University POLITEHNICA of Bucharest

Faculty of Automatic Control and Computers,
Computer Science and Engineering Department

BACHELOR THESIS

Identification of Keywords in Historical Events

Scientific Adviser:
S.L. dr. ing. Costin-Gabriel Chiru

Author:
Ioana Andronescu

Bucharest, 2014

Abstract

Many historical events are mentioned in books of different categories. Some words are even specific to a certain period linked to an important moment. The aim of this paper is to establish a link between these events and the text written is that period. Google Books Ngram Corpus was chosen for data analysis, as it contains n-grams from parsed texts over the time. In order for the event analysis to work properly, we need some words to find in the n-gram data set to see when they were mostly used in time. The words we are looking for in this correlation are extracted from Wikipedia articles linked to specific events.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Challenges and Motivation

Historical events along with their causes have been analyzed by the historians for ages and are an important tool in the field of social sciences and not only. By "event" we refer to "occurrence in the lives of individuals". A historical event is defined by one or more occurrences connected together. They can be found in papers, magazines, history books and last, but not least, in large texts written in the past. In the latter case, the events have a greater or lesser delay, as they haven't been analysed yet.

Not too long ago, identification of historical circumstances was hard to be done, as historians had to examine book by book, page by page, and no wonder that the process took a lot of time. Luckily, technology has evolved and the books have been digitized, so that the problem has been reduced to the automatic identification of various events in texts, but in a much shorter time.

As specified in [1], time for historical events is measured in very large discrete units like years or decades. Seeking for a specific word in different texts over the time, we can see that in some years it was frequently used, while in others it was barely there. These years, along with their frequencies for the given term lead us to time series which are necessary for data analysis and, implicitly, in detecting an important circumstance. In social studies the duration of an event is also relevant for the analysts to determine or predict different trends or problems.

A historical event can actually be seen as a transition from one state to another. According to [2], the before state is referred as "in a dispute" and the after state is the "out of a dispute". For example, a war is caused by some known issues and, on its turn, it also causes some changes. This observation could have a great meaning in determining the factors that caused such tragedies like war, famine, flood etc. in the past and maybe prevent them in the future.

A specific event can be certainly identified as unique by a couple of keywords, we may say. These words might be proper nouns that define, for example, a nation, region, person name, river, country or a state. We don't expect to find common nouns as being relevant, but we expect to see words that were used more frequently in that period of time than in others. Thus, identifying keywords is an important step in historical event detection.

Analyzing history data is useful in linguistic, demography, medicine, technical reliability, biology, econometrics, sociology etc. and can help understand causality for different problems for these domains.

1.2 Objectives

The aim of this paper is to create a model for automatic identification of keywords from historical events on Wikipedia. The words are searched in the Google Books Ngram Corpus which is actually a very large data collection extracted from millions of digitized books. A "bag of words" was chosen for the representation of a Wikipedia article. For a single term, we can easily find the frequency time series by making queries to the Google Books Ngram Corpus.

Assuming that we already have the parsed articles, the first step in identifying the keywords for an event is to compute the historical relevance for each word from the specified text using some algorithms to detect the peaks. The historical relevance is the scaling of the original results in a range from 0 to 10, 0 being the less relevant and 10 the most relevant.

In a second phase, we extract only the terms with a positive relevance in the period reflected by the event. We will see that most algorithms give slightly different results and we would like to see the best results, though. In order to obtain more uniform results, in the next step we extract the common keywords from each method. We would also like to cut out the relevant words that appear in more than one text because a historical event should be represented by mostly unique terms.

We would also like to know which of the methods performed the best. A solution for this is to measure the difference between the probability distribution of the original time series (with frequencies scaled from 0 to 10) for one word and the resulted time series after computing one of the algorithms. We know that the closer the result is to 0, the more the method worked. But a text contains more than one word, so we need to compute the average divergence considering each one of them.

Overall, the objectives of this research consist in finding not only the keywords reflected in a historical event, but also a good method for peak detection.

1.3 Outline

Further, the thesis is structured in chapters as following:

Chapter 2 provides the reader some background on Google Books Ngram Corpus that is very important for our data analysis. It is important to know what they are and how they are structured, where to find them and their usage. When testing the data, we process a couple of articles from Wikipedia and extract the words, cutting off the stop words and carefully keeping only the nouns and verbs. The latter part can be done with the help of a Part-of-Speech Tagger, or shorter, POS Tagger which will be explained here. The nouns will be lemmatized so that they will be reduced to a somewhat more general form. The end of this chapter includes some of the basics about the English Wikipedia offline version.

In Chapter 3 we discover techniques such as time series smoothing and peak detection and their usage. We will dig deeper into discovering peak detection algorithms: Level Difference that includes two methods, three of the Sliding Window Methods and the Double Change Method.

The details of implementation are described in Chapter 4. For one to understand how the identification of keywords from historical events works, it is needed to explain how the original data is parsed and then stored and what is expected as a result after applying the algorithms mentioned in Chapter 3. Searching a word in the Google Books Ngram Corpus or a processed article from Wikipedia needs to perform really fast and, for this reason, all data needs to be indexed.

In Chapter 5 we will see some of the evaluation methods that include plotting the data and the Kullback-Leibler divergence algorithm. used to measure the difference between the probability of the true distribution of data (original time series) and the model (modified time series after running the peak detection method).

It is important to observe that the peak detection methods actually work and we will see this in Chapter 6. We need to know which method works better or worse and this can be done using the Kullback-Leibler divergence algorithm. After this, we present some of the relevant words and see which of the algorithms from Chapter 3 works best on these.

We conclude Chapter 7 with a summary of the presented research, observations upon the performance of the algorithms presented in previous chapters. We also focus on the applicability of this work and on how this might be extended in the future.

Chapter 2

Literature Review

2.1 Google Books Ngram Corpus

Digitalization has had a great impact over the last years. More than 15 million of books from 40 university libraries around the world were scanned page by page and digitized using optical character recognition, or, shortly, OCR. Of all these, Google selected a number of 5 million of high quality digitized books written between 1500 and 2008, which is almost 4% of all books ever published. In [3] it is specified that the result was a corpus of 500 billion words in main languages like English, French, Spanish, German, Chinese, Russian and Hebrew. Instead of providing raw text, Google Books Ngram Corpus actually consists of sequences of up to 5 terms referred to as n-grams. A 1-gram might be not only a word, but also a number or a typo. Along with it, is also kept information such as the year and the counts, pages and volumes from that year. For all words from the corpus we can also find triplets of values (match count, page count, volume count) per each year.

Estimations say that there are about 361 billion words in the English n-gram corpus, according to [3]. The size of the language has increased by over 70% during the last fifty years, with almost $8,500$ words per year. A 1-gram is considered to be common if it has a frequency of over 1 per billion. The word's frequency for a year can be computed as the match count divided by the total number of words in the corpus for that year. For example, let's take the term "war". In 1916, it appeared $435,961$ times from a total of $1,175,413,415$ terms from 1916. Thus, the computed frequency is around $3,7$ x 10^{-4}.

Google has created the Google Books Ngram Viewer which is a web application for visualising time series of one or more n-grams selected by the user. These series refer to a sequence of successive points representing the years and their corresponding frequencies. Looking at the time series for "war" in Figure 2.1, we observe that the peaks seem to be more dense between 1550 and 1800. We can clearly see two peaks around 1910-1920 and 1930-1950, corresponding to World War I and to World War II, respectively.

The time series may also need to be smoothed, which means that the data for one year will be an average of the raw count for that year plus window size (smoothing value) on either side, left or right. For example, having 1990 with a window of 1, the smoothed value will be: (count for 1989 + count for 1990 + count for 1991) divided by 3. A window size of 0 shows that we will have only raw data, so just the year we're referring to.

The analysis of the corpus leads to investigation of cultural trends and culturomics, focusing on fields such as lexicography, grammar evolution and history, reflected in the English language between 1800 and 2000, as explained in [3]. The most important applicability of Google Books

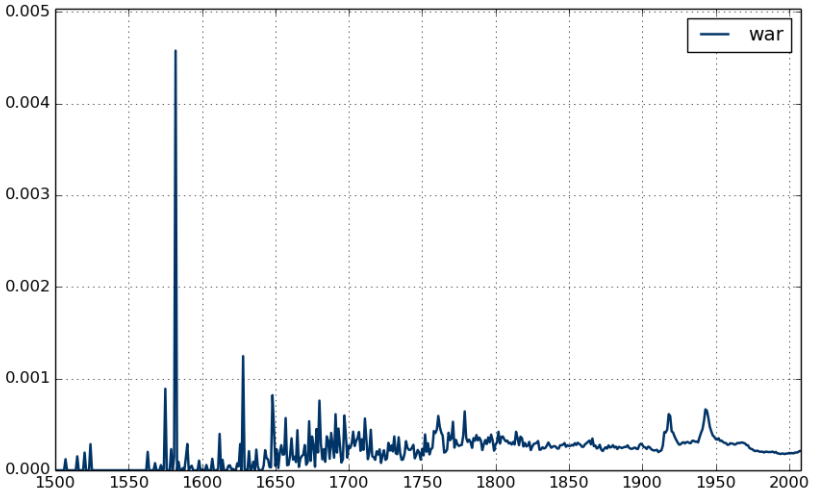

Figure 2.1: Ngram time series for "war"

Ngrams Corpus lies in the social sciences, like history, where one can detect important events in history using specialised algorithms for bursty features or peaks.

2.2 POS Tagging

Part-of-speech tagging (POS tagging), also called word-category disambiguisation, is the process of assigning a part of speech to a word from a text or a corpus. In natural languages, most word-forms are ambiguous, so looking at the definition of the term is not enough. For example, "war" could either be a noun or an adverb. This is why the context of the word should also be taken into account.

In school we've learned that there are 9 parts of speech (noun, verb, pronoun, adverb, adjective, article, preposition, conjunction and interjection), but there are actually much more POS tags. For nouns, singular forms are distinguished from the plural ones. They are also differentiated by the grammar cases such as subjective, possessive, objective or vocative.

In early times, Hidden Markov Models (HMM) were used for part of speech disambiguation by creating tables with the probabilities of occurrence of certain sequences, as it is explained in [4]. It could learn the probabilities of a verb followed by a noun, or an adverb and chose the best possibility, i.e. the case with the highest probability.

Although the HMMs had an accuracy of 90-95%, it was very time consuming, as it took all possibilities into account. Later on, Steven DeRose [5] came up with a dynamic programming solution similar to the Viterbi algorithm - VOLSUNGA - that could solve the same problem, but in linear time and with an accuracy of 96%. This algorithm uses a matrix of collocational probabilities corresponding to the co-occurence of all pairs of tags. The ambiguous words from a span in the Brown Corpus represent a set of mappings from words to individual tags. Each tag has associated a probability that can be found in the collocational matrix. The optimal path is the one with the highest probability that actually returns the desired result. The solution failed for small cases such as the ones where semantics was required. However, this approach encouraged researchers to dig deeper in into the field of part-of-speech tagging.

Figure 2.2: Ngram time series for "conquer" and "conquers"

In terms of machine learning, part-of-speech tagging can be considered a supervised learning problem. A very fast and accurate solution is the one proposed by Honnibal in [6] based on an averaged perceptron. Given a word at the column i from the table data, the part of speech tag can be predicted using the predictor columns (features) i-1 and i+1, respectively. Each feature has associated a weight that accumulates a value if the guess is right and receives a penalization of a point if it is wrong. In order for the results to be more correct without losing the value of the first predictions (iterations), an average for the weights is computed. The accuracy of this algorithm reaches up to 96% in a much shorter time than the Python's NLTK library.

2.3 Lemmatization

In linguistics, lemmatization refers to the process that groups forms of a word into a single one, which is the lemma, or base form. For example, the lemma of the words "works", "worked", "working" is "work". This is often combined with POS tags, the result being the lexeme. Lemmatization is close to stemming, though the latter one neglects the context of the word. Such being the case, it is much easier to implement and much faster, although, it has a much smaller accuracy than the first one.

As can be observed in Figure 2.2, peaks are similar for both time series for the words "conquer" and "conquers", according to [16]. Lemmatizing them would give us the term "conquer" which wouldn't affect our research. Herein, WordNetLemmatizer library from NLTK is used to remove the inflexional endings from the English words from Wikipedia articles and keep the lemmas.

2.4 English Wikipedia

Wikipedia, the well-known Internet encyclopedia has about 30 million articles in 287 languages, as specified in [7]. This community-based project offers free copies in different languages for those who may use it for various purposes. The English one remains, however, the most comprehensive of all copies.

According to [8], the latest version of the dump from 13th February 2014 is actually a very big XML file - 9.85GB of compressed data and about 48GB of uncompressed data. The file can't be opened manually due to the operating system limitations of 2GB or 4GB, but specific programs exist, e.g. "less" command for Linux can do such a work.

Lowest estimate	Highest estimate	Event	Location	From	To	Notes
15,000,000[84]	55,000,000[85]	Great Chinese Famine	People's Republic of China	1958	1962	During the Great Leap Forward under Mao Zedong tens of millions of Chinese starved to death[86] and about the same number of births were lost or postponed.[87] State violence during this period further exacerbated the death toll, and some 2.5 million people were beaten or tortured to death in connection with Great Leap policies.[88]
5,000,000[89]	10,000,000[89]	Russian famine of 1921	Soviet Russia	1921	1922	See also: Droughts and famines in Russia and the Soviet Union and Russian Civil War with its policy of War communism, especially prodrazvyorstka
4,000,000	4,000,000	Bengal famine of 1943	British India	1943	1943	The Japanese conquest of Burma cut off India's main supply of rice imports[90] However, administrative policies in British India ultimately helped cause the massive death toll.[91]
2,400,000[92]	2,400,000	Japanese occupation of the Dutch East Indies	Indonesia	1944	1945	An estimated 2.4 million Indonesians starved to death during the Japanese occupation of Indonesia. The problem was partly caused by failures of the main 1944–45 rice crop, but mainly by the compulsory rice purchasing system that the Japanese authorities put in place to secure rice for distribution to the armed forces and urban population.[92]
800,000[93]	950,000[94]	Cambodian Genocide	Cambodia	1975	1979	An estimated 2 million Cambodians lost their lives to murder, forced labor and famine from the Cambodian Communist government, of which nearly half was caused by forced starvation. Came to an end due to invasion by Vietnam in 1979.

Figure 2.3: Wikipedia table of events (famine)

Data can be parsed using specific libraries for different programming languages. It cannot be parsed like any other XML because it contains specific tags and delimiters. SQL queries can also be performed to retrieve information from Wikipedia. Another option is to get data directly from wikipedia.org at runtime using a web crawler [8]. However, this method is not recommended at all when downloading a very large number of articles because not only that it takes a lot of time, but it also slows-downs Wikipedia in a dramatic way. There still is another method to make queries on the database dump in a distributed manner, using the Hadoop MapReduce framework.

In this research, the extracted texts are the ones referring to historical events [9] that can be found in a specific table like the one in Figure 2.3. The needed columns are the event and the period of time.

Chapter 3

Historical Relevance

3.1 Time Series Smoothing

As mentioned in the previous chapter, in our research we use 1-grams from the Google Books Ngram Corpus. The analysis is based on the time series between the years 1500 and 2008 for each given word from the Wikipedia article.

Although it was put much effort into ensuring the quality of the corpus, errors are inevitable. In order to reduce the negative effect of the errors, we need to find methods to reduce the noise. Intuition tells us that we could smooth out the noise in such way to modify the points and reduce them.

Having the time series $\{s_{i,t}\}_{t=1500}^{2008}$ for a given word w_i, we can find a function to compute the smoothed series. A suggestion [10] is to compute an average of the points within the window to the left and right sides of the current point. This is often called the Moving Averages Method, as it iterates though the points and moves the windows along with them. Knowing that the window size is N, the new value of the current point in the series can be mathematically written as follows:

$$u_k\left(i,t\right) = \frac{\sum_{k=-N}^{N} s_{i,t+k}}{2 \cdot N + 1}, \forall t \in \overline{1500 + k, 2008 - k}. \tag{3.1}$$

The time interval contains only 508 values starting with 1500 + N and ending with 2008 - N. In the implementation, the time series are stored as arrays and we know that in many programming languages they usually start from 0. Thus we will consider the interval [N, 508-N] instead of [1500 + N, 2008 - N], as it is somehow more easily to write and follow.

What Algorithm 1 does is compute an average of the years from the left and right sides of the current year. After that, it moves to the next point and updates the new window for both sides. Sometimes, the last window doesn't exactly end with the last element in the time series. In this case, the values in the interval [508 - N, 508] need to be exactly the original ones so that the smoothing series don't look unnatural.

In Figure 3.1 can be observed the time series for "war" with a smoothing of 2. As we will see later on in this chapter and in the results, some methods for peak detection work better with smoothed data. However, if the window is too big, important values in the time series will be lost and the peak detection algorithms won't find some of the real peaks. This is why we will test the data with a smoothing window of size 3.

Algorithm 1 Smoothing Time Series

Require: $\{s_{i,t}\}_{t=0}^{508}$
Ensure: $\{u_{i,t}\}_{t=0}^{508}$
for $t = N \rightarrow 508 - N$ **do**
    $u_{i,t} \leftarrow 0$
end for
for $t = N \rightarrow 508 - N$ **do**
    for $t = -N \rightarrow N$ **do**
        $u_{i,t} \leftarrow \frac{s_{i,t+k}}{(2 \cdot N+1)}$
    end for
end for
for $t = 508 - N \rightarrow 508$ **do**
    $u_{i,t} \leftarrow s_{i,t}$
end for

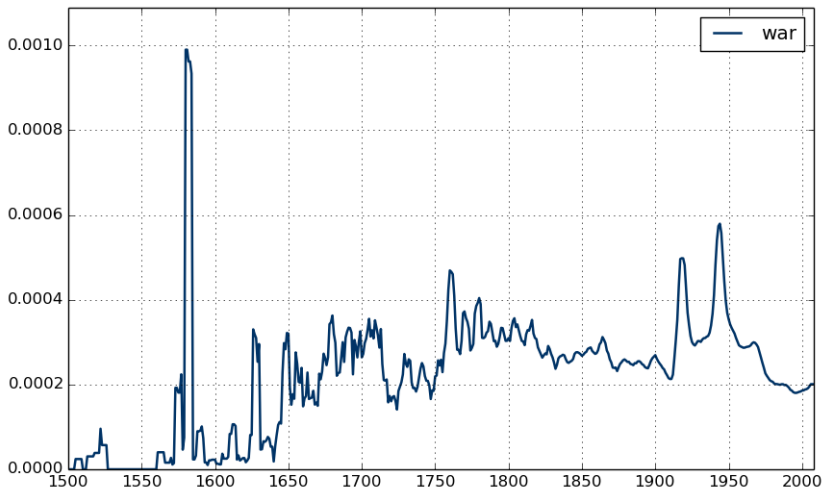

Figure 3.1: Ngram time series for "war" with smoothing window of 2

3.2 Historical Relevance Algorithms

A text can be defined uniquely by a few keywords that don't appear in exactly the same order or don't appear at all in other texts. These are the relevant words. In our research we will use Wikipedia articles specifying different historical events like World War I, American Civil War, Vietnam War etc. and seek for the words that define them.

The idea is to find the peaks of the terms that appear in each article and attach them a score from 0 (least relevant) to 10 (most relevant). Firstly, we search into the Google Ngram Corpus to get the frequency for each year from 1500 to 2008 and extract the time series for the words in the articles. After this, an algorithm for peak detection is used to get the peaks and each value is scaled to 10.

Peak detection is used in time series analysis and signal processing, where it has quite a frequent usage. Palashikar [11] talks of a few general approaches for peak detection. The first one includes smoothing the series and then applying a function on the result. A second method is to detect zero-crossing for the current point and its neighbours. This one has to find the local maximum

and then compute the difference between the points.

In 2006, Xin Zhang proposed an algorithm for burst detection to measure an unexpectedly large number of events that occur in a certain time. His approach would apply in several domains like financial, astrophysics or geology. A different data structure, Shifted Aggregation Trees (SAT) was used to store the time series so that it could perform a fast search. His method can be very well applied for gamma rays to detect the occurrence of a supernova, or to anticipate volcanic eruptions. Burst detection is different from peak detection because it doesn't find the isolated peaks and this is a big minus in our case. In the identification of keywords from the historical events it is better to use the first type of algorithms because all kinds of circumstances are important. For example, World War I and World War II are isolated events and they are the most important wars from the 20th century.

In our research we will analyze three types of methods. The first one, identified in [12] supposes that a peak can be identified by an abrupt increase followed by an abrupt decrease of the time series. A second type is the one that uses a sliding window, proposed by Palashikar [11] and includes other three methods. We will end this chapter with a new method that has two functions based on the level difference between the maximum (peak) and the initial minimum (valley before the peak) and final minimum (valley after the same peak).

Given the selected word wi, let $\{s_{i,t}\}_{t=1500}^{2008}$ be the time series that contain 508 values. We discussed in a previous section that it is better to keep the [0, 508] interval, so the new time series used in later on approaches will be $\{s_{i,t}\}_{t=0}^{508}$.

We further describe the proposed algorithms.

3.2.1 Double Change Method for Peak Detection

A simple method to detect the peaks is to check if there is a double increase or a double decrease of the time series $\{s_{i,t}\}_{t=0}^{508}$ in the same direction.

A more formal way to write it is like this:

$$m \cdot s_{i,t-1} < s_{i,t} \text{ and } m \cdot s_{i,t} > s_{i,t+1}. \tag{3.2}$$

The m parameter is written differently for each of the two cases. For the increase, m is 1 + x and for the decrease, it is 1 - x, where x represents the relevance, that is a number between 0 and 9. Thus, the final historical relevance is a natural number from 0 to 10.

Algorithm 2 presents the method in a more detailed form. As we can see, the relevance is computed directly according to "how much the time series grow" and "how much the time series fall". If there is a sequence of three elements with values of 0, the current element gets a 0 relevance.

The disadvantage of this method is that it might miss some of the peaks if the "how much it increases" (+x) and "how much it decreases" (-x) values are not chosen correctly. Choosing a value too big for the smooth window reduces the number of real peaks. Therefore, smoothing should be done with small window sizes of at most 5.

3.2.2 Sliding Window Methods for Peak Detection

The sliding window method consists on finding peaks in a certain interval, or window w around a time t from the series. In this manner, some of the noise in the time series is removed, so that the original data doesn't need particular smoothing.

Algorithm 2 Double Change Method

Require: $\{s_{i,t}\}_{t=0}^{508}$
Ensure: $\{r_{i,t}\}_{t=0}^{508}$
for $t = 1 \rightarrow 507$ **do**
  if $s_{i,t-1} > 0$ and $s_{i,t} > 0$ and $s_{i,t+1}$ **then**
    $c \leftarrow 0$
    while $c < 10$ **do**
      $s \leftarrow 1 + x \cdot (c + 1)$
      $i \leftarrow 1 - x \cdot (c + 1)$
      if $s_{i,t} > s \cdot s_{i,t}$ and $s_{i,t+1} > s \cdot s_{i,t}$ or $s_{i,t} < i \cdot s_{i,t-1}$ and $s_{i,t+1} < i \cdot s_{i,t}$ **then**
        $c \leftarrow c + 1$
      end if
    end while
    $r_{i,t} \leftarrow c$
  else $r_{i,t} \leftarrow 0$
  end if
end for

It must be mentioned that the method finds the local peaks within a certain window. According to [11], a peak is a local maximum if it either has the maximum value inside a window or it is isolated among small values. The plateaus are not taken into consideration because they don't have a local maximum.

Having the time series $\{s_{i,t}\}_{t=0}^{508}$, we can find all the peaks noted with P as follows:

$$P = \{t, s_{i,t-1} | SW\,(t) > 0\},\ \forall t \in \overline{w, 508 - w}. \tag{3.3}$$

A function SW(t) is used to detect a peak within a sliding window. We will present three different functions later. If the result value of the function SW is positive, then the current t has a peak and if it is negative, it has a valley [13]. As valleys don't concern us, we consider only the positive ones. It has to be greater than 0 because all three functions that will be discussed later are based on the difference between the current value and another one inside the window.

The main steps in detecting the peaks are the following: iterate through all t in range [w, 508-w] and then apply a function to detect the peaks for [t-w, t+w].

However, in algorithm every peak is detected no matter how small. Some of them might also interpolate in sequent windows, meaning that there will be duplicates. In order to eliminate this problem, we will apply a filtering on the resulted peak values.

We use Algorithm 3 to keep only the real peaks that are higher than the mean, as it was explained in [11]. Besides this, we want to be sure that we take them only once between the sliding windows, so that they aren't too near one from another. In this case, we remove the smaller peak if there are two interpolating values.

For all three functions, if the peaks are thin, a window size of about 30 is chosen, but we will see more about this in the results chapter. The threshold used for filtering should be around 0.1 as the time series have quite small values.

The historical relevance is given by:

$$R_{i,t} = 10 \cdot \frac{p_{i,t}}{\max p_{i,t}},\ \forall t \in \overline{0, 508}. \tag{3.4}$$

Algorithm 3 Sliding Window Filter

Require: $r_{i,j}$
Ensure: p
$\mu = mean(r_{i,j})$, $\sigma = std(r_{i,j})$
$j \leftarrow 1$, $l \leftarrow length(r_{i,j})$
for $k = 0 \rightarrow l$ **do**
  if $r_{i,k} - \mu > h \cdot \sigma$ **then**
    $p_{i,j} \leftarrow r_{i,k}$
    $j \leftarrow j + 1$
  end if
end for
for $k1 = 0 \rightarrow j$ **do**
  for $k2 = 0 \rightarrow j$ **do**
    if $abs(k1 - k2) < w$ **then**
      $remove(p_i, \min(p_{i,k1}, p_{i,k2}))$
    end if
  end for
end for

Thus, the resulted peaks are scaled to 10 so that the global maximum will have the highest relevance and the rest of them will be compared to this one.

3.2.2.1 Sliding Window Function SW1

Average Difference of Neighbours Function The first function is based on the averages from the left and right of the difference between the current point t and its neighbours within the window w.

This can be computed as follows:

$$SW1(t) = \frac{\frac{(s_{i,t} - s_{i,t-1} + \ldots + s_{i,t} - s_{i,t-w})}{w} + \frac{(s_{i,t} - s_{i,t+1} + \ldots + s_{i,t} - s_{i,t+w})}{w}}{2}, \forall t \in \overline{w, 508 - w}. \quad (3.5)$$

3.2.2.2 Sliding Window Function SW2

Difference Average of Neighbours Function The second function for peak detection in the sliding window method is similar to the previous one. Here, we don't compute the average of the difference, but get the average of the neighbours from left and right, and then we subtract it from the current point t. The formula can be written in the following way:

$$SW2(t) = \frac{(s_{i,t} - \frac{s_{i,t-1} + \ldots + s_{i,t-w}}{w}) + (s_{i,t} - \frac{s_{i,t+1} + \ldots + s_{i,t+w}}{w})}{2}, \forall t \in \overline{w, 508 - w}. \quad (3.6)$$

As we will see in Chapter 6, the results for SW1 and SW2 are very similar.

3.2.2.3 Sliding Window Function SW3

Maximum Difference of Neighbours Function The idea behind the last function is to detect the maximum difference between each neighbour on the left and on the right and compute the average of these values.

Thus, we can write it more formally like this:

$$SW3(t) = \frac{\max(s_{i,t} - s_{i,t-1}, ..., s_{i,t} - s_{i,t-w}) + \max(s_{i,t} - s_{i,t+1}, ..., s_{i,t} - s_{i,t+w})}{2}, \forall t \in \overline{w, 508 - w}. \tag{3.7}$$

In this case, the results are quite different from SW1 and SW2.

3.2.3 Level Difference Method for Peak Detection

At the end of this chapter we finally present a new method based on both local maximum and minimum. We know that a peak is defined by a point (initial minimum) where it starts to increase until a point where it reaches a local maximum, followed by a decrease to a final minimum. After these steps, a function LD is applied on these values to get the peaks.

Algorithm 4 Level Difference Algorithm

Require: $\{s_{i,t}\}_{t=0}^{508}$
Ensure: $\{p_{i,t}\}_{t=0}^{508}$
$t \leftarrow 0$
while $t = N \rightarrow 508 - N$ **do**
  $min_i \leftarrow s_{i,t}$
  while $s_{i,t} > s_{i,t-1}$ **do**
    $t \leftarrow t + 1$
    $max \leftarrow s_{i,t-1}$
  end while
  while $s_{i,t} < s_{i,t-1}$ **do**
    $t \leftarrow t + 1$
    $min_f \leftarrow s_{i,t-1}$
  end while
  $p_{i,t} \leftarrow LD(max, min_i, min_f)$
  $t \leftarrow t + 1$
end while

We found two functions that use the same principle of level difference and that can be applied in the same way as seen in Algorithm 4, but with different results.

As in the sliding window methods, the historical relevance can be written as follows:

$$R_{i,t} = 10 \cdot \frac{p_{i,t}}{\max p_{i,t}}, \forall t \in \overline{0, 508}. \tag{3.8}$$

Thus, the resulted peaks are scaled to 10 as natural numbers, the highest relevance being the one corresponding to maximum level difference.

3.2.3.1 Level Difference Function LD1

One way to detect peaks is to compute the level difference between the initial minimum before the increase and the final minimum after the decrease. Having the time series $\{s_{i,t}\}_{t=0}^{508}$, this can be written as

$$LD1 = 2 \cdot \max_{t \in (m,n) \cup (n,p)} s_{i,t} - (\min_{t1 \in (m,n)} s_{i,t1} + \min_{t2 \in (n,p)} s_{i,t2}). \tag{3.9}$$

The first step is to see from which point the values start to increase by comparing the current value to the next one. This is the initial minimum represented by minm,n si,t in the formula above. The last point in which they increase and the first point in which they decrease represents the maximum, or maxt m,p si,t . We find the final minimum, minn,p si,t in a similar way as for the first one.

The initial time series may need to be filtered using a smoothing window so that the false peaks are removed.

3.2.3.2 Level Difference Function LD2

The second function is similar to the previous one, computing the level difference in the same way. The difference is that it divides the result by two times the maximum-minimum difference written as:

$$M = 2 \cdot (\max_{t \in (m,n) \cup (n,p)} s_{i,t} - \min(\min_{t1 \in (m,n)} s_{i,t1}, \min_{t2 \in (n,p)} s_{i,t2})). \tag{3.10}$$

The local minimum above is computed between the initial and the final minimums corresponding to the peak. The LD2 function can now be written as follows:

$$L2 = \frac{LD1}{M}. \tag{3.11}$$

In this way, a filtering is produced among the values resulted from the first level different function.

Chapter 4

Implementation

The program functionalities are obtained by means of five important modules written in Python, like the ones shown in Figure 4.1. Three of these revolve around the Historical Relevance Module which is thus the most important of all.

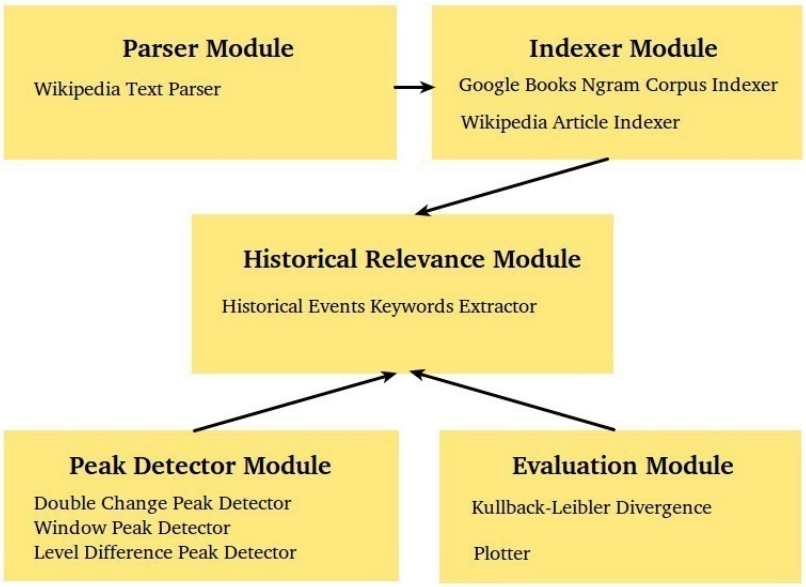

Figure 4.1: Implementation Modules Diagram

In can be observed the time series for "war" with a smoothing of 2. As we will see later on in this chapter and in the results, some methods for peak detection work better with smoothed data. However, if the window is too big, important values in the time series will be lost and the peak detection algorithms won't find some of the real peaks. This is why we will test the data with a smoothing window of size 3.

Peak Detector Module contains the algorithms that were already showed in a previous chapter. The Evaluation Module presents different methods of testing the data and comparing the results and they will be discussed in the next chapter. We will further present the features of the other three modules.

4.1 Parser Module

The events we are looking for can be found in tables like those from [ListOfWars]. At the moment there isn't any library that can retrieve exactly this information that we need, so the Wikipedia dump had to be manually parsed or with the use of some Python libraries.

Wikipedia Text Parser is not just a simple parser because it also brings the words to a desired form by removing the stop words and keeping only the nouns and verbs. First of all, it extracts all the content from the Event and Period columns as shown in Figure 4.2, along with the subsection name from the article and puts them in a list of tuples (subsection, event, period). Python's libraries used for parsing are MediaWiki Parser From Hell, Beautiful Soup and the standard library for regular expressions.

Lowest estimate	Highest estimate	Event	Location	From	To	Duration (years)	Notes, See also
60,000,000[1]	80,000,000[1]	Muslim conquest of the Indian subcontinent	South Asia	1000	1525	525	Rajput resistance to Muslim conquests, Mahmud of Ghazni, Timur, Bakhtiyar Khilji
40,000,000[2]	85,000,000[3]	World War II	Worldwide	1939	1945	7	World War II casualties and Second Sino-Japanese War[4] (this estimate includes worldwide Holocaust and concentration camps deaths)
30,000,000[5]	30,000,000	Mongol conquests	Eurasia	1206	1368	163	Mongol Empire
30,000,000[5]	30,000,000	Late Yuan warfare and transition to Ming Dynasty	China	1340	1368	29	Ming Dynasty
25,000,000[6]	25,000,000	Qing dynasty conquest of the Ming Dynasty	China	1616	1662	47	Qing Dynasty
20,000,000[7]	100,000,000[8][9][10][11][12]	Taiping Rebellion	China	1851	1864	14	Dungan revolt
15,000,000[13]	65,000,000[14]	World War I	Worldwide	1914	1918	5	World War I casualties Upper estimate includes worldwide Spanish flu deaths.
15,000,000[15]	20,000,000[15]	Conquests of Timur-e-Lang	West Asia, South Asia, Central Asia, Russia	1369	1405	37	Timurid dynasty
13,000,000[16]	36,000,000[17]	An Lushan Rebellion	China	755	763	9	Medieval warfare
8,000,000[18][19]	12,000,000	Dungan revolt	China	1862	1877	16	Panthay Rebellion

Figure 4.2: Wikipedia table of wars

The parser can also parse simple articles. Firstly, it extracts the text parts, removing Wikipedia's special tags and punctuation, then it eliminates the stop words. Using the TextBlob Aptagger library from Python, it detects the part-of-speech tags which allows keeping only the nouns and verbs for the rest of the analysis. The nouns are lemmatized with the help of the WordNetLemmatizer library so that only the base form of these words are kept. The result is a simple list of terms.

4.2 Indexer Module

Searching into a large collection of data can be extremely time-consuming, especially when we're talking of tens of GB. Apache Lucene is a very powerful text search engine that is very fast and easy to use. In this implementation we've used a Python wrapper over the original version of Lucene 4.5.1, i.e. called Apache PyLucene. Two corpora of data that are important for the current research are indexed: the Google Books Ngram Corpus and the processed Wikipedia articles.

4.2.1 Google Books Ngrams Corpus Indexer

The 1-grams from the Google Books Ngram Corpus hold a total of 25GB of data that needs to be indexed so that the search can be performed really fast. This is quite a simple module that

takes the tuples (ngram, year, match_count, volume_count) from the original Google's Ngrams Corpus and stores them as Fields in the PyLucene format. The data is stored in directories corresponding to the first letter of the 1-gram.

A fast search in the ngrams is required when computing the time series for a selected word from the article. The time series are represented as a dictionary in Python with the years as keys and frequencies as their corresponding values. Frequency is defined as the match_count for the current year divided by the total counts for all words from that year. However, it is the only module that is not used directly by the Historical Relevance Module.

4.2.2 Wikipedia Article Indexer

Wikipedia articles representing historical events must be processed before sending them to the Historical Relevance Module. Firstly, Python's Wikipedia library is used to get the text from the article in online mode and after this, the terms are extracted using the Parser Module. The fields given to the PyLucene [14] indexer are the name of the event, the period, selected words and their corresponding time series stored as strings. This approach is necessary because the main module has to process 27 articles referring to "war", each of them having about 2000-3000 words. Retrieving time series for an article would take too much time, even for an indexed data, especially when we are talking about an index directory of 25GB. In this way, the whole process takes only a few seconds.

4.3 Historical Relevance Module

This final module is the core of the application which binds three of the other four modules altogether. Its main goal is to summarize one or more articles by extracting their keywords through different methods as the ones discussed in Chapter 3.

Identification of keywords from an article is done in several steps. First of all, the information from the Wikipedia index directory including the name of the event, period, words and their corresponding time series is retrieved. In a second step, the peak detection algorithms are applied on the extracted terms from the article and are returned only the ones with a positive relevance, within the selected period plus a dispatch/delay of a few years. It is necessary to take in consideration such a time interval because on one hand, some events are reflected later in books and, on the other, Wikipedia usually contains references to the causes that determined the event.

Figure 4.3: Summarization Directory Structure

The directory structure of the obtained files can be observed in Figure 4.3. Six of the CSVs contain the keywords identified using the peak detection algorithms. The divergence.csv file contains the divergence of the original distribution and the approximated distribution resulted from the presented algorithms. This step will be detailed more in the next chapter as it is an important method of evaluation.

However, we would also like to know which are the common words detected by these six methods. The selected keywords are considered to be those that appear in more than a half of the algorithms. In this way, we have the certainty that these really are the relevant terms and not some false positives. On the other hand, some of them might be too common, meaning that they appear in more than a half of the articles and we want to eliminate them. The unique words are found in the "keywords uncommon.csv" file.

Chapter 5

Evaluation Methods

The evaluation step in the implementation is done through the Evaluation Module which is connected directly to the main part.

Some algorithms work better than the others and can give different results. A good method is the one that gives the closest approximation to the initial peaks. The peaks of the original data can be easily obtained by scaling the time series to 10, so that the maximum value reaches the highest relevance. In this section we describe two ways of testing the results. The first one uses the actual results to compute the difference between the distribution probabilities of the known data and the results. The second method is based on plotting the peaks to demonstrate visually that the methods work.

5.1 Kullback-Leibler Divergence

The Kullback-Leibler Divergence [7] is widely used in the probability theory to measure the difference between two distribution probabilities α and β . The α probability represents the expected or true data, whereas β is the model or approximation of the first one. In the discrete form it is usually written as:

$$D(\alpha||\beta) = \sum_x x \cdot \log(\frac{\alpha}{\beta}). \tag{5.1}$$

A special case is the one when or are 0. In this case, $\lim_{x\to 0} log(\frac{\alpha}{\beta})$ is also 0 because although log function is not defined in 0, $\lim_{x\to 0} x \cdot log(x) = 0$. According to Gibb's inequality, the divergence must always be a positive number [15], having equality if and only if $\alpha = \beta$.

As shown in Algorithm 5, the divergence acts differently for the 0 cases. If $pm_t = 0$, then the log function would raise an error. Mathematically, $D(\alpha||\beta) = \infty$ in this case. However, we only add a 10 so that the divergence doesn't get to be negative.

The Kullback-Leibler divergence can also be interpreted as the excess of information when approximating the real data, so that the notion of entropy can be introduced. Knowing the Shannon entropy [7] of α that is

$$Q(\alpha) = \sum_x p(x) \cdot \log(p(x)) \tag{5.2}$$

Algorithm 5 Kullback-Leibler Divergence

Require: pe, pm
Ensure: $divergence$
  for $t = 0 \rightarrow 509$ **do**
    if $pe_t \neq 0$ and $pm_t \neq 0$ **then**
      $p \leftarrow \frac{pe_t}{pm_t}$
      $divergence \leftarrow divergence + pe_t \cdot \log(p)$
    else
      if $pm_t = 0$ **then**
        $divergence \leftarrow divergence + 1$
      end if
    end if
  end for

and the cross entropy

$$Q(\alpha, \beta) = -\sum_{x} p(x) \cdot \log(q(x)) \tag{5.3}$$

then the divergence can be written as follows:

$$Q(\alpha, \beta) = Q(\alpha, \beta) - Q(\alpha). \tag{5.4}$$

The cross entropy measures the probability of an event to happen in the conditions of approximated values rather than for the real data.

5.2 Data Plots

Using more methods to test the implementation helps in finding out if the algorithms actually work and solve some of the problems, if they appear.

The data is plotted in two important points: after the algorithm and at the end, when the keywords are already extracted. Testing the results for a word in the first step helps us determine which of the methods worked best and which was the least powerful. If the observations correspond to the ones made upon the Kullback-Leibler divergence, then it means that they behaved correctly. In the next step is selected a certain number of keywords (5 in general) from the ones that represent the event. Further, the time series for these ngrams are plotted. If all of these tend to have a climax in the period corresponding to the event, we have the certainty that they were chosen correctly and that the method worked.

Chapter 6

Results

A method to detect if the algorithms really worked is to plot the resulted data. An important Python library like matplotlib was used for this part of the implementation. Further, we will present some of the graphical results and make some observations upon them.

6.1 Peak Detection Results

As mentioned in the implementation chapter, the analysed events are those referring to "war", so let's see how this word is used in time. These plots contain two functions of the time series: the first is the one corresponding to the approximated result and the second, to the original data.

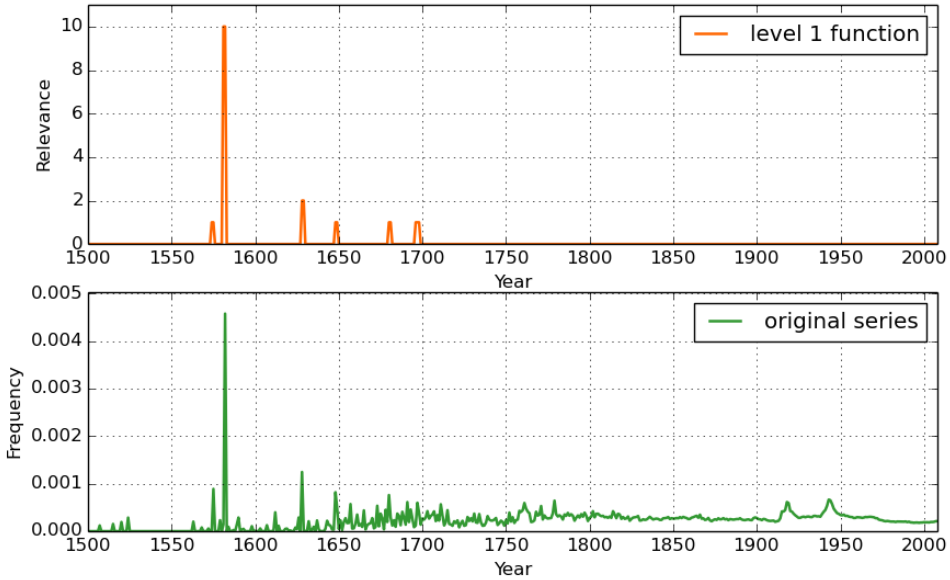

Figure 6.1: Level Difference Method LD1 with no smoothing

In Figure 6.1 we can observe that only a few of the real peaks are shown within the first method and the ones corresponding to the World War I and World War II are neglected. Still, the result

is somehow expected. These values may be very real small, so that the scaling step omits them when it transforms floats to integers.

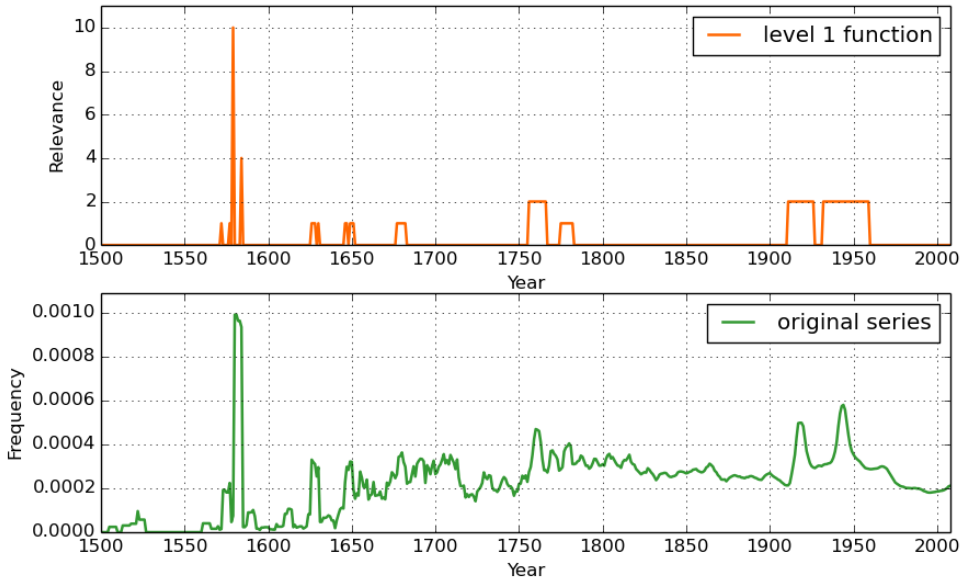

Figure 6.2: Level Difference Method LD1 with a smoothing of 2

A smoothing of two in Figure 6.2. reveals the omitted events from the previous chart. Now we can clearly see some of the most important wars over the time, such as The Ruso-Crimean War (1571), The Cretan War (1645-1669), World War I (1914-1918), World War II (1939-1945).

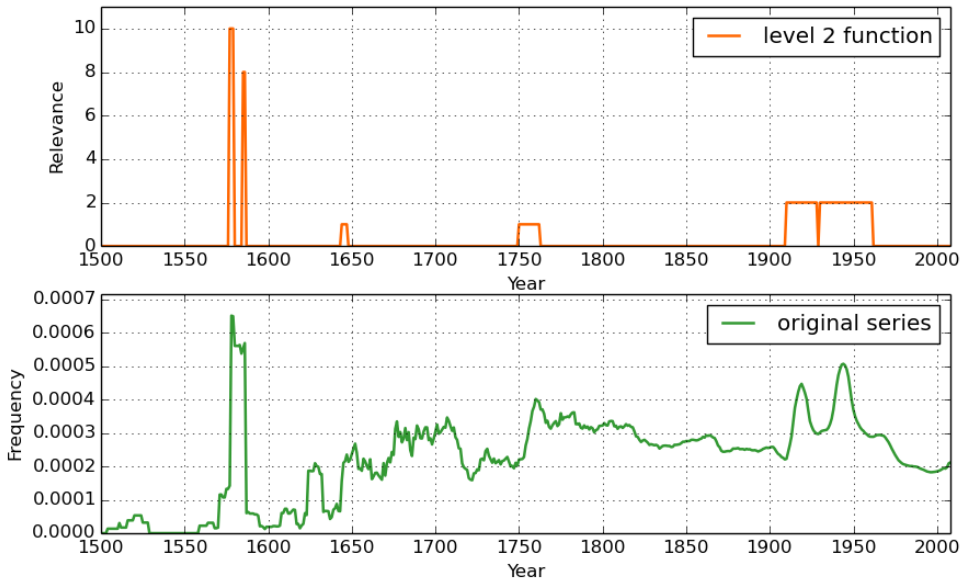

Figure 6.3: Level Difference Method LD2 with a smoothing of 2

The results from Figure 6.3. with the second method of level difference are quite similar to the first, though some of the peaks between 1650 and 1750 are neglected. The values are closer one to another forming this kind of plateau shape like the one corresponding to the 1925-1955 period (second peak from the right).

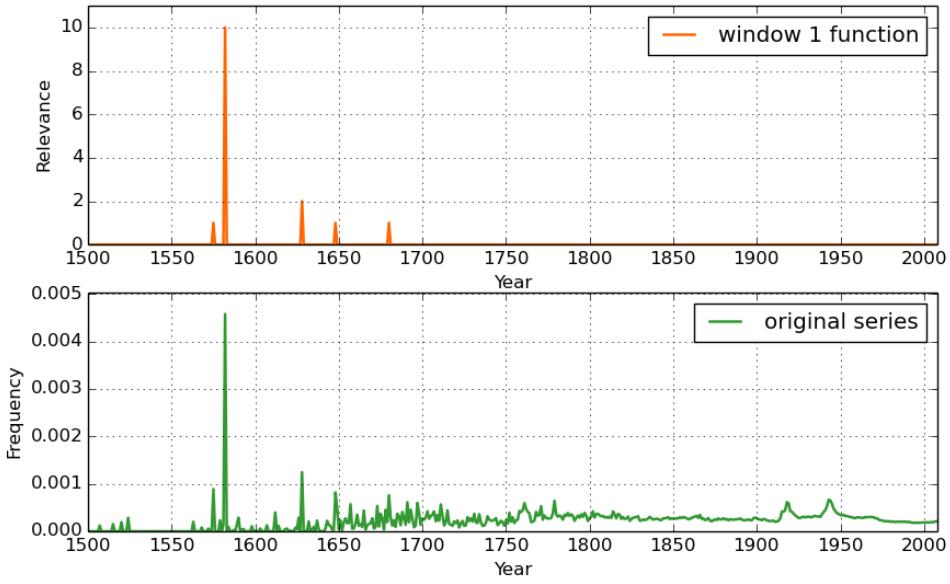

Figure 6.4: Sliding Window Method 1 with no smoothing

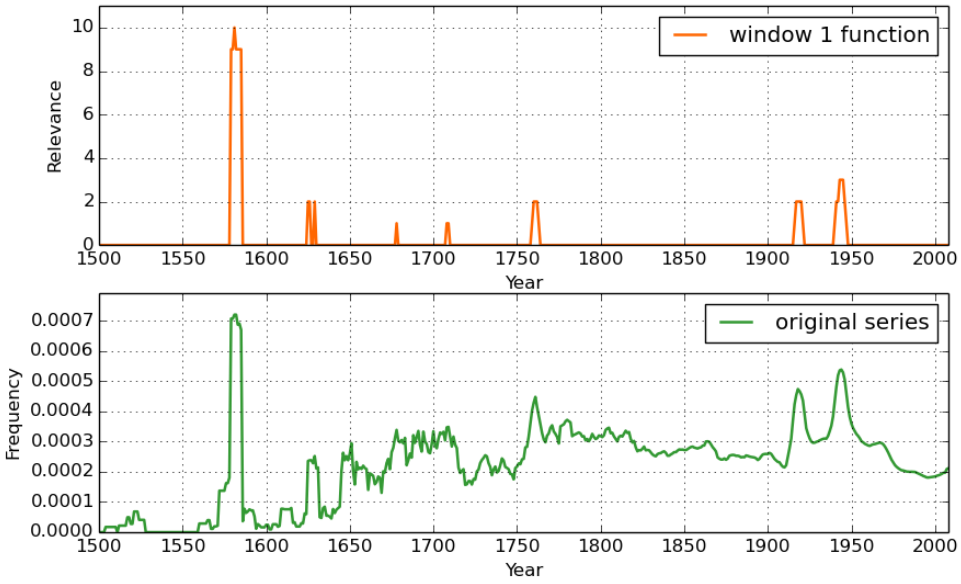

Figure 6.5: Sliding Window Method 1 with a smoothing of 3

The Sliding Window Methods 1 and 2 give similar results for a window of 30 and with the level methods also. Figure 6.4 shows how the methods work for the series with no smoothing.

A smoothing window also improves the results, as can be seen in Figure 6.5. Unlike the previous methods that give plateau-shaped peaks, in the sliding window method they are more like triangle-shaped, suggesting that it exists a year corresponding to the event when it was reached a climax.

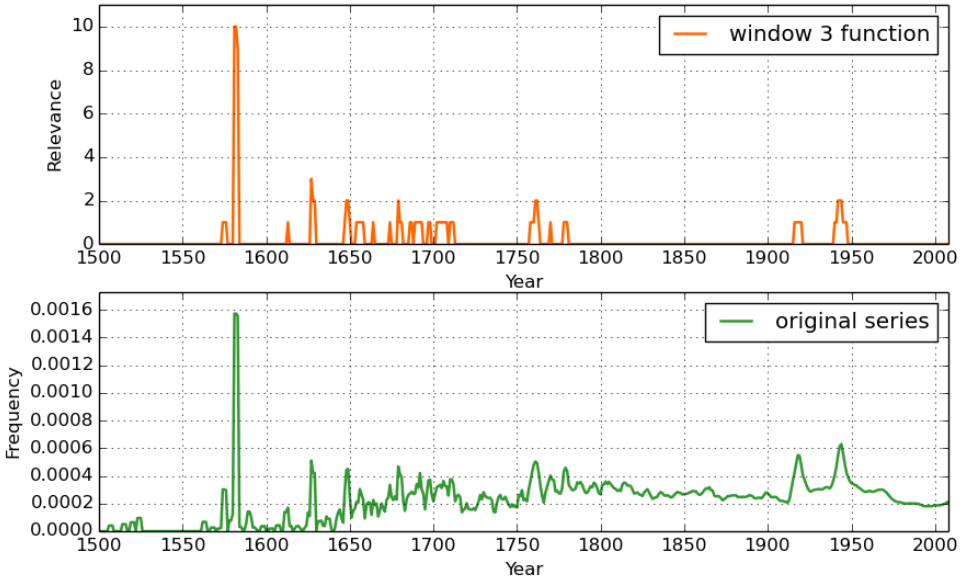

Figure 6.6: Sliding Window Method 3 with a smoothing of 3

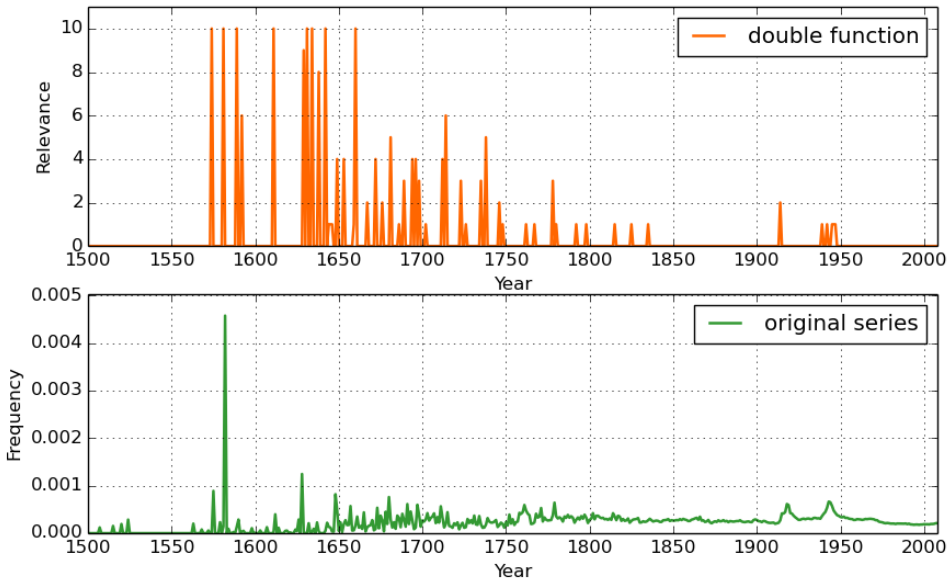

Figure 6.7: Double Change Method with no smoothing

The third Sliding Window Method is a bit different from the other two because it computes the average of the maximum between the left and right neighbours of the current point and not the real average. Thus, the results are closer to the real form of the time series, but still keeping the triangle shape that defines the method.

In Figure 6.6 can be observed the detected peaks, with a smoothing of 3. As the smoothing window gets higher, the period that defines an event gets extended. Therefore, the smoothing window must be chosen carefully so that the false positives are avoided.

The Double Change Method doesn't filter peaks at all, as can be seen in Figure 6.7. When abrupt increases and decreases are detected in a small period, the years from the times series with the highest values are assigned the highest relevance.

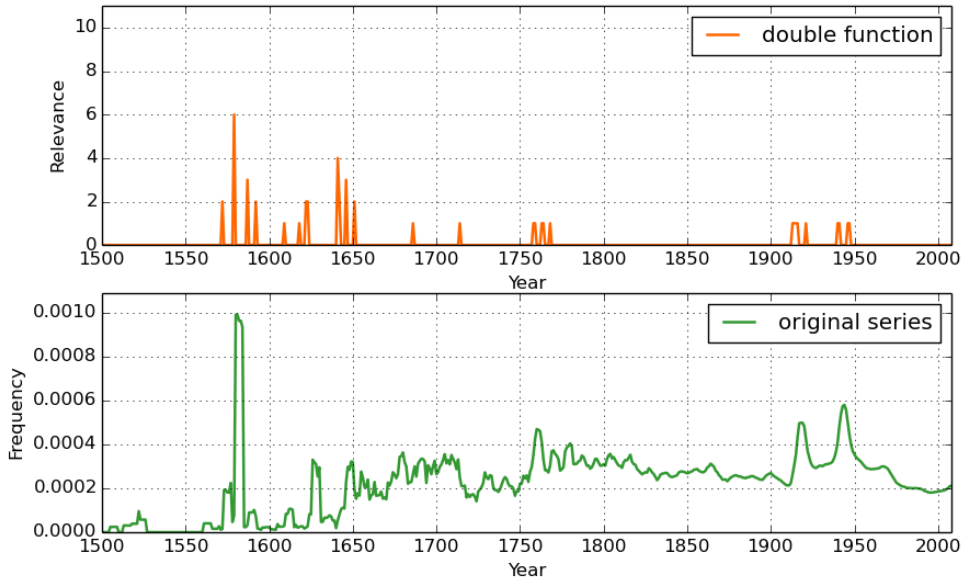

Figure 6.8: Double Change Method with a smoothing of 2

Smoothing the time series reduces the highest peaks, as shown in Figure 6.8, but also produces a sort of a filtering. This is why, the smoothing window values should be really low, like 1 or 2.

As we can observe, the third Sliding Window Method seemed to be the most accurate, being the closest to the real model and the first Level Difference Method was the weakest of all.

6.2 Kullback-Leibler Divergence Comparison

In order to obtain a general comparison between the real data and the approximation, it is computed an average of the Kullback-Leibler Divergence for all words within the article with a positive relevance. The distribution that approaches 100 the most represents the best method, while the one that approaches 0 is the weakest method.

For example, the results seen in Table 6.1 and Table 6.2 show that the best method is the second Level Difference function.

Method	Divergence
level LD1	62.07
level LD2	73.20
window SW1	59.55
window SW2	59.55
window SW3	69.92
double change	64.28

Table 6.1: Kullback-Leilbler Divergence for American Civil War

Method	Divergence
level LD1	71.30
level LD2	85.85
window SW1	76.82
window SW2	76.82
window SW3	82.67
double change	78.67

Table 6.2: Kullback-Leilbler Divergence for World War II

6.3 Resulted Keywords from Historical Events

After running the Historical Relevance Module on an article, several files resulted. One of them contains unique words that represent the event, common for all methods.

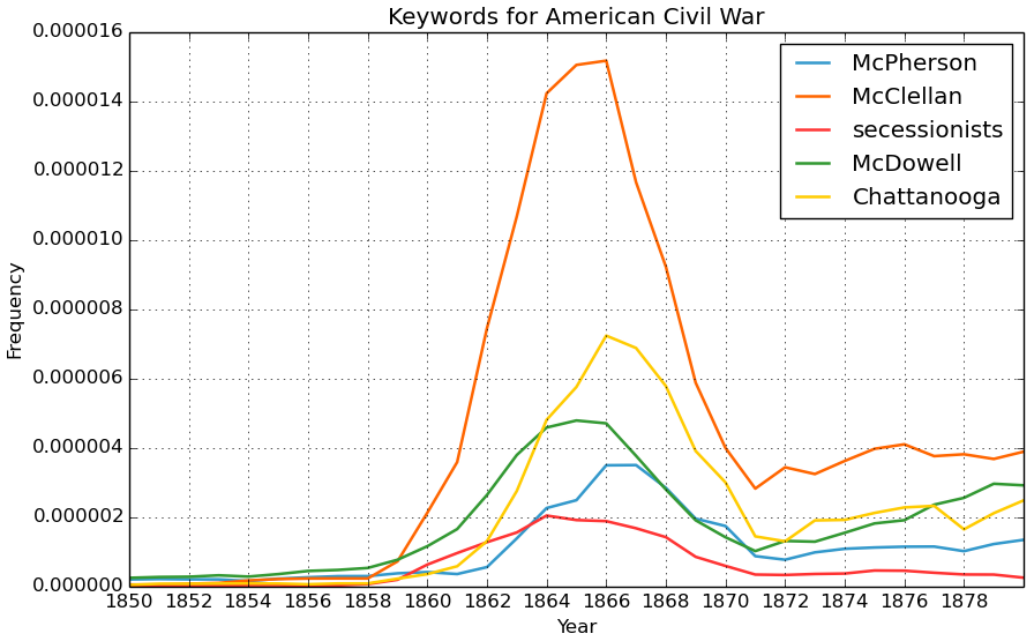

Figure 6.9: Ngram time series for the keywords of American Civil War, with smoothing of 5

In Figure 6.9 are shown 5 of the most important words used during the American Civil War. Many of them are related directly to this event, for example, "secessionists", or "McClellan" that reach the highest peaks in this period, demonstrating the fact that the methods worked.

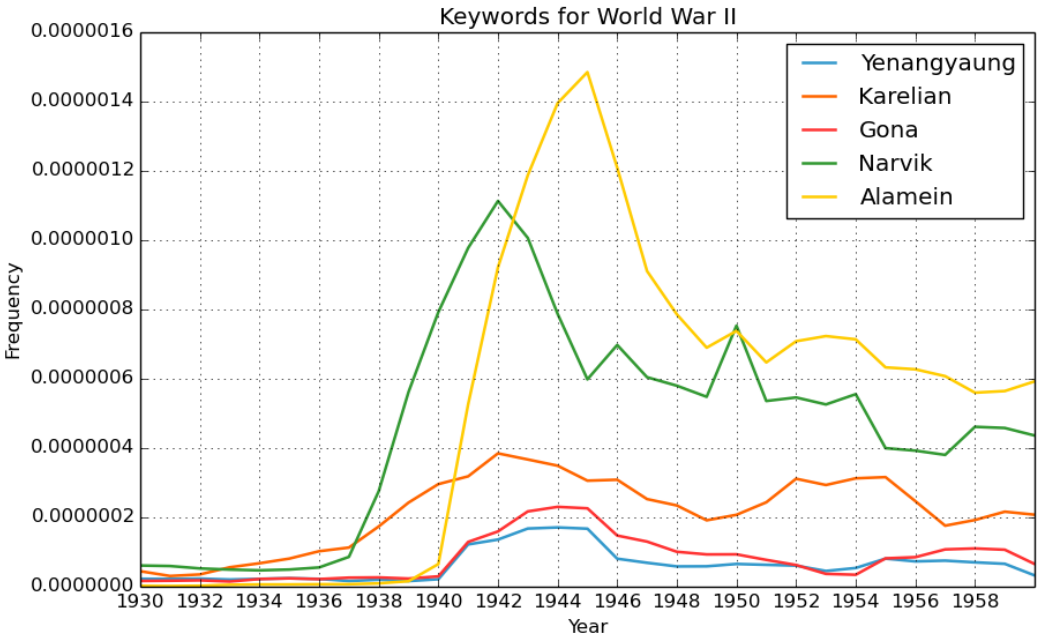

Figure 6.10: Ngram time series for the keywords of World War II, with smoothing of 3

The keywords near the World War II appear in Figure 6.10. It can be observed that these words have peaks in the period 1938-1945, corresponding to this important historical event.

In table Table 6.3 can be found five of the most relevant keywords that define the American Civil War for each of the presented methods for peak detection. We can observe that although most of them are the same, their historical relevance and order differ.

LD1	**LD2**	**SW1**	**SW2**	**SW3**	**D**
McDowell,10	McDowell,10	McClellan,9	McClellan,9	Chattanooga,6	Chattanooga,10
Chattanooga,10	Chattanooga,10	McDowell,7	McDowell,7	secessionists,6	McPherson,6
secessionists,10	secessionists,10	secessionists,6	secessionists,6	McDowell,3	McDowell,4
McPherson,10	McPherson,10	McPherson,6	McPherson,6	McClellan,3	McClellan,4
McClellan,9	McClellan,10	Chattanooga,2	Chattanooga,2	McPherson,1	secessionists,2

Table 6.3: Top 5 Keywords for American Civil War

A list of ten of the most common words from the articles that define "war" can be seen in Table 6.4. These were selected from all articles and sorted in decreasing order by the historical relevance. As it can be observed, all ten words have a historical relevance of nine.

Keyword	Historical Relevance
Aguiyi	9
Passagem	9
cuius	9
Catholicism	9
Edo	9
Tangistanis	9
Bundy	9
Karelian	9
Jesuit	9
Sufragio	9

Table 6.4: Keywords that define "war"

Table 6.5 shows that the overall best method is the third Sliding Window Method, while the weakest of all is the first Difference Level.

Method	Divergence
level LD 1	61.05
level LD 2	69.10
sliding window SW 1	66.83
sliding window SW 2	66.78
sliding window SW 3	70.47
double change	64.02

Table 6.5: General Kullback-Leilbler Divergence for all articles

Chapter 7

Conclusions

In this research a viable framework has been presented for identifying keywords from historical events that are reflected in Wikipedia articles. The historical relevance of a word is an important component of this framework as it assigns relevant scores to the terms from the articles. This approach has proved successful in detecting the words that uniquely define an event.

Several methods for peak detection were presented and compared in our effort to identify the most accurate of all. Furthermore, a study of the results has shown that all of them do a good work in identifying the keywords.

One of the most important directions for future work consist in detecting the delay in the Google Books Ngram Corpus when an event has occurred. Sometimes, it is reflected in the corpus after a certain time after it was produced. In this case, a longer period may need to be considered when identifying the keywords from an article.

Another idea to improve the existing framework is to reverse the steps used in this research for an event detection approach. For example, having the keywords that define several events, apply machine learning techniques so that the algorithm learns to classify them and detect even more events.

Bibliography

[1] P. D.Allison, Event History Analysis: Regression for Longitudinal Event Data, 46th ed., 1984.

[2] J. M. Box-Steffensmeier and B. S. Jones, Event History Modeling: A Guide for Social Scientists. Cambridge University Press, 2004.

[3] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, S. Pinker, M. A. Nowak, and E. L. Aiden, "Quantitative analysis of culture using millions of digitized books.," Science, vol. 331, no. 6014, pp. 176–82, Jan. 2011.

[4] "Part-of-speech tagging - Wikipedia, the free encyclopedia." [Online]. Available: http://en.wikipedia.org/wiki/Part-of-speech_tagging. [Accessed: 02-Jul-2014].

[5] S. J. DeRose, "Grammatical category disambiguation by statistical optimization," Comput. Linguist., vol. 14, no. 1, pp. 31–39, Jan. 1988.

[6] M. Honnibal, "A good POS tagger in about 200 lines of Python | Computational Linguistics on WordPress.com." [Online]. Available: http://honnibal.wordpress.com/2013/09/11/a-good-part-of-speechpos-tagger-in-about-200-lines-of-python/. [Accessed: 02-Jul-2014].

[7] Wikipedia [Online]. Available: http://en.wikipedia.org/wiki/Main_Page. [Accessed: 02-Jul-2014].

[8] "Wikipedia:Database download." [Online]. Available: Wikipedia:Database download.

[9] List of wars and anthropogenic disasters by death toll [Online]. Available:wiki/List_of_-wars_and_anthropogenic_disasters_by_death_toll. [Accessed: 02-Jul-2014].

[10] T. Ramalho, "An introduction to smoothing time series in python." [Online]. Available: http://www.nehalemlabs.net/prototype/blog/2013/04/05/an-introduction-to-smoothing-time-series-in-python-part-i-filtering-theory/. [Accessed: 02-Jul-2014].

[11] G. Palshikar, "Simple Algorithms for Peak Detection in Time-Series," 2009.

[12] T. Popa, "Historic Events Detection in a Large Collection of Texts," POLITEHNICA of Bucharest, 2013.

[13] R. Schneider, "Survey of Peaks/Valleys identification in Time Series," Department of Informatics, University of Zurich, 2011.

[14] Apache Lucene [Online]. Available: http://lucene.apache.org/. [Accessed: 02-Jul-2014].

[15] C. Shalizi, "Shannon Entropy and Kullback-Leibler Divergence," in Advanced Probability II or Almost None of Stochastic Processes, 2006.

[16] Google Ngram Viewer [Online]. Available: https://books.google.com/ngrams. [Accessed: 02-Jul-2014].

List of Keywords for War Events

Event	Period	Keywords
American Civil War	1861-1865	Pickens, McClellan, slaveowners, McDowell, Chattanooga, secessionists, McPherson, Vicksburg, Braxton, Rosecrans
Conquests by the Empire of Japan	1894-1945	Nomonhan, Edo, Fukuzawa, Klemen, Kanai, Higashifushimi, Yonai, Fushimi, Kotohito, Sakoku
Deluge	1655-1660	harassed, Crowns, Blessed, prove, parliament, mistake, unaffected, Brethren, discontented, Elector
Dungan revolt	1862-1877	Sarikol, Yakoob, Athalik, tungan, Tashkent, retill, Kashgari, Kokandi, dungans, Kokandis
Dungan revolt	1895-1896	Oxheart
French Wars of Religion	1562-1598	Roche, Laye, repress, Medici, engaged, step, Passions, ruleth, Selected, gallows
Iran–Iraq War	1980-1988	Tektronix, petrochemical, manipulated, RPG, Jasim, professionalization, Kuwaiti, Nasser, hovercraft, crackdown
Korean War	1950-1953	rotorcraft, NPA, USAMGIK, Malenkov, earmarked, Zakharov, Munsan, Beria, Pyongtaek, Kaesong
Qing dynasty conquest of the Ming Dynasty	1616-1662	literati, Agreeing, marching, shave, hanged, relieving, hill, Pass, tribe, ordering
Mexican Revolution	1911-1920	Sufragio, Creelman, Paso, Hermila, Madero, Venustiano, Hardman, Chihuahua, Efectivo, Huerta
Napoleonic Wars	1803-1815	Aube, Thielmann, Wittgenstein, Wagram, Polotsk, Walcheren, Tolly, Aspern, Oudinot, Brune

Nigerian Civil War	1967-1970	Biafran, Achebe, Biafrans, Umuahia, Adekunle, Ojukwu, Aguiyi, Unegbe, Soyinka, Gowon
Paraguayan War	1864-1870	Passagem, Venancio, Paraguayans, Paraguayan, Riachuelo, Caxias, Urbieta, Curupaity, Platine, Narciso
Russian Civil War	1917-1921	Priamur, Kornilov, Semenov, Semyonov, Ekaterinburg, Dunsterville, Kerensky, Czechoslovaks, Follett, Omsk
Second Sudanese Civil War	1983-2005	COMECON, Murle, Dinka, Sudan, Khartoum, Sudanese
Shaka	1816-1828	Chunu, Stanger, Wylie, Pimlico, echelon, kraal, taunt, muzzle, Terrific, park
Soviet War in Afghanistan	1980-1988	malnourished, Kosygin, UNDP, Internationalist, technocrat, Zbigniew, Amstutz, Pashtun, Pakistani, Bhutto
Spanish Civil War	1936-1939	Anarcho, Franquists, Falangists, Belchite, Niceto, Lerroux, Gascón, Pasarán, Brunete, Blum
Taiping Rebellion	1851-1864	Flashman, exhumed, Paterson, Amy, Mitter, Lovell, Dagang, Hanying
Thirty Years' War	1618-1648	catastrophe, Parker, Huguenots, lawlessness, Omar, abetted, Catholicism, Jesuit, duplicity, cuius
Vietnam War	1955-1975	Bundy, McGovern, counterinsurgency, NLF, Thieu, ARVN, regroupment, counterculture, bargirls, Cambodians
World War I	1914-1918	Caporetto, Allenby, Scapa, Sonnino, Tangistanis, demobilise, Clemenceau, Baralong, nitrogen, Magdhaba
World War II	1939-1945	Yenangyaung, Karelian, Gona, Locarno, Alamein, Stresa, Narvik, Antonescu, Katyn, Moresby

Table 1: Unique Keywords for War Events