Abstract

Text searching, categorization, and summarization are important problems in information retrieval research. One of the most common approaches to text analysis is to exploit the term frequency-inverse document frequency (tf-idf) vector model, which is very effective and efficient in representing a large document through a small vector. The tf-idf approach has the crucial drawback that it only considers the text in terms of the structure of composition. However, each natural language has its own syntactic structure. Thus, it is not sufficient to replace the text with a set of important keywords without taking into account their relative relation to the thesis and meaning of the text. In this paper, we propose a text search model based on a keyword graph model, which is based on the cognitive process (writing) model. We show how to construct a keyword graph from a text by assigning edges between two vertices (keywords) if their regions of influence overlap. Our approach allows the use of the text as a query attribute. In our model, if a user wants to find text similar to a given query text in a large repository, the query document can be searched without selecting keywords. This query-by-example in text searching is an important contribution of our work. Experiments show that our keyword graph model is superior to the tf-idf model in clearly and effectively revealing the similarity between documents. Our experiments use more than 2,000 speeches obtained from the United States White House, and show that our approach is superior to prevalent text search methods in terms of accuracy of syntactic similarity and the semantic structure of object texts.

I. Motivation

With the rapid evolution of information technology, an ever-increasing and overwhelming amount of information is becoming available, and this includes textual information. A significant amount of intellectual activity is thus devoted to gathering, classifying and analyzing text documents. Big data is an important concept in contemporary data analysis, where textual information forms the majority of the big data pool. Numerous automated systems and methods are being studied nowadays to deal with large volumes of text data beyond the capacities of traditional human read-and-write cognitive process. These methods need to take into account the trade-off between speed and accuracy. Consider the problem of text categorization. One simple way of doing this is through a simple reading and understanding of the semantic structure of the document at hand. However, this is not an efficient method if more than a million documents have to be processed. Hence, the accuracy of understanding generally conflicts with the speed of text processing.

A reliable model for searching texts is the text frequency-inverse document frequency (tf-idf) framework, which transforms a document into a simple real number vector. The size of a tf-idf fingerprint is dependent on the size of the selected words and not the size of the document. Despite its simplicity, experience has shown that methods based on tf-idf, or similar metrics based on a bag-of-words model, perform relatively well in many circumstances. In particular, tf-idf works well in analyzing the global characteristics of text data, such as text categorization and classification. However, as far as analyzing underlying semantics of texts is concerned, tf-idf is too weak to reveal the meaning of complex linguistics structures because it is a typical model for compositional analysis that does not consider the subtle syntactic structure of words.

In this paper, we introduce a new text searching method based on the cognitive reading and writing model. We assume that we have a large number of documents in a text repository and are asked to find a set of documents with similar content for a given text query \( q \). We can use the tf-idf model for this with a few preprocessing steps. We first obtain a list of important words in \( T_q \) and search for similar texts through a set of keywords obtained from a query document. Although this approach is efficient, it does not consider the linguistic structure of texts. In order to address this shortcoming, we introduce a keyword graph, a new characteristic feature of text data. We construct a pair of related keywords if the influence regions of two keywords overlap. Applying this process to the entire text, we construct a keyword interaction graph model for a given text. The keyword graph requires more information than the simple tf-idf model.

The remainder of this paper is structured as follows. We summarize previous work on text analysis in Section 2. Section 3 and 4 are devoted to the preliminaries of keyword graph transformation, and the meanings of theoretical graph parameters are explained in terms of textual meaning. We propose an algorithm to extract a keyword graph from a given text. In Section 5, we test our proposed method using around 2,000 documents obtained from the White House, the official residence and workplace of the United States president. We will show how to select texts similar to the one at hand in terms of semantic contents. We conclude in the final section, and reflect on the space for interesting problems for future research created by our contribution.

II. Related Work

Text mining or analytics is used to derive useful information from text data. One approach to this is to perform a statistical pattern learning search to locate frequent patterns. A typical
mining procedure consists of text categorization, clustering, concept learning, sentiment analysis and summarization. A new variation of text mining based on social network analysis has recently been developed. Since social networks contain large amounts of text data, including personal information, it would be interesting to embed a social network structure in the text space.

In text classification and retrieval, the first step is to obtain a tf-idf fingerprint from the object text. Term frequency-inverse document frequency is the main tool in text analysis, such as in text categorization and summarization [1], [2], [3]. There several applications of the tf-idf measure in text analysis and semantic analysis. Feature selection is a crucial procedure in text mining [4], [5], [6], [7] provide a useful survey of traditional analysis techniques in text mining. Local and global dictionaries from domain texts appear to be important data structures for all previous research.

Text visualization is another important issue in text searching. There are numerous text visualization methods and systems where a few keywords are introduced by adjusting the text size and position according to its relative importance [8]. Wordle (www.wordle.net) is the most successful online text visualization tool at present.

An interesting approach to extract social relationships among humans using text has recently been proposed. Elson et al. have presented a method for extracting social network relationships using nineteenth-century British novels [9]. Further, a few physicists have successfully generated the characteristic structure in Korean literature using a complex system model. Two characters are typically provided in edge relation if they are in conversation. Emails are used by McCallum et al. as typical textual data to explore social network relationships. Some researchers have proposed graph models to characterize text structure [10], text space to characterize clustering [11], [5], artificial social network relationships [12], and graph models [13], [14], [15], the fuzzy set concept [16] and wordnet [17] for text analysis. In all these approaches, the co-occurrence of two subject words is a primitive fact used to analyze target texts [18].

Others researchers have proposed a method to automatically construct a signed social network using text to obtain resulting networks with the polarity property [19]. In a management information system (MIS), the networks of keywords is studied to obtain the implications of predicting knowledge evolution. Some psychologists have also studied mining structural information from a given text by observing stop lists, word stemming, and dimensionality using Singular Value Decomposition (SVD) [18] or a k-Nearest Neighbors (k-NN) algorithm [20].

Information retrieval is a crucial issue in text processing. Text retrieval generally begins with finding optimal attributes (keywords). Text comparison is then replaced by the comparison of the fingerprints of the given texts. Tf-idf is a typical approach for this. Although the "fingerprint vector" comparison model is very efficient, it does not take into account the complex semantic content of texts. Various similarity measures have been proposed to overcome the drawbacks of composition-based text retrieval. However, keyword selection is generally not easy for beginners, and poorly selected keywords will not yield satisfactory results. To address issues with keyword-based search, the "query-by-example" model is considered. In this model, we are not concerned with how to select keywords to optimally represent the entire text, but merely "throw" the query document into a text repository. Our main concern in this paper is to develop a "query-by-example" searching method in text retrieval.

III. Preliminaries

As mentioned in the foregoing, we use a graph "representative" of text searching rather than the simple keyword fingerprint used in most previous research in the area. In this section, we propose an algorithm to construct a keyword interaction graph from a given text. Let us assume that we have a large collection of text, namely a Big Document Repository BDR = \{T_i\}. For any \(T_i\), we already have a set of keywords \(K_i = \{k_{i,1}, \ldots, k_{i,m}, \ldots\}\). In this paper, we select words that frequently appear in the text while disregarding "stop words" or meaningless verbs such as "make," "get," "is(are)," etc. We set 50 \(\leq |K_i| \leq 60\) and construct the corresponding keyword interaction graph \(G(T_i, K_i)\).

\(T_i\) can be considered a sequence of words, and thus \(T_i = (w_{1,i}, w_{2,i}, \ldots, w_{k,i}, \ldots, w_{n,i})\), where the first word of \(T_i\) is \(w_{1,i}\) and the last word is \(w_{n,i}\). The length of \(T_i\) is defined as the number of words in \(T_i\); \(|T_i| = n_i\). We now need to prepare the keyword set from \(\bigcup_k w \in T_i\). There are two methods for this: a manual/supervised mode and an automated mode.

In the manual mode, the result of the collection procedure is dependent on the user. Hence, for instance, if we want to focus on some respects of the economy in our text search, we can select technical words related to the economy, such as GDP, inflation, stock, index or oil price. In the automated mode, we first construct a global dictionary from \(T_i\), and then delete stopping words such as the articles ("the," "a" and "an"), demonstratives ("this," "that"), determiners ("which") etc. We then obtain important words that appear frequently in the text.

We thus obtain a set of keywords, \(K_i = \{k_{i,1}\}.\) Other researchers have tried to use keyword relations to extract textual knowledge in an MIS [21]. It is easy to see that a few keywords should be identical to a few words in \(T_j\). These words are the key markers in our keyword interaction graph. It is quite probable that if two keywords \(k_{i,x}\) and \(k_{i,y}\) occur close to each other in the original text \(T_i\), they are closely semantically related. This is the basic concept of a keyword interaction/influence graph.

IV. Keyword Influence Graph and its Application

A. Overview of our model

The following figure explains the structure of our approach. We want to compare texts \(T_a\) and \(T_b\) for deep semantics. It is very difficult to compare them directly in an automated manner, since this requires complicated natural language tools and systems. Simplifying the texts into linear vectors (tf-idf) may cause loss of important information due to the generic simplicity of the keyword vector. We thus propose our graph
model, which is simpler than textual comparison and more informative than the linear vector model.

The most important issue in our model is the construction of the "keyword" graph structure using texts to reveal the underlying semantic structure of the subject text in question. We believe that the linear structure of words in any text has strong features in common with any other text, regardless of the natural language. This is based on Chomsky’s seminal idea of a general language model.

B. Text structure from the view of Cognitive Process Model

All previous work related to network extraction from text is based solely on the syntactic structure of the text, and considers the sequence of words or the adjacency of word pairs, as well as some semantic features (such as sentiment analysis).

In this paper, we wish to follow a different direction, i.e., a cognitive process frame. Since human reading and writing is a complicated mental process, there are several models purporting to explain the writing/reading process. The considerable research on cognitive models of human writing primarily focuses on cognitive processes involved in writing, such as how words are prepared in human memory and how they are arranged and selected to be written [22], [23].

One reliable model that is generally accepted as standard classifies memory into short-term memory and long-term memory. Words are move back and forth between the two classes of memory in this model. This implies that all words stored in the human brain are managed in a hierarchical manner. When a word is used in a particular place in the text, it has been prepared in advance in the short-term memory and is ready to be "fired" onto the paper. Hence, if we take into account the lifetime of such a word in the short-term memory, we can extract a more accurate interaction model for words by comparing it with a syntax-only word interaction model. We now introduce keyword graph extraction in Figure 2.

C. Constructing Keyword Influence Graph

In the figure, the thick horizontal line shows the sequence of text: left for start, right for end. The location of different keywords is marked by small geometric symbols, such as ▲, △, ◊, □. In the following, \{ki\} denotes keywords selected from text Ti. Let ki(r) denote a keyword in ki appearing in the r-th order, since a keyword may appear more than once in a text. We now introduce the important concept of the "influence region of a keyword" in Ti. If a keyword ki(r) appears frequently in a sentence or over the course of a few sentences, we know that the word ki(r) is kept in short-term memory for a while. Otherwise, if a keyword appears sparsely in the text, it is seldom kept in short-term memory, or is moved back to the long-term memory. Since the actual cognitive mechanism of the brain is unknown, we estimate the state of the writer’s brain in terms of the appearing patterns of selected keywords.

If the textual distance between two occurrences of a keyword is within the threshold δ, we say that the instances form a segment of a region of influence. Thus, the more frequently a keyword appears, the greater the region of influence. In this paper, intervi(ki, r) represents the r-th appearance of the keyword ki. The threshold value δ is very important to obtain a useful keyword influence graph (keyword graph for short). A large value of δ leads to a greater number of influential segments.

The interval intervi(ki, k) (for all k) is assigned a vertex of the keyword graph Gi(K), where K is a list of keywords and Ti is the text at hand. This means that the number of vertices of Gi(K) is the same as |K|. The vx of Gi(K) denotes the corresponding vertex of keyword kx. We assign edges to Gi(K) as follows. If two intervals intervi(kp, k) and intervi(kq, l) overlap horizontally, we introduce an edge between vpx and vqy, which renders Gi(K) a parallel edge graph.

As is easy to see in Figure 3, edges should be added at \{(1, 3), (3, 2), (1, 2), (3, 1), (4, 1)\} from left to right. Since Gi(K) is a simple graph, \{(x, y)\} should be the same as \{(y, x)\}. Note that the distance between k(4) and k(5) is greater than the threshold δ, and thus these two keywords cannot form a continuous segment of influence. The number of connected components of Gi(K) will increase when we set a smaller δ and admit fewer keywords to K. The distance measure can use words as well as statements as a unit of measurement. When we use a "statement"-based distance measure, we count the number of statements between two occurrences of a keyword. If two keywords appear in a statement, the distance is 0. In
this paper, we use a "statement"-based distance measure, since the writing cognitive model treats statements as the unit of cognitive action.

D. Applying keyword graph to text analysis

Since a keyword graph contains more information than a tf-idf fingerprint, it can be applied more extensively and efficiently than tf-idf. It is quite natural that if two documents are similar, so are their keyword graphs. The central issue is the converse: whether, if two similar keyword graphs are obtained from two different texts, the texts in question are sufficiently similar. We test this in our experiment, which is described later. Analyzing $G_i(K)$ is not easy, as general graphs methods (such as subgraph isomorphism and clique finding) are difficult to characterize. Thus, we simplify the entire keyword graph into a spanning tree (with maximal edge weights). Let $S_i(K)$ represent the spanning tree. The node with the maximal degree of $S_i(K)$ is regarded as the most significant keyword in $K$, since the keyword represented by this node frequently appears in the text and the segments of influence of other keywords partially overlap with this word.

In previous work, we proved that the most insignificant keywords are found in the terminal nodes of the spanning tree of social networks extracted from fictional text [24]. Therefore, the leaf nodes in $S_i(K)$ indicate the important characteristics of the semantic structure of the object text.

If $G_i(K)$ is very dense and $G_j(K)$ is sparse, it means that the content of $T_j$ is not deeply related to the subject represented by $K$. In our experiment, we found that if $T_i$ is related to $K$, $G_i(K)$ is likely to be connected to a single component. Thus, the number of components of $G_i(K')$ for a given $K'$ reveals important information. Therefore, if we want to compare the contents of two documents $T_i$ and $T_j$ in terms of a particular subject (which appears in the keyword list), it is simpler and more useful to compare $G_i(K)$ and $G_j(K)$, or $S_i(K)$ and $S_j(K)$, rather than the entire texts $T_i$ and $T_j$.

We explain the procedure using the flowchart in Figure 4. The input consists of two parts: the subject text and a list of keywords automatically selected from the global word index in $\{T + i\}$, the document repository. We can also select a few keywords manually in order to focus on technical terms related to specific subjects, such as the economy, world politics, entertainment, the global environmental crisis, etc. Once we obtain the keyword influence graph $G_i(K)$, we may analyze it directly or through a simplified spanning tree $S_i(K)$.

Once we have successfully constructed $S_i(K)$, this tree can be transformed into a hierarchical rooted tree by choosing $v_r$, the node in $S_i(K)$ with the maximum degree, as shown in the figure. We regard $v_r$ as the most significant keyword among $K = \{k_i\}$. We can thus classify the document through $v_r$. For complex semantic analysis, we can also apply several graph similarity measures or tree similarity measures in order to compare the "semantic" contents of the selected text $\{T_i\}$. Further, we can rank keywords according to importance in terms of the tree level of each node $i$ of $S_i(K)$. In this case, tree depth (distance from root) or height (distance from the leaf node) may be selected as the referent, depending on the purpose of the application. For example, when we apply a depth measure, the importance of the keywords is linearly ordered as follows:

$$v_r \Rightarrow \{a, b, c, d, e\} \Rightarrow \{l_1, l_2, l_3\} \Rightarrow \ldots$$


E. Text searching with query-by-example

The most common approach in document search primarily relies on a few keywords, as in most digital libraries or searching portals. We apply our keyword graph to search by document unit query, which means that we do not rely on appropriate keyword selection to specify the target document. In order to search a document similar to a sample query document $T_Q$, $T_Q$ can be used as a query unit. If $T_Q$ is submitted, we first construct a list of frequent words using $T_Q$, and disregard stop words and articles.

Let $T_i$ denote a set of texts in a text repository. We want to find a "similar" text in $T_i$ for a given query text $T_Q$. The preprocessing step involves constructing the list $f_Q$ of important frequent keywords in $T_Q$. $f_a$ denotes the selected (and preprocessed) keyword list in the text $T_a$. For this, we first parse $T_Q$ and select the most frequent words in it. We then construct a keyword influence graph $G_Q(V, E, f_Q)$, where words in $f_Q$ are used as the vertex set of $G_Q$. It should be noted that we remove all isolated vertices in $G_Q$. The basic idea of our model is to compare two keyword influence graphs to replace the computation of text similarity in the natural language domain. Let us explain this step in detail.

In our model, textual similarity between two texts $T_i$ and $T_j$ is not symmetric. In the following, we represent the similarity measure by $Sim(T_i|T_j)$, which is read as the textual similarity of text $T_j$ to $T_i$. This $Sim(T_i|T_j)$ is one measure of the degree of similarity between two keyword influence graphs $G_i(V, E, f_i)$ ($G_j(f_j)$ for short) and $G_j(V, E, f_j)$ ($G_i(f_i)$ for short). Note that $G_j(f_j)$ is constructed from $f_j$ frequent keywords of $T_j$. Following this, we compare the graph similarity between $G_i(f_i)$ and $G_j(f_j)$. It should be noted that the vertex set $V(G_i(f_i))$ cannot be identical to $V(G_j(f_j))$, since we disregard isolated vertices. This may happen if a word $w_a$ appearing in $T_j$ does not appear in $T_i$, where vertex $w_a$ in $G_j(f_j)$ is a singleton.

Numerous graph similarity measures have been developed for different applications [25], [26]. We use a primitive graph object comparison model. Note that $G_i(f_i)$ and $G_j(f_j)$ are all vertex-labeled graphs. This enables us to compare them more efficiently than vertex-unlabeled graphs, the comparison of which is a well-known NP-hard, graph isomorphism problem. The simplest method of graph comparison is to check the number of vertices that are common between two graphs. In this case, the primitive counting object is the complete graph (clique) $K_1$ (which is a single vertex). We extend $Sim(T_i|T_j)$ to $Sim_k(T_i|T_j)$. Let $G(T_i; K_r)$ denote the set of all $K_r$-vertex-induced subgraphs in $T_i$.

$Sim_p(T_i|T_j)$ is defined as follows:

$$Sim_p(T_i|T_j) = \frac{|G(T_i; K_p) \cap G(T_j; K_p)|}{|G(T_j; K_p)|}$$

Hence, if $T_j$ is nearly part of $T_i$, the value of $Sim_p(T_i|T_j)$ approaches 1. Otherwise, $T_j$ is not included in $T_i$, and $Sim_p(T_i|T_j)$ will approach zero. It is easy to see that if $T_i$ is similar to $T_j$ in terms of linguistic structure, then $Sim_p(T_i|T_j) \approx Sim_p(T_j|T_i) \approx 1$. If a text forms any part of another text, then $Sim_p(T_i|T_j) \approx 1$ and $Sim_p(T_j|T_i) \approx 0$.

1. If two documents share no common parts, we expect $Sim_p(T_i|T_j) \approx Sim_p(T_j|T_i) \approx 0$. Thus, our asymmetric graph similarity measure is quite useful not only for searching similar documents, but also for finding similar parts of texts.

It is worth noting that our comparison model can be a generalized text comparison procedure. The simplest text comparison method uses a frequent keyword list (i.e., the text content without regarding the relationship among words in textual space), which is covered by our model $G(T_i; K_1)$. If two texts are compared according to a co-occurrence of words model, this is reduced to the edge comparison model of $G(T_i; K_2)$. It is easy to determine if $T_i$ is plagiarized from $T_j$, for then, for all $r = 1, 2, 3, 4, \ldots, n$, $G(T_i; K_r) \approx G(T_j; K_r)$.

Let us compute $Sim_p(T_i|T_a)$ using the two graphs in the figure. When $p = 1$, that is, $\{[a, b, c, d, e, f, g, h, i, j, k] \} / 21 = 1/12 = 0.083$. In a similar case, we can obtain $Sim_3(T_i|T_a) = 0.5$. Also, $Sim_4(T_i|T_a) = 0.6$.

F. Visualization of keyword network for document abstraction

There are several issues in text summarization and visualization. A typical example of word-meaning space is to construct a word network (wordnet for short) by consulting a dictionary and a thesaurus. The abstract and a short introduction are used to give a brief summary, or clues, of the chapters to follow. Our keyword graph can be a very useful visualization scheme to depict the entire structure of a target document. When a reader is looking at the network structure of important keywords, he/she may be better able to understand the keywords. For our experiment, we construct a keyword graph from a report, spanning more than 200 pages, which discusses the future economic state of Korea.

We gave this keyword graph to 10 undergraduate students (group A) before they were allowed to read the report. We
showed the abstract to another group of 10 students (group B), but did not show them the keyword graph. We then asked members of both groups to read the report and outline important issues in it. We found that students in group A, the group which was given access to the keywords, exhibited significantly better understanding of the report than students in group B. In case two keywords $k_a$ and $k_b$ are linked by an edge, $(k_a,k_b)$, encountering one keyword $k_a(k_b)$ encourages the reader to expect the other $k_b(k_a)$.

V. EXPERIMENTAL RESULTS

Our experiment consists of two parts. The first part shows the usefulness of keyword influence graph visualization. We selected the Holy Bible as the “long” input text to show the expressiveness of the keyword influence graph, shown in Figure 7. We selected 70 keywords from the Bible according to their frequency and semantic importance. The number of parallel edges represent the overlapping strength between any two keywords in an influence interval. Figure 7 shows the keyword graph from a part of the Old Testament. The word corresponding to each vertex number is as follows: 1 = ’Yahweh’, 2 = ’Lord’, 3 = ’Israel’, 4 = ’land’, 5 = ’Jesus’, 6 = ’mind’, 7 = ’Sin’, 8 = ’David’, …, 24 = ’God’, …, 36 = ’fire’, 37 = ’blood’ and 38 = ’glorification’).

In the second experiment, we searched a text similar to a given one for a query text. For this purpose, we downloaded about 2,000 speeches from the White House website delivered from 2010 to 2013. We chose 10 typical speeches from these ($W_1$ $W_{2000}$), as described in 1. The speeches were given by US President Barack Obama, First Lady Michelle Obama and other officials. In the preprocessing step, we applied the Stanford tokenizer [27] to split documents into statements and tokens. All words were transformed to lowercase after tokenization, and we ignored words containing special characters and numbers. When constructing word influence graphs, the edge weight between two nodes is the number of times the two corresponding intervals overlap. All visualization work was carried out using the web-based graph visualizer Gephi [28].

Table I summarizes speech texts used in the experiment and the characteristics of their corresponding keyword influence graph discussed in the previous section. We list the speaker and the topic information, the number of total words in $T_i$, and the number of keywords in $T_i$ by $|f_i|$. Table-II lists a 10-sample keyword influence graph $G_i$, 1 ≤ $i$ ≤ 10, the number of vertices (keywords) $|V|$, and the number of edges. “Max degree” denotes the number of adjacent edges of maximal degree. “Max Label” represents the vertex name (word) of the node with of the maximal degree, which can be regarded as the main keyword of the speech as it is more strongly related to other keywords. Table-III shows the number of primitive components (clique of the complete graph $K_r$) in $G_i$. It shows that the number of $K_r$ cliques decreases drastically with increasing $r$, which is expected.

We then compute $Sim_r(T_i|T_1)$, which shows the similarity between $T_i$ and the base text $T_1$. If $Sim_r(T_i|T_1)$ ≈ 1, we believe that $T_i$ contains parts quite similar to $T_1$, that $T_1$ is similarly contained within $T_i$. Figure 8 shows $G_1(f_1), G_2(f_1), G_3(f_1)$, where $f_i$ is defined the frequent keyword list of text $T_1$.

Table-?? shows the top word ranking among the 2,000 White House texts given a query text $T_1$ document. For each query text $T_i$, we used the $K_r$ clique, where $r = 1, 2, 3$. The most similar document obtained using $K_1$ will be the same as that using keyword composition or the text fingerprint approach. The cosine measure can be considered as another version of $K_1$ composition comparison. $K_2$ enables us to find the most similar document using the word co-occurrence model, since a frequent co-occurred word pair will be reconstructed by an edge of the $G_i(T_j)$ graph. The higher the value of $r$, the more similar the text in terms of semantic structure.
TABLE I. A BRIEF SUMMARY OF SOME EXPERIMENTAL RESULTS WITH TWO DIFFERENT KEYWORD SETS.

| Text | Speaker | Main Topic | words | \(|T_i|\) |
|------|---------|------------|-------|--------|
| \(T_1\) Obama | Remarks by the President in Town Hall Meeting in Nashua | 11,199 | 73 |
| \(T_2\) Obama et al. | Discussion on Insurance Reform at Bipartisan Meeting on Health Care Reform | 14,426 | 65 |
| \(T_3\) Obama | Remarks by the President and Governor Romney in the Third Presidential Debate Lynn University | 17,727 | 65 |
| \(T_4\) Obama | Remarks by the President in Town Hall Meeting In Henderson, Nevada | 10,590 | 56 |
| \(T_5\) Obama | Remarks by the President on the Economy at Knox College, Galesburg, IL | 6,107 | 55 |
| \(T_6\) Obama | Remarks by the President and First Lady in Town Hall with Students in Mumbai, India | 8,518 | 30 |
| \(T_7\) Obama | Press Conference by the President | 9,604 | 56 |
| \(T_8\) Obama et al. | Remarks by the President and Press Briefing by Press Secretary Robert Gibbs | 9,736 | 56 |
| \(T_9\) Obama and Robert Gibbs | Remarks by the President in Town Hall with LinkedIn | 10,644 | 36 |

TABLE II. TOP WORD MATCHING RANK USING QUERY \(T_1\) DOCUMENT.

<table>
<thead>
<tr>
<th>Query</th>
<th>Text</th>
<th>Match</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>(T_1) Obama</td>
<td>3</td>
<td>work america energy people economy</td>
<td>W_582</td>
<td>W_1101</td>
<td>W_1102</td>
<td>W_1103</td>
<td>W_1104</td>
</tr>
<tr>
<td>(T_2) Obama et al.</td>
<td>2</td>
<td>work america job people year</td>
<td>W_1101</td>
<td>W_1102</td>
<td>W_1103</td>
<td>W_1104</td>
<td>W_1105</td>
</tr>
<tr>
<td>(T_3) Obama</td>
<td>2</td>
<td>work america energy people family</td>
<td>W_1101</td>
<td>W_1102</td>
<td>W_1103</td>
<td>W_1104</td>
<td>W_1105</td>
</tr>
<tr>
<td>(T_4) Obama</td>
<td>2</td>
<td>work america energy people bill</td>
<td>W_1101</td>
<td>W_1102</td>
<td>W_1103</td>
<td>W_1104</td>
<td>W_1105</td>
</tr>
<tr>
<td>(T_5) Obama</td>
<td>2</td>
<td>work america job right kids</td>
<td>W_1101</td>
<td>W_1102</td>
<td>W_1103</td>
<td>W_1104</td>
<td>W_1105</td>
</tr>
</tbody>
</table>

TABLE III. TEXT RANKING BY SIMILARITY USING QUERY \(T_1\) DOCUMENT.

<table>
<thead>
<tr>
<th>Query</th>
<th>(Sim_r)</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>(T_1) Obama</td>
<td>r = 1</td>
<td>W_20</td>
<td>W_21</td>
<td>W_22</td>
<td>W_23</td>
<td>W_24</td>
</tr>
<tr>
<td>(T_2) Obama et al.</td>
<td>r = 2</td>
<td>W_20</td>
<td>W_21</td>
<td>W_22</td>
<td>W_23</td>
<td>W_24</td>
</tr>
<tr>
<td>(T_3) Obama</td>
<td>r = 3</td>
<td>W_20</td>
<td>W_21</td>
<td>W_22</td>
<td>W_23</td>
<td>W_24</td>
</tr>
<tr>
<td>(T_4) Obama</td>
<td>r = 4</td>
<td>W_20</td>
<td>W_21</td>
<td>W_22</td>
<td>W_23</td>
<td>W_24</td>
</tr>
<tr>
<td>(T_5) Obama</td>
<td>r = 5</td>
<td>W_20</td>
<td>W_21</td>
<td>W_22</td>
<td>W_23</td>
<td>W_24</td>
</tr>
</tbody>
</table>

We tested for \(r = 3\) in this paper. The experiment clearly shows that the most similar texts among 2,000 example texts are highly dependent on the value of \(r\).

Figure 8 shows the most similar graph for a query graph \(G_i(T_i|T_1)\). Sub-figure (a), (b) and (c) show the maximal clique keyword influence graph. The most similar text to the query text \(T_1\) in our experiment is shown in Figure 9.

VI. CONCLUSIONS

Text analysis is an important task in information processing. While many practically useful methods have been proposed, it is still challenging to develop a representation of texts that reflects their syntactic and semantic structure. In this paper, we proposed a graph-based modeling method founded on the human cognitive writing model. The main contributions of this paper are summarized as follows:

- We presented a novel graph-based text analysis framework that reflects the human interaction model in writing. Our proposed method computes the intervals that indicate kept-in-memory the regions of influence of each word. The keyword influence graph is constructed by connecting pairs of nodes whose intervals overlap. This keyword influence graph is useful to compare the semantic similarity of target texts, which are hidden underneath the linear adjacency relations of each word. Our influence interval is a good model to trace the meaningful intervals of words.

- Our experiment showed that our new similarity measure \(Sim_r(T_j|T_i)\) is a generalized text comparison model for an appropriate \(r\) value. The text fingerprinting and word co-occurrence models are easily obtained by setting \(r = 1\) and \(r = 2\), respectively.

- The asymmetric measure \(Sim_r(T_j|T_i)\) is not only useful in determining if \(T_i\) is similar to \(T_j\), but also in ascertaining if a text forms parts of another.

- The proposed model enables us to apply the text itself as a query unit. This is the main contribution of our paper. Query-by-example liberates us from having to choose keywords from texts, which is the main disadvantage of previous text-retrieval models.

- We conducted a series of experiments to show the effectiveness of our proposed word influence graph model, with more than 2,000 speeches obtained from the White House speech repository. The experiments show our result is so competitive text clustering and retrieval. Even though a keyword influence graph is not computationally cheap in comparison with a simple fingerprint vector, our approach is relatively more effective and efficient in finding "truly similar" texts to a query text in terms of meaning and linguistic structure. Our approach can be extended regardless of the characteristics of the natural language at hand.

Our plan for future research is to further develop our proposed method: figuring out graph characteristics both theoretically and empirically, and conducting intensive experiments on its applications. Some optimization problems in constructing and using graphs under specific constraints will be considered as well, e.g., indexing the (static) text to construct a word influence graph for a dynamic keyword set more efficiently. We will apply our model to Chinese, Japanese and a few European languages (such as German and Spanish).

REFERENCES


Fig. 9. The most similar text to query text $T_1$, $G_{W_{20}}(W_{20}|T_1)$ has the most similar K1 values in the graph, corresponding to 95.89%. $G_{W_{28}}(W_{28}|T_1)$ has the most similar K2 values in the graph, corresponding to 30.67%. $G_{W_{34}}(W_{34}|T_1)$ has the most similar K3 values in the graph, corresponding to 38.89%.

\[
\text{MaxClique} = \{ \text{america, job, people, work, right} \}
\]

Fig. 8. Example of graphs constructed with \( f_1 \). If it is not the original text, smaller keywords appear in the graph.


