IMPLEMENTATION OF AUTOMATED TEXT SUMMARY USING MINIMUM DOMINATING SET THEORY

by B A

Submission date: 24-Apr-2020 08:45PM (UTC+0530)
Submission ID: 1306583256
File name: Major_ResPaper_updated_3.pdf (437.84K)
Word count: 2058
Character count: 10297

IMPLEMENTATION OF AUTOMATED TEXT SUMMARY USING MINIMUM DOMINATING SET THEORY

ALIND GUPTA, AKSHIT JINDAL, RUSHIL JAYANT

Applied Mathematics Department Delhi Technological University India
Email: alindgupta99@gmail.com, akshitjindal4@gmail.com, rushildj3@gmail.com

Abstract: The idea is to use concept of Minimum Dominating Set in Graph Theory to develop a technique for text summarization of a document. For this we use the learning procedure of Word Embeddings that maps words and expressions in a text document to vectors of real numbers which helps the algorithm to learn similarity between the words. Then the sentences are vectorized as a whole to give us a weighted complete graph showcasing degree of relation between each sentence. The Minimum Dominating Set of this weighted complete graph gives us the final text summary. This technique is then compared with other existing techniques used by online text summarization websites to conclude its superiority over their techniques.

Keywords: Minimum Dominating Set, Word Embedding, GloVe, Text Summarization, Graph Theory

1. INTRODUCTION

Minimum Dominating Set in a graph gives us the set of vertices which together with their adjacent vertices gives us every vertex of the whole graph as well as the edges between them.

This property makes it very useful for real life applications like minimal grouping.

Suppose you have hazardous chemicals that cannot be stored together. What is the minimum number of storage units you need to store all your chemicals? It can also be used in Scheduling problems.

Suppose that some employees are attending a meeting. Each employee writes down a list of events they want to attend.

What is the smallest number of time slots needed so that everyone can attend all sessions that they want to attend (assuming there are enough rooms)?

They are also useful routing problems.

How many internet routers do you need so that every computer in your business has internet access?

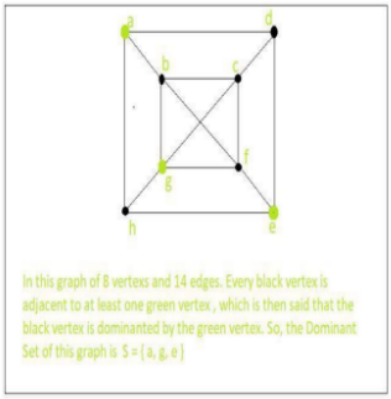
So, we can understand how a Minimum Dominating Set can help us summarize a text document while keeping the core meaning attached to it intact.

Our aim is to introduce a new technique to produce a text summarizer based on this theory and determine if it gives us results which are comparable to existing techniques used by online websites for the same.

2. DETAILED THEORY

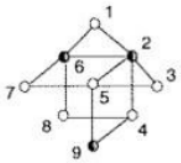
2.1 Dominating set

In graph theory, the definition of a dominating set differs for a simple graph and a weighted graph. For a simple graph G=(V,E), a dominating set D is the set of vertices such that for each vertex v in V, either v is in D or v is adjacent to atleast one element of D. For a weighted graph G, a dominating set D is the set of vertices such that for each vertex v in V, either v is in D or the summation of the weights of the all the edges between v and the elements in D is greater than or equal to a threshold value.



2.2 Minimum Dominating Set

The minimum dominating set for a given graph as the name suggests is defined as the dominating set with the smallest cardinality. The domination number for a given graph is the cardinality of the minimum dominating set of the graph.

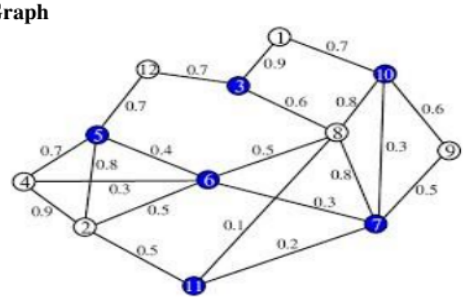


S = {2,6,9}

PN[2,S] = {3}
PN[6,S] = {7,8}
PN[9,S] = {9}

S is minimal dominating

2.3 Minimum Dominating Set of a Weighted Graph



The above diagram shows an example of a weighted graph G=(V,E) with |V| = 12 and |E| = 21 and the set Γ0= {3,5,6,7,10,11} as a generalized dominating set for the given graph.

2.4 Determining the adjacency matrix for the text

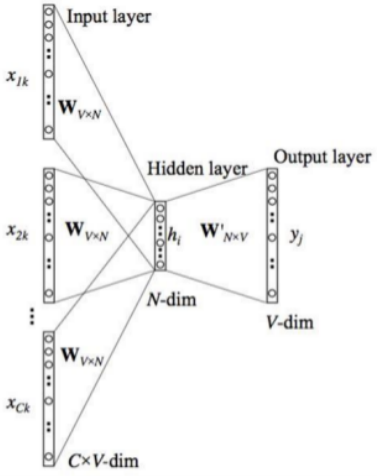
Given a text file our task is to convert it into a graph. For doing so we first find vector form of a sentence.

2.4.1 Word Embeddings

- Word embeddings are an important tool used in many Natural Language Processing tasks.
- Word embeddings are used to map words to high dimensional vectors since many automated tasks work well real numbers or vectors of real numbers rather than characters and strings of characters.
- These high dimensional vectors are such that they convey the semantic meaning of the word. This is achieved through the training process of the word embeddings and is dependent on the context upon which it is trained.

2.4.2 Training Word Embeddings

- For each target word (the word for which the word vector needs to be determined), a group of words is chosen which is called the context for the word.
- The context for each target word is chosen such that the context and the target word occur together very often in the training data.
- The context is given as the input to the neural network. The task for the neural network is for given context, to predict the target word. The prediction error is then backpropagated to adjust the weights.
- To generate the word embeddings, only the right half of the network is retained.
- The hidden layer in the middle returns the word embedding for the target word.



2.4.3 GloVe File

GloVe: This stands for Global Vectors for Word Representation

GloVe is a Machine Learning algorithm which is used to determine vector embeddings for words. Training is performed by feeding large text data into a neural network (whose architecture resembles that of an autoencoder) from a corpus (which is a large collection of text data from a variety of sources), and the resulting word vectors showcase intriguing linear substructures and relationships between different words in the word vector space.

2.5 Forming The adjacency Matrix

Now since we have tool to convert words into vector, so what we do is find vector form of each word in a sentence and then take the mean of these vectors to determine the vector representation of the sentence.

Algorithm to vectorize a sentence

```
Algorithm 1: Vectorize(sentence)
  Result: Vector representation of sentence
  valid=[];
  while word in sentence do
      if word present in glove file then
          valid.add(vector(word));
      else
          ignore;
      end
  end
  return mean(valid)
```

Now taking each sentence as a vertex in the weighted graph, we define the edges between every pair of vertices in the graph, where the weight of each edge is given by the cosine similarity between the two sentences.

The Cosine distance between u and v (where u and v are two vectors), is defined as

$$1 - \frac{u \cdot v}{||u||_2 ||v||_2}.$$

Algorithm to Determine Cosine Similarity between two sentences

```
Algorithm : cosine Similarity(sentence1,sentence2)
  Result: cosine similarity between two sentences
  vector1=vectorize(sentence1);
  vector2=vectorize(sentence2);
  cosine similarity= cosine-Distance(vector1,vector2);
  return cosine similarity;
```

The more similar the sentences are, their cosine similarity return values closer to 1. Similarly, it returns value close to zero if the sentences are not similar.

After finding values of cosine similarity between every pair of sentences, we form adjacency matrix of the graph.

```
Algorithm : generateAdjacencyMatrix(lines)
  Result: Graph representation of text file
  graph = [];
  while line1 in lines do
      row=[];
      while line2 in line do
          cosineDistance = cosineSimilarity(line1,line2);
          row.add(cosineDistance);
      end
      graph.add(row);
  end
  return graph;
```

2.6 Finding the Minimum Dominating Set of the graph

Now we have the graph. The next task is to find the minimum dominating set (MDS).

We will use the formula

$$Z(\beta) = \sum_{c_1,\ldots,c_N} \left[\delta^1_{c_j} e^{-\beta} + \delta^0_{c_j} \Theta\Big(\sum_{i \in \partial j} c_i w_{i,j} - \theta \Big) \right]$$

Where

$$\delta^a_b = \begin{cases} 1 \text{ if } a = b \\ 0 \text{ if } otherwise \end{cases}$$

And

$$\Theta(x) = \begin{cases} x \text{ if } x \geq 0 \\ 0 \text{ if } otherwise \end{cases}$$

This formula is used to determine the configuration which represents the minimum dominating set of the graph by assigning scores to different configurations. The configurations are generated by considering all possible combinations of nodes of the graph. The one which has maximum score can be treated as minimum dominating set of the graph.

2.7 Optimization of the above formula

Since finding configuration score for each of the configuration is a time consuming task with complexity $O(2^n)$,where n is the number of vertices in the graph, we reduce this complexity by pre determining the number of line required in the summary ,which reduces the time complexity to $O({}^nP_r)$ where r is the size of the generated summary.

2.8 Determining the summary of the Text File

The configuration with maximum score is considered to be the minimum dominating set configuration for the graph. The vertices present in the dominating set act as the summary for the input text file.

3. IMPLEMENTATION AND RESULT

3.1 Automatic Text Summarization for small files

In this research paper, we define a "small" text file as a text file such that the graph generated for the input text file consists of less than or equal to 60 nodes. The number 60 has been chosen because for a graph with a maximum of 60 nodes, the tweaked brute-force algorithm to determine the Minimum Dominating Set is practically viable in terms of time complexity. In this section, we will demonstrate how to automatically summarise a "small" text file. In the next section we will demonstrate how to handle the case for "large" text files.

Pseudocode:

```
Algorithm : Summarising a "small" text file
graph = generateAdjacencyMatrix(textFile)
maxConfiguration = MDS(graph)
generatedSsummary = []
i=0;
while i is less than N do
    if maxConfiguration[i]=='1' then
        generatedSummary.add(lines[i]);
    else
        continue;
    end
    i++
end
return(generatedSummary)
```

3.2 Automatic Text Summarization for large text files

To generate the summary for a "large" text file (here, large is defined in the opposite sense of a "small" text file), applying the previously stated algorithm though acceptable in theory, is not practically viable. This is because of the following reasons: -

- Fixing an absolute number for the size of the generated summary makes the summarisation technique non-scalable.
- For very large text files, fixing the size of the generated summary still does not significantly affect the time taken to generate the summary.

To tackle these obstacles, the pre-processing of the text file can be done. The pre-processing of the file can be done in two manners:

- **Reducing the number of nodes in the graph:**

For almost all text files, we can safely assume that sentences which are in close proximity to each other talk about the same subject matter and produce and cosine similarities very close to 1. Therefore, for a large text file, sentences can be grouped together into a single node such that the total number of vertices present in the graph is reduced to within 60.

Pseudocode:

```
Algorithm : Reducing the number of nodes for a text file
numSentencesInNode = int(N/60)+1
reducedNodesLines = []
x=0;
while x is less than N-numSentencesInNode do
    groupedLine = []
    i=0
    while i is less than numSentencesInNode do
        groupedLine.add(lines[x+i])
        i++;
    end
    reducedNodesLines.add(groupedLine)
    x += numSentencesInNode
end
return(reducedNodesLines)
```

- **Dividing the text file into multiple sub-files:**

In case if we are dealing with a 'large" text file which consists of multiple sub-topics, uniformly grouping the sentences can violate our assumption defined in the previous sub-section because sentences at the borderline of two sub-topics in the text file have a significant probability of talking about two different contexts. Therefore, in such a case, grouping the sentences according to the subtopics is a better option.

Pseudocode:

```
Algorithm : Splitting a text file into multiple files
splittedFiles = []
textParser = textFile.start();
while textParser != textFile.end() do
    newFile = createNewFile()
    while textParser != newSubtopic AND textParser != textFile.end() do
        newFile.write(textParser)
        textParser++;
    end
    newFile.close()
    splittedFiles.add(newFile)
end
return(splittedFiles)
```

3.3 Comparison

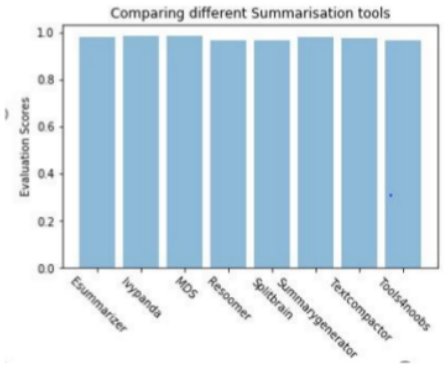
In order to determine the efficiency of our program we compare the summary generated by our program to the rest of the publicly available summarisation tools generated summaries.

In our research we have considered various online platforms tools that generate summary with high efficiency such as Esummarizer, Ivypanda, Resoomer, Splitbrain, Textcompactor etc.

In order to compare these two types of summaries, we will compare them both to the standard summary of the text file. The one which is more similar to the standard summary will be considered better.

The similarity between a automatically generated summary and the human generated summary is the evaluation score for the automatically generated summary which is equal to the cosine similarity between the two.

We compare the evaluation scores of the different generated summarisations and present the results visually



It can be seen that Evaluation score of the MDS (our program generated summary) is the highest. Hence it can be inferred that the MDS generated summary is more similar to the original text than the others.

CONCLUSIONS

After implementing text summarization using our MDS Technique, we conclude the following:

1. A new technique to summarize a text document has been introduced based on Graph Theory.
2. This technique is generating summary which is more accurate than the existing techniques in online text summary generating websites.

ACKNOWLEDGMENTS

We thank Dr. Sangita Kansal, Head of Department, Applied Mathematics, DTU and Mrs. Payal Dabas, Assistant Professor, DTU for their expert guidance throughout the making of this project. We would also like to thank our families and friends for their timely motivation and constant support.

REFERENCES

1. Dominating Sets in Directed Graphs Chaoyi Pang Rui Zhang Qing Zhang Junhu Wang The Australian e-Health Research Centre, ICT Centre, CSIRO, Australia Department of Computer Science and Software Engineering, The University of Melbourne, Australia School of Information and Communication Technology, Gri±th University,
2. Applications of Distance - 2 Dominating Sets of Graph in Networks K.Ameenal Bibi 1, A.Lakshmi2 and R.Jothilakshmi 3 1,2 PG and Research Department of Mathematics, D.K.M College for women (Autonomous), India.
3. GloVe: Global Vectors for Word Representation Jeffrey Pennington, Richard Socher, Christopher D. Manning Computer Science Department, Stanford University, Stanford, CA 94305

★★★

IMPLEMENTATION OF AUTOMATED TEXT SUMMARY USING MINIMUM DOMINATING SET THEORY

ORIGINALITY REPORT

11%	2%	8%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Corneil, D.G.. "Dominating sets in perfect graphs", Discrete Mathematics, 19901214 Publication	2%
2	Submitted to University of Reading Student Paper	2%
3	"Computational Science and Its Applications – ICCSA 2010", Springer Science and Business Media LLC, 2010 Publication	1%
4	"Applied Mathematics and Scientific Computing", Springer Nature, 2019 Publication	1%
5	Submitted to Indian Institute of Information Technology, Design and Manufacturing - Kancheepuram Student Paper	1%
6	Submitted to Birla Institute of Technology and Science Pilani Student Paper	1%

7	Submitted to University of Central Florida Student Paper	1%
8	Fedor V. Fomin. "On Two Techniques of Combining Branching and Treewidth", Algorithmica, 06/2009 Publication	1%
9	Submitted to Central Catholic High School Student Paper	1%
10	Submitted to UT, Dallas Student Paper	1%
11	Thomas Mensink, Thomas Jongstra, Pascal Mettes, Cees G.M. Snoek. "Music-Guided Video Summarization using Quadratic Assignments", Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval - ICMR '17, 2017 Publication	<1%
12	Xue Lei, Yi Cai, Qing Li, Haoran Xie, Ho-fung Leung, Fu Lee Wang. "Chapter 10 Combining Local and Global Features in Supervised Word Sense Disambiguation", Springer Science and Business Media LLC, 2017 Publication	<1%
13	Joseph S. Martinich. "A vertex-closing approach to thep-center problem", Naval Research Logistics, 04/1988	<1%

Publication

14	Lecture Notes in Computer Science, 2015. Publication	<1%
15	Fomin, F.V.. "On the domination search number", Discrete Applied Mathematics, 20030501 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		