International Journal of Computer, Information Systems and Knowledge Management

Vol.1, No.1, pp.8-16, March 2014

Published by British Research Institute UK (www.gbjournals.org)

KEYWORD EXTRACTION FROM E-BOOKS AND CONVERSION TO READER FRIENDLY FORMATS

Mrs. P. M. Chawan

Lekha Muraleedharan

Tushar Kaley

Akshat Sandh

Praveen Pamnani

Akshat Sandh

Praveen Pamnani

Department of Computer Science, V.J.T.I

ABSTRACT : *An overwhelming number of users use Ebooks as their primary formats. Gone are the days when buying a physical copy was the only thing to do. Even though technology has advanced to extreme extents in display and visualization of texts (display technology like elink, formats such as PDFs), our team is of the opinion that enough time and effort has not been put in into making data more audiovisual, and easier to understand and grasp. It is with his aim that we attacked this problem. Researching formats that are best suited for visualization and using NLP techniques to implement our end game, is the crux of this project.*

Keywords: PDF, Natural language processing, TextRank, Topic signature, KEA, Rhetorical parsing, STC, LINGO, Presentation, Mindmap

INTRODUCTION

With the advent of technology, we have too many distractions. It may be at work, at school or even when we talk. Therefore it becomes important to find a way to either concentrate or strip down work/data to its basic contents.

In today’s environment, with the explosion of data and availability of the web, it becomes necessary to change the static and dynamic components of data. It becomes important to realize what data is important, to filter out useless colors and animations (increase the static component), but at the same point make it more dynamic by allowing for instant web searches, etc. In all our research we found no software that fills in this gap, and therefore we have decided to attack this problem head-on and fill in this void.

We plan to build something that takes data inputs from different formats (eg. PDF, Text, etc) and converts it to simple text. It also must parse the data itself and give out what is important. To find important keywords and then, to use it to different ends is the end game.

Also, the data itself is not important if you can’t use it. This is where the usability part of our project comes in. We plan to use said keywords, to make auto generated presentations and formats, which maintain structure and hierarchy of the original document.

PROBLEM ELABORATION

In order to create an eBook assistant with the aim of helping students, teachers, readers and publishers absorb and disperse book content in a fast, effective and easy to grasp manner; we need to focus on two aspects. The first is keyword extraction and the second is display techniques.

Keyword Extraction

Keyword extraction is a process, which is used to understand text, and realize what is important, by machines. This is where the problem comes in. There is no standardization when it comes to writing; these are so many types of grammar implementation and so many words that understanding all of them may seem impossible.

But there are ways and means to implement this, these are:-

Supervised keyword extraction: Such algorithms have two phases; they first learn on a training dataset, and then they execute on an input dataset to identify keywords. Given the right training data, their output is far superior to other methods; however if the input data is quite different from the training data, the output suffers in quality.

Unsupervised keyword extraction: This class of algorithms uses heuristics and a one-size-fits-all approach to find keywords in all types of input text.

Display Techniques

After we extract keywords then what? We need to output it in such a way that is acceptable and useful to users. Therefore it makes sense to conduct a survey to understand what display techniques are required. The techniques that we considered are:

1) MindMap

2) Presentation

ALGORITHMS

Objectives

Our objective was to compare and contrast the available techniques for keyword extraction, summarization and clustering on any given body of English text to obtain keywords and key phrases.

We compared algorithms based on the following criteria:

Nature of algorithm: Algorithms may be supervised, unsupervised, rule based, statistical etc and each of these categories affect the nature of keywords generated and their relevance with respect to the context of the input text.

Generality: Less dependency on the subject matter of the input text, i.e. the algorithm must give equally good output on technical books, fictional novels, news articles, language books etc.

Performance: Time required for the algorithm, memory required, number of steps and load when implemented in Java or equivalent object oriented language, and how the performance varies as the thresholds and internal parameters are customized.

The algorithms we focused on were TextRank, Keyword Extraction Algorithm, Topic Signature Approach, Rhetorical Parsing Algorithm, Lingo, Maui, Mahout and STC. Their features are summarized below.

ALGORITHMS SUMMARY

Textrank

Overview

Textrank is a graph based keyword extraction algorithm, based on the concept of PageRank algorithm first used by Google for web pages and search indexing. Textrank converts words or phrases in the input text to nodes, and links two nodes if they are associated with each other, i.e. occur nearby each other in the input text. After constructing such a graph from input text, the algorithm performs many iterations in which the rank of each vertex is found. Rank of a node is high if the number of neighbors it has is high, and also if rank of those neighbors themselves is high. In this manner, ranks are updated in each iteration till convergence is reached. Nodes are then sorted in decreasing order of rank. The highest rank nodes are keywords.

Pros and Cons

Advantages of the TextRank for our application is that it is general purpose, topic independent and completely unsupervised and algorithm based thus needing no statistical reference data, training data predefined rule set to work against. It can also be efficiently implemented in code to work on input sizes ranging from a single paragraph to a whole book. Thresholds are customizable to determine the number and nature of keywords that form the output.

KEA

Overview

KEA is an algorithm for extracting keyphrases from text documents. It can be either used for free indexing or for indexing with a controlled vocabulary. Kea is implemented in Java and is platform independent.

KEA works on the input documents which are in ".txt" format. It makes use of thesaurus(if provided) also refers the stop words and makes sure the stop words do not appear in the keyword extraction output.

Before being able to extract keyphrases from new documents, KEA first needs to create a model that learns the extraction strategy from manually indexed documents. This means, for each document in the input directory there must be a file with the extension ".key" and the same name as the corresponding document.

Pros and Cons

While KEA was known to give very good quality keywords as output, the main drawback of KEA was that it is a supervised algorithm and hence it depends on the similarity between the subject matter of the input data and the previously seen training set. This makes it unsuitable to be used in a general purpose keyword extraction tool intended to work on books of all subjects.

Topic signature

Overview

Topic signature approach is a statistical keyword extraction and summarization technique. It follows the basic intuition that a word or phrase is important if its frequency in a given input passage is significantly different from its average frequency in the English language. Eg. Words such as so, and , but, thus are usually frequent in any given input. However, since there are anyways frequent words in the English language, they cannot be considered keywords. On the other hand, a word whose expected frequency is low but observed frequency in a given input is high is a good candidate for being a keyword. The algorithm calculates a statistic lambda for each word and selects the words whose lambda value is greater than 10 as keywords.

Pros and Cons

Topic signature is a purely statistical approach to keyword extraction and summarization. The output depends completely on the configuration of thresholds and statistical parameters. In general it was a good candidate for our application however we finally decided in favor of TextRank over Topic Signature approach due to its better execution complexity and generality.

Rhetorical parsing

Overview

The Rhetorical Parsing Algorithm was proposed by Daniel Marcu in 1997-1998. It is a bottom-up, data-driven, rule based algorithm for parsing a large body of text and understanding the underlying structure and meaning of the text. It makes of Mann and Thompson's theory of rhetorical relations, and the various nuances and rules within the detailed algorithm are based on large scale analysis of a large corpus of common English language text.

The basic principle of the algorithm is that each section, or paragraph or sentence of a piece of text contains a main, central idea known as nucleus; as well as the remaining supporting information known as satellite. There exists a rhetorical relation (eng evidence, concession, elaboration etc) between nucleus and satellite.

Using this principle the algorithm builds a binary tree out of the flat text. The binary tree is built bottom-up. As we move from lower to higher level in the tree we keep important sections of text known as salient units, and leave out remaining unimportant units. Salient unit at a leaf node consists of the leaf itself. For all internal nodes salient unit consists of union of salient nodes of all its nuclear children.

Thus level by level we analyze the text and identify important key-phrases. At highest level are top (say top 2 or top 5) key-phrases; at second level top 10 and so on.

How text is partitioned into individual units and how rhetorical relations are identified and how the rhetorical structure tree is iteratively built is defined by an exhaustive list of rules. These rules are based on an analysis of a large corpus of the English language.

Pros and Cons

The advantage of the Rhetorical Parsing Algorithm is that is it an iterative rule based algorithm and it is generic and not restricted to any domain of source text. In addition to that its algorithmic complexity is largely linear, making it an efficient algorithm. In addition the algorithm provides a deterministic way for building an un-ambiguous rhetorical structure tree. It is also proved to give superior results to naive summarization techniques and many commercial mainstream summary systems.

The disadvantage of the algorithm is that its parsing rules are based on structure of the English language, hence while it may work well on some books, it may not give optimal results on technical content or books with unconventional use of language. Space complexity is high since binary tree is built for the entire body of text. Thus the algorithm was rejected for our purpose.

Lingo

Overview

LINGO is a web clustering algorithm. This algorithm first describes the meaning of the clusters and then depending on the results decides the content .

Pros and Cons

Since LINGO is more of a clustering algorithm it is more suited for web mining and data analytics purposes rather than single document keyphrase extraction. The algorithm suffered in performance characteristics for our purpose.

MAUI

Overview

Maui is a multi-purpose algorithm. It performs not only keyword extraction but algorithm topic indexing and tagging. Maui uses both supervised approach using a language model built from some training data, as well as statistical measures such as TFxIDF, along with some heuristics such as position of word in text(start of paragraph etc) to ascertain whether a given word or phrase is a keyword. Maui can also perform extraction with restrictions imposed such as a specifying the subject or domain of input and searching for keywords relevant to the domain.

Pros and Cons

Maui is a suitable algorithm when the requirement is of a small number of keyphrases or tags. Had we decided to breakdown our input into chunks the size of a single paragraph, Maui would have been the algorithm of choice. However for true coherence a slightly larger input size and correspondingly a larger set of output keywords seemed appropriate hence Maui was rejected.

Mahout

Overview

Mahout is a clustering algorithm, based on concepts originating from data mining. It implements partitioning clustering and hierarchical clustering. Mahout groups words together based on their similarity and characteristics and uses the obtained clusters to determine importance of words. The cluster in effect serves as a model which is then used for categorizing words as keywords. Mahout has a supervised element in the sense that it forms clusters and classifies words as keywords based on the knowledge it has gained from prior inputs. Mahout is also useful for multi document clustering i.e. taking a group of documents as input, finding keywords and main topics in each document and classifying the documents in order of importance or in order of relevance with a given topic.

STC

Overview

STC is a bottom up, agglomerative hierarchical clustering algorithm. It builds a suffix tree, starting from the leaf nodes and progressively moves upwards. The leaf nodes are the words or phrases from one or more input documents. Similar nodes are combined based on their association and interrelation with each other. In this manner, smaller clusters are iteratively combined to form bigger and bigger clusters. Once this is done, the clusters are classified based on their relative importance and from this a list of keywords or keyphrases is obtained.

Pros and Cons

STC similar to Maui performed better when the requirement was of small set of output keywords. Also the performance of clustering algorithms in terms of time and space complexity was prohibitive.

Comparison of keyword extraction algorithms

Name	**Nature of Algorithm**	**Generality**	**Performance**
TextRank	Unsupervised, graph based ranking	Excellent	Excellent
KEA	Supervised, statistical	Fair	Good
Topic Signature	Statistical	Good	Good
Rhetorical Parsing	Rule Based	Good	Excellent
Lingo	Clustering	Good	Fair
Maui	Supervised, statistical.	Fair	Excellent
Mahout	Clustering	Good	Fair
STC	Clustering	Good	Fair

METHODOLOGY

- PDF to Plaintext

A .pdf file is taken as input for the application. Text is extracted from the e-book and is split into chapter-wise files which form input to the textrank algorithm. The extraction is done in such a way that the hierarchy of chapters and subtopics is maintained and separate text files are generated for each leaf in this hierarchical tree.

- Keyword Extraction

The textrank algorithm accepts a plaintext file as input. The textrank algorithm is run on the file and a list of keywords obtained is returned as output.

The algorithm performs the following steps to convert input to output:

Prepare graph: A graph is created from the words and phrases in the text file. A node in the graph is a word or a group of words. Two nodes are connected by an edge if the nodes are associated. An association means the two words occur in close vicinity.

The program reads the entire input text file into a string. The input is then segmented into paragraphs and then into sentences. The sentences are tokenized and a graph is built with the tokens as nodes. Links in graphs are added based on associations between nodes.

Calculate normalized weights: Now the algorithm calculates weights for each node. A node's weight depends on the number of other nodes associated with it as well as the weights of the neighbors themselves. Each node begins with a fixed value as weight, which is continuously revised as the algorithm progresses.

The graph traversal occurs for N iterations(where N is size of graph) or until standard error falls below a threshold. At each iteration the rank of a node is revised based on the given formula which depends on number of neighbors and rank of neighbor.

List keywords: Once the algorithm converges, nodes are serialized and arranged in decreasing order of weight. Keywords are those nodes with a high value of normalized weight. A parameter max_results determines how many nodes are marked as keywords.

Transform keywords: In order to facilitate creation of mindmap and presentation, the simple list of keywords is transformed into xml elements or a list respectively, before being written to the appropriate files. The toString() function in TextRank class handles this.

- Keywords to Mindmap

An output mindmap xml file is created line by line. The wrapper code inserts tags for internal nodes while the textrank code itself adds the leaf nodes to the output file.

Freemind stores its mindmaps in a simple xml format. The root node in this xml tree is <map> and under it are internal and leaf nodes with the tag <node>. The text property of the node elements describes the content of each node of the mindmap. Additional attributes may be added to the node element to describe its positioning, format, and revision history. However our primary code outputs simply the node contents and uses default formatting and layouts.

- Keywords to Presentation

Presentation is created using the leaf nodes obtained from textrank.

CONCLUSION

Ebooks have now become one of the major forms of consumption of text. With great strides being taken in display technology and formats, the actual readability was left behind. Well, now this paralysis analysis stage is now over.

In the alpha and phased beta testing of our project, we have found that it is extremely well received. Also, when the documentation was shown to our colleagues, the interest that they have shown in developing for this open source environment has been a reassuring factor, a sort of validation for the effort that we have put in and as to the correctness of the direction we have chosen to take this project in. We are happy to say that one of our colleges has already started building a plug-in that will do the same for ePubs, hopefully in a couple of months we will integrate this with our application.

REFERENCES

Online course on Natural Language Processing by Christopher Manning and Dan Jurafsky on Coursera https://www.coursera.org/course/nlp

Speech and Language Processing (2nd Edition) by Daniel Jurafsky , James H. Martin

Foundations of Statistical Natural Language Processing by Christopher D. Manning and Hinrich Schiitze

Steve Branson, Ari Greenberg Clustering Web Search Results Using Suffix Tree Methods.

Fei Liu, Feifan Liu, Yang Liu Department of Computer Science The University of Texas at Dallas. AUTOMATIC KEYWORD EXTRACTION FOR THE MEETING CORPUS USING SUPERVISED APPROACH AND BIGRAM EXPANSION (978-1-4244-3472-5/08 ©2008 IEEE)

Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin and Craig G. Nevill-Manning .KEA: Practical Automatic Keyphrase Extraction.

Eibe Frank and Gordon .W.Paynter and Ian .H. Witten, Carl Gutwin, Craig .G . Nevill-Manning. Domain- Specific Keyphrase Extraction

Rada Mihalcea Department of Computer Science,University of North Texas Graph-based Ranking Algorithms for Sentence Extraction Applied to Text Summarization

Rada Mihalcea and Paul Tarau, Department of Computer Science, University of North Texas. TextRank: Bringing Order into Texts

Rada Mihalcea and Paul Tarau, Department of Computer Science and Engineering, University of North Texas. A Language Independent Algorithm for Single and Multiple Document Summarization

Elena Lloret , Dept. Lenguajes y Sistemas Inform_aticos,Universidad de AlicanteAlicante, Spain. TEXT SUMMARIZATION : AN OVERVIEW (TIN2006-15265-C06-01)

Steve Jones, Stephen Lundy, Gordon W. Paynter, Department of Computer Science, University of Waikato,Private Bag 310.5, Hamilton, New Zealand. Interactive Document Summarisation Using Automatically Extracted Keyphrases (0-7695-1435-9/02(c) 2002 IEEE)

TextRank algorithm implementation in java. https://github.com/turian/textrank

Apache PDFBox. http://pdfbox.apache.org/

FreeMind. http://freemind.sourceforge.net/wiki/index.php/Main_Page

Apache POI XSLF. http://poi.apache.org/ http://poi.apache.org/slideshow/index.html

Apache commons.lang http://commons.apache.org/proper/commons-lang/

Eclipse Indigo http://www.eclipse.org/

Git. http://git-scm.com/

Github. https://github.com/

commons-logging-1.1.1.jar http://commons.apache.org/downloads/download_logging.cgi

commons-math-1.2.jar http://commons.apache.org/downloads/download_math.cgi

log4j-1.2.15.jar http://logging.apache.org/log4j/1.2/download.html

porterstemmer.jar http://snowball.tartarus.org/download.php

opennlp-tools-1.3.0.jar http://opennlp.sourceforge.net/

maxent-2.4.0.jar https://sourceforge.net/projects/maxent/

sptoolkit.jar http://text0.mib.man.ac.uk:8080/scottpiao/sent_detector

trove-2.0.2.jar http://trove4j.sourceforge.net/

jwnl-1.4rc1.jar http://sourceforge.net/projects/jwordnet

jdom-1-1.jar http://jdom.org/downloads/index.html

ObjectAid UML Explorer for Eclipse http://www.objectaid.com/home