of times of occurrence of the term y of each cell and each cell in the co-concurrence matrix represents the document x. An m-dimensional vector of each term and n-dimensional vector for each term means that there are m terms and n number of documents for each term. On the basis of Information theory, values in every cell can be given a weight to lessen the effect of frequent/common words.

Text Rank Summarization algorithm works on constructing a graph by creating vertex for every sentences in the document. After the graph is constructed, a stochastic matrix is formed with

the help of a damping factor and the ranking is done by finding the eigen vectors of the eigen values.

We create a graph and use the current vertex and the other vertices as input and create a adjacency list of every vertex based on similarity between the vertex with other vertices. When there is no common words, no edge would be there between them and if they have common words, a weighted edge would be created where the weight on the edge would be equal to the similarity.

BACKGROUND WORK

Automated Text summarization is very difficult and challenging, since when we as people sum up a bit of text, we as a rule read it completely to build up our agreement, and afterward compose a synopsis featuring its gist. Since PCs need human information and language capacity, it makes programmed text outline a troublesome and non-insignificant undertaking. Prior way to this job involved vectorial representation, keyword-matching, weighted keyword matching on the basis of appearance of terms in documents.

THEORITICAL AND CONCEPTUAL STUDY

Now that we have learnt about LSA, we need to discover a way to insert the documents in the Vector Space which encrypts their theoretical content of the document. In LSA, we can implement this using Matrix Linear Algebra. The following are the assumptions we have before starting the analysis:
1) Bag of words model being used in LSA
2) Order of the words in the documents will not affect how it is inserted in the vector space we created.
3) Each document in a single vector notation is about a single object/product.
Vector space: It is described in terms of dictionary/ vocabulary. Every word in the dictionary/vocabulary as its own definite orthogonal dimension.
For example: Let's consider Dr Seuss story "*Green Eggs and Ham*", this needs to be converted to lowercase and stopwords are to be removed after which stemming is performed. Results are as follows: boat, box, car, dark, eat, egg, fox, goat, good, green, ham, mouse, rain, sammy, train, tree
Now the Vector Space has 16 dimensions associated with one of these terms. Each document that contains sentences taken from the story shall be described as a 16- dimensional vector. This method assumes by default that the 16 words in dictionary /vocabulary are perpendicular/orthogonal to each other which explains the theory that the presence of a particular word in a document is not dependent on the presence of remaining words in the dictionary/vocabulary. After the vector space is defined, now we must look into how to count the words that is we need to find a way to determine the score of each word present in the dictionary/vocabulary. Easiest way to do this will be going through every document and count the number of occurrences of every dictionary word.

Example: I like eggs and green ham (after pre processing) would be {green, egg, ham}. Output of counting occurrences and inserting would be
$(0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0)$

Example: Here they are!!! Eat them!!! Eat them!!!! Would be {eat, eat} and inserted as
$(0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$.

Other than counting there are better ways to inserting into the documents which are known as **Weighting schemes**. Here they use term frequency and inverse term frequency into consideration and normalize the data. Term Frequency means the frequency of the occurrence of each and every word in the dictionary/vocabulary. This can be counted in 2 ways: simply counting the frequency of words or giving a 1 notation if the word is used else 0. The first method proved to be better than the latter. Every word is not equal in the information content it carries. Terms/words which occurs in every document would give very little information about it. But when a rare word in a document appears might help a lot in deciding about the information content which in particular refers to inverse document frequency.

After applying one of the two schemes, we need to normalize the vector of the document to transform it to unit length. This helps in reducing the calculating time where we might use cosine similarity between document where the cosine rule gives

$$\frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}||\mathbf{b}|} = \cos\theta$$

So, from here we can use dot product and compute the cosine of the angle between the vectors of the document. Now, we need to define the subspace of our documents by using vectors of our documents. We do this by defining a **TERM FREQUENCY_INVERSE DOCUMENT FREQUENCY (TF_IDF)** matrix with the terms/words in the dictionary as rows and the document vectors as columns. Now we give rank to this matrix by considering the size of the dictionary and the total number of documents. The rank indicates the number of parameters needed to successfully describe the document under consideration. This matrix would be further factorized with the help of Singular Value Decomposition (SVD). This decomposition transforms the input matrix into three components namely **U**, **Σ**, and **V**, such that **X = U Σ V^T^**, where **V^T^** is the transpose of a matrix **V**, **Σ** is a diagonal matrix uniquely defined by **input matrix**.

Let's assume the following matrix as the TF_IDF matrix.

$$\mathbf{M} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 1 \end{bmatrix}$$

where we have two terms and three documents. Now we do SVD on the matrix M and obtain all the values as 2 decimal places. The following is the pictorial representation. The white circles are the real positions of our documents and the black dots are their positions on our best rank-1 approximation to the original space. The singular value decomposition tells us the best way of fitting the white points onto a line while losing the minimum of information

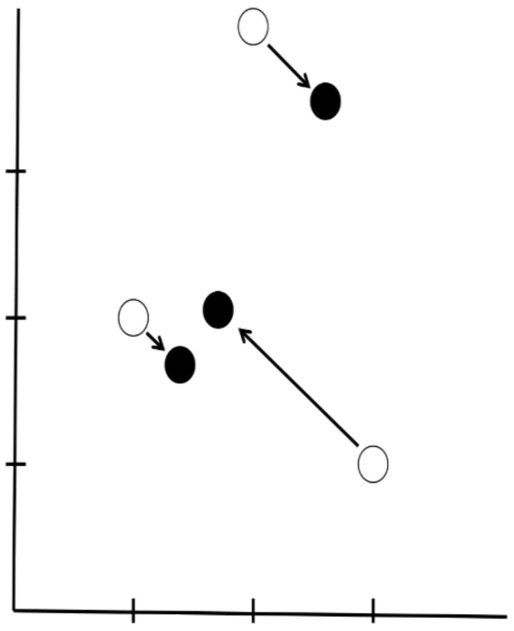


Fig1: Graphical Representation of LSA

We can compute the coordinates of our new matrix (the black points).The black points that was at (1, 2) moves to a position 2.19 along the rank 1 subspace, the point that was at (2, 4) moves to a position 4.39 along this line, and the point that was at (3, 1) moves to a position 2.62 along this line.

Text Rank algorithm works on constructing a graph by creating vertex for every sentence in the document. After the graph is constructed, a stochastic matrix is formed with the help of a damping factor and the ranking is done by finding the eigen vectors of the eigen values.

In our project, the data preparation is same as the Latent Sematic Analysis until we produce a wordlistRDD where every wordlist represents the vertex of the matrix.
Then to get the edges of the vertex, We will use a graphRDD which uses worlist(vertex) and the other vertices as input and create a adjacency list of every vertex based on similarity between the vertex with other vertices. When there is no common words, no edge would be there between them and if they have common words, a weighted edge would be created where the weight on the edge would be equal to the similarity.

ALGORITHMIC IMPLEMENTATION

DataSet Used:

We are using the Kaggle Dataset on Amazon Fine Food Reviews. The following are the details of the data set

Number of Reviews – 568,454

Number of Users -- 256,059

Number of products --74,258

Users with > 50 reviews – 260

Timespan – Oct 1992 - Oct 2012

Also we are using Amazon reviews dataset from snap, Stanford. The following are the details of the data set

Number of Reviews – 34,686,770

Number of Users -- 6,643,669

Number of products --2,441,053

Users with > 50 reviews – 56,772

Timespan – Jun 1995 - Mar 2013

In our implementation, we are considering the Kaggle dataset named Amazon Fine Food Reviews. As part of pre processing we consider the columns ProductId, UserId, Helpfulness Numerator, Helpfulness Denominator, Score, Summary, Text Columns are considered for the summarization.

As part of pre-processing, we group the data with product id and store the data related to every product id into an individual text file which are the individual documents.

We then calculate term-frequency Vector for every sentence selected. From the matrix obtained from Term-frequency, we calculate document-frequency vector. We then proceed to calculating Inverse document frequency from the document-frequency vector obtained and the total number of sentences.

From the values obtained till now, we get the TF-IDF matrix by multiplying IDF with Term Frequency matrix by using the numpy module of python. The TF-IDF matrix has words as rows and the sentences as columns. TF_IDF matrix is then factorized by using Singular Value Decomposition with the help of numpy module and thereby collect the important sentences from the singular matrix. During the decomposition, we obtain three matrices namely U, S and transpose of V.

U - left singular vector matrix,

S - diagonal matrix of non-negative singular values sorted in descending order

V-Transpose -right singular vector matrix.

LSA method works on choosing k concepts out of the V-Transpose matrix and from each concept, it selects the sentence with the largest value to the summary.

We chose 10 concepts with 5 top sentences each to interpret in a better way.

Text Rank Summarization algorithm works on constructing a graph by creating vertex for every sentence in the document. After the graph is constructed, a stochastic matrix is formed with the help of a damping factor and the ranking is done by finding the eigen vectors of the eigen values.
In our project, the data preparation is same as the Latent Sematic Analysis until we produce a wordlistRDD where every wordlist represents the vertex of the matrix.
Then to get the edges of the vertex, We will use a graphRDD which uses worlist(vertex) and the other vertices as input and create a adjacency list of every vertex based on similarity between the vertex with other vertices. When there is no common words, no edge would be there between them and if they have common words, a weighted edge would be created where the weight on the edge would be equal to the similarity.
To calculate the similarity, the following formula is used

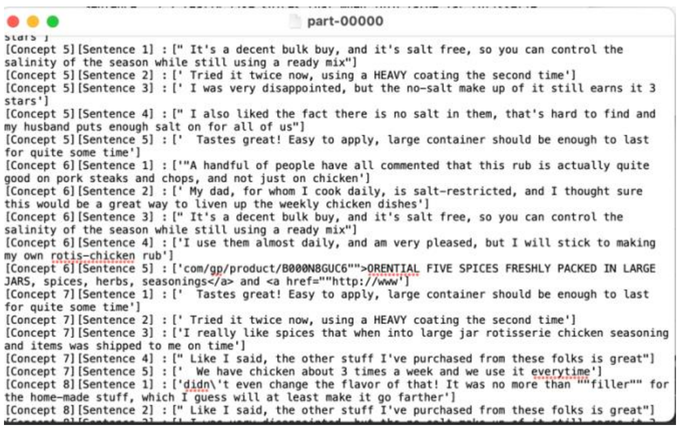
$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \& w_k \in S_j\}|}{log(|S_i|) + log(|S_j|)}$$

After the required graph is constructed, we implement the Text Rank to calculate the rank of every vertex.
Then we selected the top k ranked sentences and add them to the final summary.
The text rank is the modified version of PageRank and the expression to calculate the same is as below

$$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j)$$

RESULTS AND ANALYSIS

When we tried the Latent Sematic Analysis algorithm on all sentences with no filter applied i.e., size >30 and <10, it gave results which had very little number of words thus making no real sense. After applying the filter of number of words greater than 10, the results obtained had long sentences with multiple concepts and made sense because they had more words and have the tendency to represent many concepts. Generally negative reviews are less compared to the positive reviews and thus they didn't show much in the top concepts. Also negative reviews are user specific. Text rank prefers sentences that has more words and edges and hence it is stable and unbiased. Both Text Rank and Latent Sematic Analysis can be the Step 1 of any summarization task.



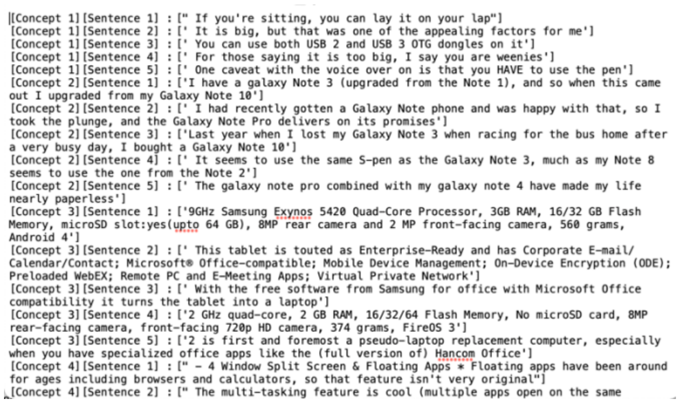
```
stars']
[Concept 5][Sentence 1] : [" It's a decent bulk buy, and it's salt free, so you can control the
salinity of the season while still using a ready mix"]
[Concept 5][Sentence 2] : [' Tried it twice now, using a HEAVY coating the second time']
[Concept 5][Sentence 3] : [' I was very disappointed, but the no-salt make up of it still earns it 3
stars']
[Concept 5][Sentence 4] : [" I also liked the fact there is no salt in them, that's hard to find and
my husband puts enough salt on for all of us"]
[Concept 5][Sentence 5] : ['  Tastes great! Easy to apply, large container should be enough to last
for quite some time']
[Concept 6][Sentence 1] : ['"A handful of people have all commented that this rub is actually quite
good on pork steaks and chops, and not just on chicken']
[Concept 6][Sentence 2] : [' My dad, for whom I cook daily, is salt-restricted, and I thought sure
this would be a great way to liven up the weekly chicken dishes']
[Concept 6][Sentence 3] : [" It's a decent bulk buy, and it's salt free, so you can control the
salinity of the season while still using a ready mix"]
[Concept 6][Sentence 4] : ['I use them almost daily, and am very pleased, but I will stick to making
my own rotis-chicken rub']
[Concept 6][Sentence 5] : ['com/gp/product/B000N8GUC6"">ORENTIAL FIVE SPICES FRESHLY PACKED IN LARGE
JARS, spices, herbs, seasonings</a> and <a href=""http://www']
[Concept 7][Sentence 1] : ['  Tastes great! Easy to apply, large container should be enough to last
for quite some time']
[Concept 7][Sentence 2] : [' Tried it twice now, using a HEAVY coating the second time']
[Concept 7][Sentence 3] : ['I really like spices that when into large jar rotisserie chicken seasoning
and items was shipped to me on time']
[Concept 7][Sentence 4] : [" Like I said, the other stuff I've purchased from these folks is great"]
[Concept 7][Sentence 5] : ['  We have chicken about 3 times a week and we use it everytime']
[Concept 8][Sentence 1] : ['didn\'t even change the flavor of that! It was no more than ""filler"" for
the home-made stuff, which I guess will at least make it go farther']
[Concept 8][Sentence 2] : [" Like I said, the other stuff I've purchased from these folks is great"]
```

Fig2 : Output of LSA program on Food reviews dataset

The reviews in each food product text file is summarized and ordered by each user.

The actual review given by a user is '"Maybe I'm too picky, but I won't be buying this again. I ordered 3 different items at the same time. The other two have been awesome; this one, not so much. Tried it twice now, using a HEAVY coating the second time. Just has no flavor. I was very disappointed, but the no-salt make up of it still earns it 3 stars. My dad, for whom I cook daily, is salt-restricted, and I thought sure this would be a great way to liven up the weekly chicken dishes. Nope. I wound up dumping the whole bottle into my home-made canister of rub...didn't even change the flavor of that! It was no more than ""filler"" for the home-made stuff, which I guess will at least make it go farther. Like I said, the other stuff I've purchased from these folks is great...I use them almost daily, and am very pleased, but I will stick to making my own rotis-chicken rub."

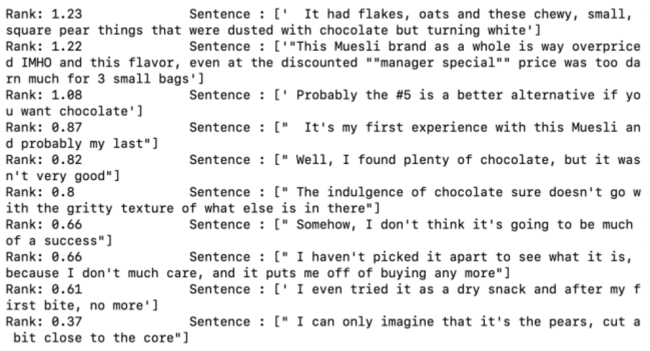
The LSA program condenses the above review to just “I was very disappointed, but the no-salt make up of it still earns it 3 stars” which is an accurate summary of the whole review given but in less fewer words.



```
[Concept 1][Sentence 1] : [" If you're sitting, you can lay it on your lap"]
[Concept 1][Sentence 2] : [' It is big, but that was one of the appealing factors for me']
[Concept 1][Sentence 3] : [' You can use both USB 2 and USB 3 OTG dongles on it']
[Concept 1][Sentence 4] : [' For those saying it is too big, I say you are weenies']
[Concept 1][Sentence 5] : [' One caveat with the voice over on is that you HAVE to use the pen']
[Concept 2][Sentence 1] : ['I have a galaxy Note 3 (upgraded from the Note 1), and so when this came
out I upgraded from my Galaxy Note 10']
[Concept 2][Sentence 2] : [' I had recently gotten a Galaxy Note phone and was happy with that, so I
took the plunge, and the Galaxy Note Pro delivers on its promises']
[Concept 2][Sentence 3] : ['Last year when I lost my Galaxy Note 3 when racing for the bus home after
a very busy day, I bought a Galaxy Note 10']
[Concept 2][Sentence 4] : [' It seems to use the same S-pen as the Galaxy Note 3, much as my Note 8
seems to use the one from the Note 2']
[Concept 2][Sentence 5] : [' The galaxy note pro combined with my galaxy note 4 have made my life
nearly paperless']
[Concept 3][Sentence 1] : ['9GHz Samsung Exynos 5420 Quad-Core Processor, 3GB RAM, 16/32 GB Flash
Memory, microSD slot:yes(upto 64 GB), 8MP rear camera and 2 MP front-facing camera, 560 grams,
Android 4']
[Concept 3][Sentence 2] : [' This tablet is touted as Enterprise-Ready and has Corporate E-mail/
Calendar/Contact; Microsoft® Office-compatible; Mobile Device Management; On-Device Encryption (ODE);
Preloaded WebEX; Remote PC and E-Meeting Apps; Virtual Private Network']
[Concept 3][Sentence 3] : [' With the free software from Samsung for office with Microsoft Office
compatibility it turns the tablet into a laptop']
[Concept 3][Sentence 4] : ['2 GHz quad-core, 2 GB RAM, 16/32/64 Flash Memory, No microSD card, 8MP
rear-facing camera, front-facing 720p HD camera, 374 grams, FireOS 3']
[Concept 3][Sentence 5] : ['2 is first and foremost a pseudo-laptop replacement computer, especially
when you have specialized office apps like the (full version of) Hancom Office']
[Concept 4][Sentence 1] : [" - 4 Window Split Screen & Floating Apps * Floating apps have been around
for ages including browsers and calculators, so that feature isn't very original"]
[Concept 4][Sentence 2] : [" The multi-tasking feature is cool (multiple apps open on the same
```

Fig 3: Output of LSA program on Amazon reviews dataset

Similarly, Fig is the output of LSA on the Amazon review dataset. In both the files, the output is divided into concepts. Each concept contains similar summaries of all the reviews.

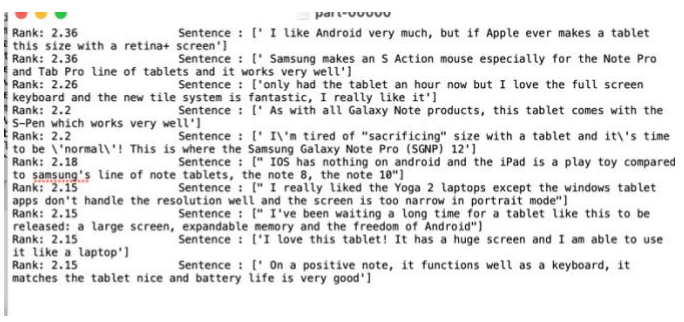


```
Rank: 1.23              Sentence : ['  It had flakes, oats and these chewy, small,
square pear things that were dusted with chocolate but turning white']
Rank: 1.22              Sentence : ['"This Muesli brand as a whole is way overprice
d IMHO and this flavor, even at the discounted ""manager special"" price was too da
rn much for 3 small bags']
Rank: 1.08              Sentence : [' Probably the #5 is a better alternative if yo
u want chocolate']
Rank: 0.87              Sentence : ["  It's my first experience with this Muesli an
d probably my last"]
Rank: 0.82              Sentence : [" Well, I found plenty of chocolate, but it was
n't very good"]
Rank: 0.8               Sentence : [" The indulgence of chocolate sure doesn't go w
ith the gritty texture of what else is in there"]
Rank: 0.66              Sentence : [" Somehow, I don't think it's going to be much
of a success"]
Rank: 0.66              Sentence : [" I haven't picked it apart to see what it is,
because I don't much care, and it puts me off of buying any more"]
Rank: 0.61              Sentence : [' I even tried it as a dry snack and after my f
irst bite, no more']
Rank: 0.37              Sentence : [" I can only imagine that it's the pears, cut a
 bit close to the core"]
```

Fig 4: Output of Textrank on Food reviews dataset

The text rank algorithm gives weighted ranks to each review and based on the parameters and ranks, the reviews are displayed in sorted order. The first 10 weighted ranks are shown in the output. At a glance, we can see the top summaries of the product and what the majority of reviews state.

An important aspect of text rank is that it gives ranking over all sentences in a text so it can be easily adapted to extracting both short and long summaries. Using graphs, textrank identifies connections between entities in text and implements concept of recommendation.



```
Rank: 2.36              Sentence : [' I like Android very much, but if Apple ever makes a tablet
this size with a retina+ screen']
Rank: 2.36              Sentence : [' Samsung makes an S Action mouse especially for the Note Pro
and Tab Pro line of tablets and it works very well']
Rank: 2.26              Sentence : ['only had the tablet an hour now but I love the full screen
keyboard and the new tile system is fantastic, I really like it']
Rank: 2.2               Sentence : [' As with all Galaxy Note products, this tablet comes with the
S-Pen which works very well']
Rank: 2.2               Sentence : [' I\'m tired of "sacrificing" size with a tablet and it\'s time
to be \'normal\'! This is where the Samsung Galaxy Note Pro (SGNP) 12']
Rank: 2.18              Sentence : [" IOS has nothing on android and the iPad is a play toy compared
to samsung's line of note tablets, the note 8, the note 10"]
Rank: 2.15              Sentence : [" I really liked the Yoga 2 laptops except the windows tablet
apps don't handle the resolution well and the screen is too narrow in portrait mode"]
Rank: 2.15              Sentence : [" I've been waiting a long time for a tablet like this to be
released: a large screen, expandable memory and the freedom of Android"]
Rank: 2.15              Sentence : ['I love this tablet! It has a huge screen and I am able to use
it like a laptop']
Rank: 2.15              Sentence : [' On a positive note, it functions well as a keyboard, it
matches the tablet nice and battery life is very good']
```

Fig 5: Output of text rank algorithm on Amazon dataset

To analyze the results, ROUGE-1 can be used to compare the algorithm generated summary with human generated summary.

Based on the frequent words, ROUGE will identify the ratio of correctly identified key words.

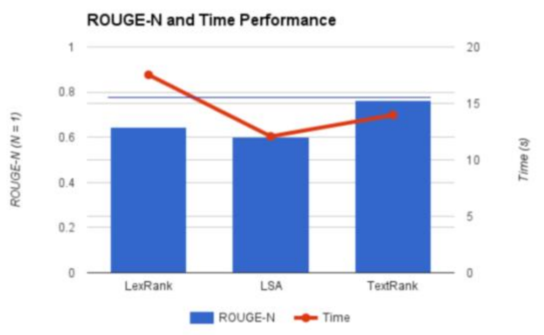


Fig 6: Graph showing evaluation of the three algorithms

From the graph, we can clearly see that TextRank is the best algorithm out of the three. The blue line at y ~ 0.7 indicates the human generated output and text rank generated output is the closest to it.

Since the evaluation of the text summary involves comparing many human written summaries with the machine generated summary by checking unigram co-occurrences, it is tedious to evaluate these two approaches.

CONCLUSION AND FUTURE WORK

The conclusion from the results and comparison is that LSA summarizes and computes the similarities between all the reviews whereas text rank weighs each review and ranks them. From the comparison we can see that Text Rank gives the closest results to human generated summaries. But both algorithms are just the first steps to text summarization.

The future work includes having a hybrid model including both abstractive and extractive text summarizers. Extractive algorithms include LSA, TextRank, kl divergence. For better results, using both models will guarantee a better ROUGE score and a better summarizer.

References

[1] Ping Chen and Rakesh Verma. 2006. A query-based medical information summarization system using ontology knowledge.

In Computer-Based Medical Systems, 2006. CBMS 2006. 19th IEEE International Symposium on. IEEE, 37–42.

[2] Vishal Gupta and Gurpreet Singh Lehal. 2010. A survey of text summarization extractive techniques. Journal of Emerging Technologies in Web Intelligence 2, 3 (2010), 258–268.

[3] Effective extractive summarization using frequency-filtered entity relationship graphs