Context dependent keyword extraction for short text matching

M.Tech Project Report

Phase I

Submitted in partial fulfillment of the requirements
for the degree of

Master of Technology

by

Jayaprakash S

Roll No: 123050045

under the guidance of

Prof. Dr. Pushpak Battacharya

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai

Acknowledgement

I would like to thank my guide, Prof. Dr. Pushpak Battacharya for the consistent support and guidance he provided throughout the semester. I also would like to thank the team Mr. Arjun and Mr. Swapnil for valuable guidance and discussions.

TODO:Write more

Abstract

Information is represented in several formats like text, images, videos, etc and each of them convey information to user in different rate. We see world wide web is part of peoples day to day life. News in word wide web occupies large part of online people.

In this work, we are trying to retrieve an Image that is relevant to the given news article.

Contents

Chapter 1

Introduction

1.1 Motivation

1.2 Applications

1.2.1 Recommendations

1.2.2 Summary

1.2.3 Ad-relevance

1.2.4 Document-similarity

1.3 Challenges

1.4 Roadmap

Chapter 2

A Closer look at the problem

2.1 Classification of Images

2.1.1 Representative Images

2.1.2 Iconic Images

2.1.3 Aspectual Images

2.1.4 Cited Images

2.1.5 Historical Images

Chapter 3

Literature Study

3.1 Unsupervised Keyword Extraction Methods

Unsupervised keyword extraction methods mostly rely on the relationship between the words in the text. Importance of the word is estimated based on exploiting the relationship to all other words.

3.1.1 TextRank

TextRank is a graph based unsupervised ranking algorithm formulated for text processing via extracting keywords and sentence extraction from documents. This method is based on Graph-based ranking algorithms like HITS algorithm reference , Google Search Engines PageRank reference where is used for social networks, citation analysis and analysis of link structure world wide web. These graph based algorithms exploit the global information computed iteratively rather than looking at only surrounding information.

The basic idea of the graph based ranking models is voting or recommendation. If one vertice connected to another vertex, basically it is casting a vote for that other vertex. The higher number of votes the one vertex gets, the same amount of importance it gets. The importance of the vertex which is casting vote is going to determine the amount of importance should be given to that vote. Effectively the score associated with the vertex is determined by votes that are cast for it and scores of vertices that are casting vote for it.

Formally, considering G = (V, E) as directed graph with set of vertices V and set of edges E, where E is the subset of VxV. Let In(V_i) be the set of vertices that are voting V_i

(predecessors) and Out(V_i) be the set of vertices that V_i is voting to. The score of vertex V_i is calculated as follows, reference

$$S(V_i) = (1 - d) + d \sum_{j \in InV_i} \frac{1}{|Out(V_j)|} * S(V_j) \tag{3.1}$$

here d is called as damping factor that can be set between 0 and 1 initially, which has the role of adding a probability directly jumping from one vertex to another vertex (actually it signifies the default or implicit voting given by any vertex to all other vertices to avoid dead-end in random walk).

Starting from arbitrarily assigned values to all vertices in the graph, the vertex values are computed iterated until convergence or given threshold is reached. After convergence is reached, the scores of the vertices represents the importance of the vertices within the graph.

Document as graph

To use the graph ranking algorithm for natural language text, first we should build a graph that represent the given text and interconnected words and relations between them. Based on the application, text units of various sizes can be used as vertices (examples words, sentences, phrases etc). similarly based on the application we can decide type of relations should exist between vertices e.g. lexical or semantic relations, contextual overlap etc.

Overall steps of this algorithms is,

1. Identify the text units of document and add them vertices to the graph.
2. Figure out the relations between text units, that best suits the application. Edges that connects the vertices can be undirected or directed, Weighted or Unweighted.
3. Assign the initial scores of vertices arbitrarily and iterate through ranking algorithm until convergence.
4. Sort the vertices based on final score. and selects top-K vertices as candidate vertices or text units.
5. [Optional]Post processing is applied to vertices or textual units.

The expected result of keyword extraction task is set of keywords or phrases for a given natural language text. Any relation between two lexical units can be used as connection between two vertices. Here in this paper, co-occurrence relation with the controlled distance is used as edges or connection between vertices. Two vertices are said to be connected if that two lexical units tend to co-occur within a window of N words, where N can vary.

The vertices added to graph contains lexical units of certain types, for instance in this paper reference they have used the individual words as the vertices and connection between vertices (individual words) represents that they co-occurred in the text within the window size N.

Undirected Edges

Graph used for ranking keyword is undirected, whereas original algorithm was developed for directed graphs. If the two words tend to co-occur then they are mutually connected to each other, so each vetices in-degree equal to out-degree.

Weighted Edges

Edges in the TextRank model is weighted, they directly indicate the strength of connection between two vertices. In this if we two words tend to co-occur frequently then they will have strong (more weight) connection.

By considering above undirected and Weighted cases, the original graph based ranking algorithm has been modified into as follows,

$$WS(V_i) = (1-d) + d * \sum_{j \in InV_i} \frac{W_{ji}}{\sum_{k \in OutV_j W_{jk}}} * WS(V_j) \tag{3.2}$$

TODO: Diagram - Text

TODO: Diagram - Text graph

Process : First the given document was tokenized, and syntatic classes of each word (part of speech tag) is identified. It is said that picking only certain syntatic classes gives the better precision (nouns and adjectives). Only the unigrams considered as vertices. Graph ranking algorithm run on the constructed graph. Top fraction of vertices selected based on score given to vertices on convergence. If the selected unigram words tend to cooccur in the text they are combined together and considered as multi-word keywords or keyphrases.

Results : This algorithm was tested against 500 science articles where keywords algorithm was compared with manually annotated keywords. It is shown that this algorithm achieves highest F-score 36.2% when edges are considered as undirected with co-occurence window size (N) is 2.

3.1.2 RAKE

3.2 Supervised Keyword Extraction Methods

3.2.1 KEA

Features

Process

Classification

Results

3.2.2 Naive Bayes

Chapter 4

Experiments

4.1 Unsupervised Approaches

4.1.1 Boosting based on occurrence and co-reference

Simple method for the determining top keywords will be counting the occurrences of each word in the given document. However the problem with this method is Natural language text the word is represented in different forms. Stemming may help normalizing different verb representation with morphemes. But Images and news articles mostly centered around the entities and then verb as relations if required. When we want to find the fraction of sentences that does covers entities or nouns we can use the co-reference resolution and anaphora resoultion.

Steps of the experiments are,

1. Given document is tokenized in to lexical units.
2. All possible noun phrases from the parse tree are considered as candidate for final key phrases.
3. Stanford co-referencing run on the sentences of articles. From the output of co-reference, the frequency of each noun phrase is calculated.
4. Apart from using the frequency of noun phrases in the articles

4.1.2 TextRank - Modified

4.2 Supervised Approaches

4.2.1 Naive Bayes

Chapter 5

Conclusion

In this work, we have studied various ways of mapping between large text and short text through extracting keyword from large text.

In future, we are planning to using the external resources to better understand the relation between the large text and short text. For example, using the click-through logs will be helpful.

Chapter 6

Future work

In this work, we have studied various ways of mapping between large text and short text through extracting keyword from large text.

In future, we are planning to using the external resources to better understand the relation between the large text and short text. For example, using the *click-through logs* will be helpful.

6.1 Short term Plan

1. Complete current experiments
2. Analysis of outcome
3. Improving within current framework and available resources

TODO: explain above points

6.2 Long term Plan

1. trying to utilize the external resources apart from Article + meta data
2. utilizing the click through logs
3. utilizing the semantic class from Freebase, or others

TODO:elaborate above with examples what it will solve

6.2.1 Click-through logs