Clustering of Web Documents Using Graph Representations

Adam Schenker, Horst Bunke, Mark Last and Abraham Kandel

Summary. In this paper we describe a clustering method that allows the use of graph-based representations of data instead of traditional vector-based representations. Using this new method we conduct content-based clustering of two web document collections. Clustering of web documents is performed to organize the documents with little or no human intervention. Benefits of clustering include easier browsing and improved retrieval speed. In order to measure the performance of our graph-matching approach, we compare it to the popular vector-based k-means method. We perform experiments using different graph distance measures as well as various document representations that utilize graphs. The results with the k-means clustering algorithm show that the graph-based approach can outperform traditional vector-based methods.

Key words: Graph distance, Graph representations, k-Means

1 Introduction

Clustering is the separation of a collection of objects into groups, called clusters, such that objects within the same cluster are similar to each other, yet dissimilar to the objects in other clusters. Clustering is an unsupervised method, meaning no labeled training examples are provided. Many different clustering algorithms have been proposed, such as k-means, fuzzy c-means, hierarchical agglomerative, and graph partitioning [1].

Clustering of natural language documents is an important research area for two major reasons. First, clustering a document collection into categories enables it to be more easily browsed and used. Second, clustering can improve the performance of search and retrieval on a document collection. Hierarchical clustering methods [1], for example, are used often for this purpose. When representing documents for clustering the vector model is typically used [2]. In this model, each meaningful term that can appear in a document becomes a feature (dimension).

The vector model is simple and allows the use of traditional clustering methods that deal with numerical feature vectors. However, it discards information such as the order in which the terms appear, where in the document the terms appear, how close the terms are to each other, and so forth. By keeping this kind of structural information we could possibly improve the performance of the clustering. The problem is that traditional clustering methods are often restricted to working on purely numeric feature vectors due to the need to compute distances between objects or to find some representative element of a cluster of objects, both of which are easily accomplished with numerical feature vectors. Thus either the original data needs to be converted to a vector of numeric values by discarding possibly useful structural information (that we do when using the vector model to represent documents) or we need to develop new, customized algorithms for a specific representation.

In order to overcome this problem, we have introduced an extension of classical clustering methods that allows us to work with graphs as fundamental data structures instead of being limited to vectors of numeric values [3, 4]. Our approach has two main benefits:

1. It allows us to keep the inherent structure of the original documents by modeling each document as a graph.
2. We can apply straightforward extensions to use existing clustering algorithms rather than needing to create new algorithms from scratch. In this paper we will address comparison of different graph similarity measures and document representations in the context of document clustering. We will use a graph-based k-means clustering algorithm to cluster two web document collections. We will use the cosine and Jaccard similarity measures [2] with the vector model representation as a baseline for comparison.

Recently, several papers have appeared in the literature that deal with graph representations of documents. Liang and Doermann represented the physical layout of document images as graphs [5]. In their *layout graphs* nodes represent elements on the page of a document, such as columns of text or headings, while edges indicate how these elements appear together on the page (i.e., spatial relationships). This method is based on the formatting and appearance of the documents when rendered, not the textual content (words) of a document as in our approach. Lopresti and Wilfong compared web documents using a graph representation that primarily utilizes HTML parse information, in addition to hyperlink and content order information [6]. In their approach they use *graph probing*, which extracts numerical feature information from the graphs, such as node degrees or edge label frequencies, rather than comparing the graphs themselves. In contrast, our representation uses graphs created solely from the content, and we use the graphs themselves rather than a set of extracted features. The *subject graphs* of Tomita et al. [7] are constructed using weights calculated from term occurrence frequencies; our method does not calculate any weights, and most of our models do not use any frequency information.

Other graph-based approaches to text or web representation that are well known in the literature include Sowa's *conceptual graphs* [8] and *directed acyclic word graphs* (DAWGs) [9]. Conceptual graphs provide for powerful knowledge representation capabilities, but are not widely used for web documents because "they are based on deep analysis, and so require well maintained dictionaries and an excessive amount of time to operate" [8]. DAWGs are used for compact representation and recognition of individual words in a text rather than representation of entire documents.

Clustering with graphs is well established in the literature. However, the paradigm in those methods is to treat the entire clustering problem as a graph: nodes represent the items to be clustered and weights on edges connecting two nodes indicate the distance (dissimilarity) between the objects the nodes represent. The usual procedure is to create a minimal spanning tree of the graph and then remove the remaining edges with the largest weight in the MST until the number of desired clusters (connected components) is achieved [10]. After applying the algorithm the edges indicate which objects belong to which clusters. In our method, by contrast, each object is represented by a graph (not a node), and we perform standard clustering methods on these graphs.

Lately there has been some progress with performing clustering directly on graph-based data. For example, an extension of self-organizing maps (SOMs) which allows the procedure to work with graphs has been proposed [11]; graph edit distance and weighted mean of a pair of graphs were introduced to deal with graph-based data under the SOM algorithm. Clustering of shock trees using tree edit distance has also been considered [12]. Both of these methods have in common that they use graph (or tree) edit distance for their graph distance measures. One drawback of this approach is that the edit cost functions must be specified for each application. Sanfeliu et al. have investigated clustering of attributed graphs using their own "function-described graphs" as cluster representatives [13]. However, their method is rather complicated and much more involved than our straightforward extension of a classical, simple clustering algorithm.

The remainder of this paper is organized as follows. In Sect. 2 we give the formal notations relating to graphs that will be used throughout the paper. We describe the graph-based extension of the k-means algorithm and the various graph distance measures in Sect. 3. The details of the different graph representations we utilize during clustering are provided in Sect. 4. In Sect. 5 we explain our experimental procedures and present the results. Finally, some concluding remarks are given in Sect. 6.

2 Formal Notation

In this section we will give the formal mathematical notation which pertains to graphs and their role in performing clustering with the k-means algorithm. *Graphs* are a mathematical formalism for dealing with structured entities and systems. In basic terms a graph consists of *vertices* (or *nodes*), which correspond to some objects or components. Graphs also contain *edges*, which indicate relationships between the vertices. The first definition we have is that of the graph itself. Each object (web document, in the context of this paper) in the data set we are clustering will be represented by such a graph:

Definition 1. *A graph G is defined by a four-tuple (quadruple): $G = (V, E, \alpha, \beta)$, where V is a set of vertices (also called nodes), $E \subseteq V \times V$ is a set of edges* *connecting the vertices, $\alpha : V \rightarrow \Sigma_V$ is a function labeling the vertices, and $\beta : E \rightarrow \Sigma_E$ is a function labeling the edges (Σ_V and Σ_E being the sets of labels that can appear on the nodes and edges, respectively).*

The graphs we will use in this paper are directed graphs with node and edge labels. The next definition we have is that of a *subgraph*. One graph is a subgraph of another graph if it exists as part of the larger graph:

Definition 2. *A graph $G_1 = (V_1, E_1, \alpha_1, \beta_1)$ is a subgraph of a graph $G_2 = (V_2, E_2, \alpha_2, \beta_2)$, denoted $G_1 \subseteq G_2$, if $V_1 \subseteq V_2$, $E_1 \subseteq E_2 \cap (V_1 \times V_1)$, $\alpha_1(x) = \alpha_2(x)\ \forall x \in V_1$, and $\beta_1((x,y)) = \beta_2((x,y))\ \forall (x,y) \in E_1$. Conversely, graph G_2 is also called a supergraph of G_1.*

Next we have the important concept of the maximum common subgraph (mcs) for short, which is the largest subgraph a pair of graphs have in common:

Definition 3. *A graph G is a maximum common subgraph (mcs) of graphs G_1 and G_2, denoted $mcs(G_1, G_2)$, if: (1) $G \subseteq G_1$ (2) $G \subseteq G_2$ and (3) there is no other subgraph G' ($G' \subseteq G_1$, $G' \subseteq G_2$) such that $|G'| > |G|$.*

In the above definition, $|G|$ is intended to convey the "size" of the graph G; often it is taken to be $|V|$, i.e., the number of vertices in the graph. In most of the graph representations used in this paper we will define the size of a graph to be $|G| = |V| + |E|$, i.e., the sum of the number of nodes and edges in the graph. Complementary to Definition 3, we also have the concept of MCS:

Definition 4. *A graph G is a minimum common supergraph [14] (MCS) of graphs G_1 and G_2, denoted $MCS(G_1, G_2)$, if: (1) $G_1 \subseteq G$ (2) $G_2 \subseteq G$ and (3) there is no other supergraph G' ($G_1 \subseteq G'$, $G_2 \subseteq G'$) such that $|G'| < |G|$.*

Now that we have our formal notation, we are in a position to proceed to describing the k-means algorithm extended to cluster graphs instead of vectors.

3 Clustering with Graphs

3.1 Basic Clustering Algorithm

The k-means clustering algorithm is a simple and straightforward method for clustering data [15]. The basic algorithm is given in Fig. 1. Usually during clustering we represent each object, which consists of m numeric values, as a vector in the space $\Re^m$. When representing documents in this manner, each value is associated with a specific term (word) that may appear on a document, and the set of possible terms is shared across all documents; this is called the vector-space model of information retrieval. The values may be binary, indicating the presence or absence of the corresponding term. The values may also be nonnegative integers, which represent the number of times a term appears on a document (i.e., term frequency). Non-negative

```
Inputs:   the set of n data items and a parameter, k, defining the number of clusters to create
Outputs:  the centroids of the clusters and for each data item the cluster (an integer in [1,k]) it
          belongs to

Step 1.   Assign each data item randomly to a cluster (from 1 to k).
Step 2.   Using the initial assignment, determine the centroids of each cluster.
Step 3.   Given the new centroids, assign each data item to be in the cluster of its closest centroid.
Step 4.   Re-compute the centroids as in Step 2. Repeat Steps 3 and 4 until the centroids do not change.
```

Fig. 1. The k-means clustering algorithm

real numbers can also be used, in this case indicating the importance or weight of each term. These values are derived through a method such as the popular inverse document frequency model $(tf \cdot idf)$ [2], which reduces the importance of terms that appear on many documents. Regardless of the method used, each series of values represents a document and corresponds to a point (i.e., vector) in a Euclidean feature space. This model is often used when applying data mining techniques to documents, as there is a strong mathematical foundation for performing distance measure and centroid calculations using vectors. However, this method of document representation does not capture important structural information, such as the order and proximity of term occurrence, or the location of term occurrence within the document. It is also common to restrict the number of dimensions by selecting some small set of discriminating or important terms, as the number of possible terms that can occur across a collection of documents can be quite large.

When representing data by vectors, the distances between two objects can be computed using the Euclidean distance in m dimensions:

$$dist_{EUCL}(x,y) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2} \tag{1}$$

where x_i and y_i are the ith components of vectors $x = [x_1, x_2, \ldots, x_m]$ and $y = [y_1, y_2, \ldots, y_m]$, respectively. However, for applications in text and document clustering, the cosine similarity measure [2] is often used due to its length invariance property. We can convert this to a distance measure by the following:

$$dist_{COS}(x,y) = 1 - \frac{x \bullet y}{\|x\| \cdot \|y\|} \tag{2}$$

Here $\bullet$ indicates the dot product operation and $\|\ldots\|$ indicates the magnitude (length) of a vector. Another popular distance measure for determining document similarity is the extended Jaccard similarity [2], which is converted to a distance measure as follows:

$$dist_{JAC}(x,y) = 1 - \frac{\sum_{i=1}^{m} x_i y_i}{\sum_{i=1}^{m} x_i^2 + \sum_{i=1}^{m} y_i^2 - \sum_{i=1}^{m} x_i y_i} \tag{3}$$

We have determined that if methods of computing distance between graphs and constructing a representative of a set of graphs are available it is possible to extend many clustering and classification methods to work directly on graphs. First, any distance calculations between objects to be clustered, which are represented by graphs and not vectors, is accomplished with a graph-theoretical distance measure as we will discuss in Sect. 3.2. Second, since it is necessary to compute the distance between objects and cluster centers, it follows that the cluster centers (representatives) must also be graphs. Therefore, we compute the representative of a cluster as the median graph of the set of graphs in that cluster (as we will describe in Sect. 3.3).

3.2 Graph Distance Measures

As we mentioned above, we need a graph-theoretical distance measure in order to use graphs for clustering. We have implemented several distance measures and will compare their clustering performance. For brevity we will refer to the distance measures below as MCS, WGU, and MMCS.

The first distance measure MCS is a well-known graph distance measure based on the mcs [16]:

$$d_{MCS}(G_1, G_2) = 1 - \frac{|mcs(G_1, G_2)|}{\max(|G_1|, |G_2|)} \tag{4}$$

where G_1 and G_2 are the graphs to compare, $mcs(G_1, G_2)$ is their maximum common subgraph, $|\dots|$ is the size of a graph, and $\max(\dots)$ is the usual maximum operation. Here we define the size of a graph to be the sum of the number of nodes and edges in the graph. The concept behind this distance measure is that as the size of the maximum common subgraph of a pair of graphs becomes larger, the more similar the two graphs are (i.e., they have more in common). The larger the maximum common subgraph, the smaller $d_{MCS}(G_1, G_2)$ becomes, indicating more similarity and less distance. If the two graphs are in fact identical, their maximum common subgraph is the same as the graphs themselves and thus the size of all three graphs is equal: $|G_1| = |G_2| = |mcs(G_1, G_2)|$. This leads to the distance, $d_{MCS}(G_1, G_2)$, becoming 0. Conversely, if no maximum common subgraph exists, then $|mcs(G_1, G_2)| = 0$ and $d_{MCS}(G_1, G_2) = 1$. This distance measure has been shown to be a metric [16], and produces a value in $[0, 1]$.

A second distance measure WGU which has been proposed by Wallis et al. [17] is:

$$d_{WGU}(G_1, G_2) = 1 - \frac{|mcs(G_1, G_2)|}{|G_1| + |G_2| - |mcs(G_1, G_2)|} \tag{5}$$

This distance measure behaves similarly to MCS. If the maximum common subgraph does not exist (i.e., $|mcs(G_1, G_2)| = 0$), then $d_{WGU}(G_1, G_2) = 1$. If the maximum common subgraph is identical to the original graphs, $|G_1| = |G_2| = |mcs(G_1, G_2)|$, then the graphs G_1 and G_2 are identical and thus $d_{WGU}(G_1, G_2) = 0$. The denominator used in this method is based on the idea of "graph union." It represents the size of the union of the two graphs in the set theoretic sense; specifically adding the size of each graph ($|G_1| + |G_2|$) then subtracting the size of their intersection ($|mcs(G_1, G_2)|$) leads to the size of the union (the reader may easily verify this using a Venn diagram). The motivation for doing this is to allow for changes in the smaller graph to exert some influence over the distance measure, which does not happen with MCS [17]. This measure was also demonstrated to be a metric, and creates distance values in $[0, 1]$.

The third distance measure MMCS, proposed by Fernández and Valiente, is based on both the maximum common subgraph and the MCS [14]:

$$d_{MMCS}(G_1, G_2) = |MCS(G_1, G_2)| - |mcs(G_1, G_2)| \tag{6}$$

where $MCS(G_1, G_2)$ is the minimum common supergraph of graphs G_1 and G_2. The concept that drives this distance measure is that the maximum common subgraph provides a "lower bound" on the similarity of two graphs, while the MCS is an "upper bound." If two graphs are identical, then both their mcs and MCS are the same as the original graphs and $|G_1| = |G_2| = |MCS(G_1, G_2)| = |mcs(G_1, G_2)|$, which leads to $d_{MMCS}(G_1, G_2) = 0$. As the graphs become more dissimilar, the size of the maximum common subgraph decreases, while the size of the MCS increases. This in turn leads to increasing values of $d_{MMCS}(G_1, G_2)$. For two graphs with no maximum common subgraph, the distance will become $|MCS(G_1, G_2)| = |G_1| + |G_2|$. MMCS has also been shown to be a metric [14], but it does not produce values normalized to the interval $[0, 1]$, unlike the previously described distance measures. Note that if it holds that $|MCS(G_1, G_2)| = |G_1| + |G_2| - |mcs(G_1, G_2)|\ \forall G_1, G_2$, we can compute $d_{MMCS}(G_1, G_2)$ as $|G_1| + |G_2| - 2|mcs(G_1, G_2)|$. This is much less computationally intensive than computing the MCS.

We will describe our graph representation of documents in detail in Sect. 4. However, we wish to mention here an interesting feature our graph representation has on the time complexity of determining the distance using (4–6). For general graphs the computation of the mcs is NP-Complete. Methods for computing the mcs are presented in [18, 19]. However, for the graph representations of web documents presented in this paper, the computation of the maximum common subgraph is $O(n^2)$, with n being the number of nodes, due to the existence of unique node labels in the graph representations (i.e., we need only examine the intersection of the nodes, since each node has a unique label) [20]. Thus the maximum common subgraph, G_{mcs}, of a pair of graphs with unique node labels, G_1 and G_2, can be created by the following procedure:

1. Find the nodes V_{mcs} by determining the subset of node labels that the original graphs have in common with each other and create a node for each common label.
2. Find the edges E_{mcs} by examining all pairs of nodes from step 1 and introduce edges that connect pairs of nodes in both of the original graphs with identical edge labels.

Note that the calculation of the MCS can be reduced to the mcs problem [21]. Therefore the computation of the MCS can also be performed in $O(n^2)$ time.

3.3 Median of a Set of Graphs

The second ingredient required to apply clustering to graphs is that of a graph-theoretic cluster representative of a set of graphs. For this we have used the concept of the *median graph* [22], which is the graph which has the minimum average distance to all graphs in the cluster:

$$G = \arg \min_{\forall s \in S} \left(\frac{1}{n} \sum_{i=1}^{n} dist(s, G_i) \right) \tag{7}$$

Here $S = \{G_1, G_2, \ldots, G_n\}$ is a set of n graphs for which we want to compute the median (and thus $|S| = n$) and G is the median graph. The median is defined to be a graph in set S. Thus the median of a set of graphs is the graph from that set which has the minimum average distance to all the other graphs in the set. The distance $dist(\ldots)$ is computed using one of (4–6) above. There also exists the concepts of the generalized median and weighted mean [22], where we do not require that G be a member of S, but we will not consider them here because they are quite expensive to compute. In the case where the median is not unique (i.e., there is more than one graph that has the same minimum average distance) we select one of those graphs at random as the representative for the k-means algorithm. This variation of the k-means algorithm, where we use a median instead of a mean as cluster representatives, is also known as k-medoids [23].

4 Graph Representations of Web Documents

In this section we describe methods for representing web documents using graphs instead of the traditional vector representations. All representations are based on the adjacency of terms in a web document. These representations are named: *standard*, *simple*, *n-distance*, *n-simple distance*, *raw frequency* and *normalized frequency*.

Under the *standard* method each unique term (word) appearing in the document, except for *stop words* such as “the,” “of,” and “and” which convey little information, becomes a node in the graph representing that document. Each node is labeled with the term it represents. Note that we create only a single node for each word even if a word appears more than once in the text. Second, if word a immediately precedes word b somewhere in a “section” s of the document, then there is a directed edge from the node corresponding to term a to the node corresponding to term b with an edge label s. We take into account certain punctuation (such as periods) and do not create an edge when these are present between two words. Sections we have defined for web documents are: *title*, which contains the text related to the document’s title and any provided keywords (meta-data); *link*, which is text that appears in hyperlinks on the document; and *text*, which comprises any of the readable text in the document (this includes link text but not title and keyword text). Next we remove the most infrequently occurring words on each document, leaving at most m nodes per graph (m being a user provided parameter). This is similar to the dimensionality reduction process for vector representations [2]. Finally we perform a simple stemming method and conflate terms to the most frequently occurring form by relabeling nodes and updating edges as needed. An example of this type of graph representation is given in Fig. 2. The ovals indicate nodes and their corresponding term labels. The edges are labeled according to title, link, or text. The document represented by the example has the title "YAHOO NEWS," a link whose text reads "MORE NEWS," and text containing "REUTERS NEWS SERVICE REPORTS." If a pair of terms appears together in more than one section, we create an edge for each section with the appropriate section label. Note there is no restriction on the form of the graph and that cycles are allowed. Also, disconnected components may occur in the graphs, which is not a problem with our approach. While this method of document representation appears superficially similar to the bigram, trigram, or N-gram methods, those are statistically oriented approaches based on word occurrence probability models [24]. The methods presented here, with the exception of the frequency representations described below, do not require or use the computation of term probability relationships.

The second type of graph representation we will look at is what we call the *simple* representation. It is basically the same as the standard representation, except that we look at only the visible text on the page (no title or meta-data is examined) and we do not label the edges between nodes. Thus we ignore the information about the "section" where the two respective words appear together. An example of this type of representation is given in Fig. 3.

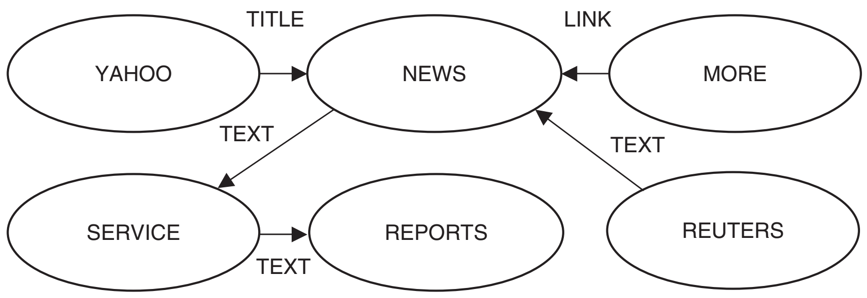

Fig. 2. Example of a *standard* graph representation of a document

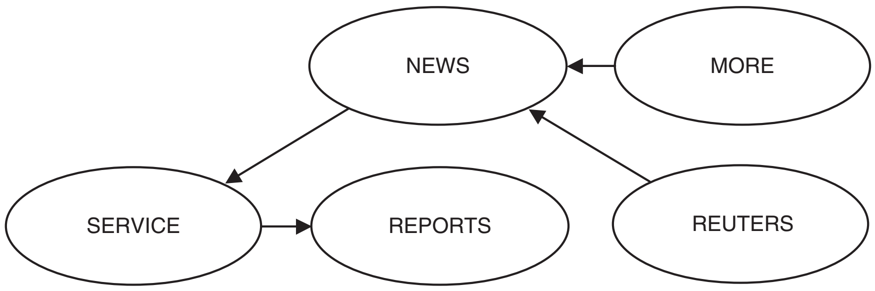

Fig. 3. Example of a *simple* graph representation of a document

The third type of representation is called the n-*distance* representation. Under this model, there is a user-provided parameter, n. Instead of considering only terms immediately following a given term in a web document, we look up to n terms ahead and connect the succeeding terms with an edge that is labeled with the distance between them (unless the words are separated by certain punctuation marks); here "distance" is related to the number of other terms which appear between the two terms in question. For example, if we had the following sequence of text on a web page, "AAA BBB CCC DDD," then we would have an edge from term AAA to term BBB labeled with a 1, an edge from term AAA to term CCC labeled 2, and so on. The complete graph for this example is shown in Fig. 4. The mcs for this representation is derived in the same manner as described previously, where we require the edge labels to be an exact match in both graphs.

Similar to n-distance, we also have the fourth graph representation, n-*simple distance*. This is identical to n-distance, but the edges are not labeled, which means we only know that the "distance" between two connected terms is not more than n.

The fifth graph representation is what we call the *raw frequency* representation. This is similar to the simple representation (adjacent words, no section-related information) but each node and edge is labeled with an additional frequency measure. For nodes this indicates how many times the associated term appeared in the web document; for edges, this indicates the number of times the two connected terms appeared adjacent to each other in the specified order. The raw frequency representation uses the total number of term occurrences (on the nodes) and co-occurrences (edges).

A problem with this representation is that large differences in document size could lead to skewed comparisons, similar to the problem encountered when using Euclidean distance with vector representations of documents. Under the *normalized frequency* representation, instead of associating each node with the total number of times the corresponding term appears in the document, a normalized value in $[0, 1]$ is assigned by dividing each node frequency value by the maximum node frequency value that occurs in the graph; a similar procedure is performed for the edges. Thus each node and edge has a value in $[0, 1]$ associated with it, which indicates the normalized frequency of the term (for nodes) or co-occurrence of terms (for edges).

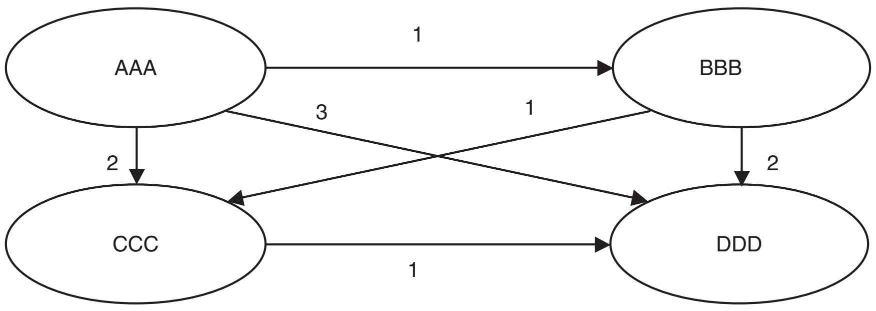

Fig. 4. Example of a n-*distance* graph representation of a document

For the raw frequency and normalized frequency representations the graph size is defined as the total of the node frequencies added to the total of the edge frequencies, rather than the previous definition of $|G| = |V| + |E|$. We need this modification to reflect the frequency information in the graph size. As an example, consider two raw frequency graphs each with a node "A"; however, term "A" appears twice in one document and 300 in the other. This difference in frequency information is not captured under the previous definition. Further, when we compute the mcs for these representations we take the minimum frequency element (either node or edge) as the value for the mcs. To continue the above example, node "A" in the mcs would have a frequency of 2, which is $\min(2, 300)$.

5 Experiments and Results

5.1 Web Document Data Sets

In order to evaluate the performance of the graph-based k-means algorithm as compared with the traditional vector methods, we performed experiments on three different collections of web documents, called the *F-series*, the *J-series*, and the *K-series* [25]; the data sets are available under these names at `ftp://ftp.cs.umn.edu/dept/users/boley/PDDPdata/`. These data sets were selected because of two major reasons. First, all of the original HTML documents are available, which is necessary if we are to represent the documents as graphs; many other document collections only provide a preprocessed vector representation, which is unsuitable for use with our method. Second, ground truth assignments are provided for each data set, and there are multiple classes representing easily understandable groupings that relate to the content of the documents. Some web document collections are not labeled or are presented with some other task in mind than content-related clustering (e.g., building a predictive model based on user preferences).

The F-series originally contained 98 documents belonging to one or more of 17 subcategories of four major category areas: *manufacturing*, *labor*, *business and finance*, and *electronic communication and networking*. Because there are multiple subcategory classifications for many of these documents, we have reduced the categories to just the four major categories mentioned above in order to simplify the problem. There were five documents that had conflicting classifications (i.e., they were classified to belong to two or more of the four major categories) which we removed, leaving 93 total documents. The J-series contains 185 documents and ten classes: *affirmative action*, *business capital*, *information systems*, *electronic commerce*, *intellectual property*, *employee rights*, *materials processing*, *personnel management*, *manufacturing systems*, and *industrial partnership*. We have not modified this data set. The K-series consists of 2,340 documents and 20 categories: *business*, *health*, *politics*, *sports*, *technology*, *entertainment*, *art*, *cable*, *culture*, *film*, *industry*, *media*, *multimedia*, *music*, *online*, *people*, *review*, *stage*, *television*, and *variety*. The last 14 categories are subcategories related to entertainment, while the entertainment category refers to entertainment in general. These were originally news pages hosted at Yahoo (`http://www.yahoo.com`). Experiments on this data set are presented in [26].

For the vector-model representation experiments there were already several term-document matrices available for our experiments at the same location where we obtained the document collections. We selected the matrices with the smallest number of dimensions. For the F-series documents there are 332 dimensions (terms) used, while the J-series has 474 dimensions; the K-series used 1,458 dimensions. We performed some preliminary experiments and observed that other term-weighting schemes (i.e., $tf \cdot idf$, see [2]) improved the accuracy of the vector-model representation for these data sets either only very slightly or in many cases not at all. Thus we have left the data in its original format.

5.2 Clustering Performance Measures

We use the following three clustering performance measures to evaluate the performance of each clustering. The first two indices measure the matching of obtained clusters to the "ground truth" clusters, while the third index measures the quality of clustering in general.

The first index is the *Rand index* [27]. To compute the Rand index, we perform a comparison of all pairs of objects in the data set after clustering. If both objects in a pair are in the same cluster in both the ground truth clustering and the clustering we wish to measure, this counts as an "agreement." If both objects in the pair are in different clusters in both the ground truth clustering and the clustering we wish to investigate, this is also an agreement. Otherwise, this is a "disagreement." The Rand index is computed as:

$$R_I = \frac{A}{A+D} \tag{8}$$

where A is the number of agreements and D is the number of disagreements, as described above. Thus the Rand index is a measure of how closely the clustering created by some procedure matches ground truth. It produces a value in the interval $[0, 1]$, with 1 representing a clustering that perfectly matches ground truth.

The second performance measure we use is *mutual information* [26, 28], which is defined as:

$$\Lambda^M = \frac{1}{n}\sum_{l=1}^{k}\sum_{h=1}^{g} n_l^{(h)} log_{k\cdot g}\left(\frac{n_l^{(h)} \cdot n}{\sum_{i=1}^{k} n_i^{(h)} \sum_{i=1}^{g} n_l^{(i)}}\right) \tag{9}$$

where n is the number of objects, k is the number of clusters produced by our clustering algorithm, g is the actual number of ground truth clusters, and $n_i^{(j)}$ is the number of items in cluster i (in the created clustering) associated with cluster j (in the ground truth clustering). Note that k and g may not necessarily be equal, which would indicate we are attempting to create more (or fewer) clusters than exist in the ground truth clustering. However, for the experiments described in this paper we will create an identical number of clusters as is present in ground truth. Mutual information represents the overall degree of agreement between the clustering created by some method and the categorization provided by the ground truth clustering with a preference for clusters that have high purity (i.e., are homogeneous with respect to the objects clustered, as given by the clusters they belong to in ground truth). Higher numbers indicate clusters that are homogeneous (i.e., created clusters which contain objects mostly belonging to a single ground truth cluster). Lower numbers indicate less similarity between the clustering that was created and ground truth; a value of zero signifies no statistical correlation between the two clusterings (i.e., they are independent).

The third performance measure we use is the *Dunn index* [29], which is defined as:

$$D_I = \frac{d_{min}}{d_{max}} \tag{10}$$

where d_{min} is the minimum distance between any two objects in different clusters and d_{max} is the maximum distance between any two items in the same cluster. The numerator captures the worst-case amount of separation between clusters, while the denominator captures the worst-case compactness of the clusters. Thus the Dunn index is an amalgam of the overall worst-case compactness and separation of a clustering, with higher values being better. It does not, however, measure clustering accuracy compared to ground truth as the other two methods do. Rather it is based on the basic underlying assumption of any clustering technique: items in the same cluster should be similar (i.e., have small distance, thus creating compact clusters) and items in separate clusters should be dissimilar (i.e., have large distance, thus creating clusters that are well separated from each other).

5.3 Results

In Tables 1–3 we show the clustering performance for the F-series, J-series, and K-series when using different graph distance measures (Sect. 3.2). The performance of the traditional vector-based approach using distances based on cosine and Jaccard similarity is also given for comparison. Because of the random initialization of the k-means algorithm, each number indicates the average performance taken over ten experiments. We used a maximum of 50 nodes per graph (i.e., $m = 50$, see Sect. 4) for the F and J data sets, while we used 70 nodes per graph for K, due to the higher number of classes and documents. The standard representation was used for the distance measure comparison experiments. The value of k used in the experiments matches the number of clusters present in the ground truth clustering for each data set; thus $k = 4$ for the F-series, $k = 10$ for the J-series, and $k = 20$ for the K-series.

We see that the graph-based methods that use normalized distance measures (MCS and WGU) generally performed similarly to or better than vector-based methods using cosine or Jaccard. MMCS, which is not normalized to the interval $[0, 1]$, performed poorly for all data sets. To see why this occurs, we have provided the following example. Let $|G_1| = 10$, $|G_2| = 10$, $|mcs(G_1, G_2)| = 0$, $|MCS(G_1, G_2)| = 20$, $|G_3| = 20$, $|G_4| = 20$, $|mcs(G_3, G_4)| = 5$, and $|MCS(G_1, G_2)| = 35$. Clearly

Table 1. Distance measure comparison for the F-series data set using the standard representation and 50 nodes per graph maximum

distance measure	Rand index	mutual information	Dunn index
Cosine (vector-based)	0.6788	0.1101	0.4168
Jaccard (vector-based)	0.6899	0.1020	0.6188
MCS	0.7748	0.2138	0.7202
WGU	0.7434	0.1744	0.7967
MMCS	0.6594	0.1120	0.3132

Table 2. Distance measure comparison for the J-series data set using the standard representation and 50 nodes per graph maximum

distance measure	Rand index	mutual information	Dunn index
Cosine (vector-based)	0.8648	0.2205	0.3146
Jaccard (vector-based)	0.8717	0.2316	0.5703
MCS	0.8618	0.2240	0.6476
WGU	0.8757	0.2598	0.7691
MMCS	0.1809	0.0273	0.1381

Table 3. Distance measure comparison for the K-series data set using the standard representation and 70 nodes per graph maximum

distance measure	Rand index	mutual information	Dunn index
Cosine (vector-based)	0.8537	0.2266	0.0348
Jaccard (vector-based)	0.8998	0.2441	0.0730
MCS	0.8957	0.1174	0.0284
WGU	0.8377	0.1019	0.0385
MMCS	0.1692	0.0127	0.0649

graphs G_3 and G_4 are more similar to each other than graphs G_1 and G_2 since G_1 and G_2 have no common subgraph whereas G_3 and G_4 do. However, the distances computed for these graphs are $d_{MCS}(G_1, G_2) = 1.0$, $d_{MCS}(G_3, G_4) = 0.75$, $d_{MMCS}(G_1, G_2) = 20$, and $d_{MMCS}(G_3, G_4) = 30$. So we have the case that the unnormalized distance is actually greater for the pair of graphs that are more similar. This is both counter-intuitive and the opposite of what happens in the cases of the normalized distance measures. Thus this phenomenon leads to the poor clustering performance for MMCS.

In Tables 4–6 we show the clustering performance for the F-series, J-series, and K-series for the different graph representations presented in Sect. 4. For these experiments we use the MCS distance measure (4). For the representations n-distance and

Table 4. Representation comparison for the F-series data set using MCS distance and 50 nodes per graph maximum

representation	Rand index	mutual information	Dunn index
Cosine (vector-based)	0.6788	0.1101	0.4168
Jaccard (vector-based)	0.6899	0.1020	0.6188
Standard	0.7748	0.2138	0.7202
Simple	0.6823	0.1314	0.7364
2-distance	0.6924	0.1275	0.7985
5-distance	0.6731	0.1044	0.8319
2-simple distance	0.7051	0.1414	0.7874
5-simple distance	0.7209	0.1615	0.8211
Raw frequency	0.7070	0.1374	0.7525
Normalized frequency	0.7242	0.1525	0.7077

Table 5. Representation comparison for the J-series data set using MCS distance and 50 nodes per graph maximum

representation	Rand index	mutual information	Dunn index
Cosine (vector-based)	0.8648	0.2205	0.3146
Jaccard (vector-based)	0.8717	0.2316	0.5703
Standard	0.8618	0.2240	0.6476
Simple	0.8562	0.2078	0.5444
2-distance	0.8674	0.2365	0.6531
5-distance	0.8598	0.2183	0.7374
2-simple distance	0.8655	0.2285	0.7056
5-simple distance	0.8571	0.2132	0.6874
Raw frequency	0.8650	0.2141	0.6453
Normalized frequency	0.8812	0.2734	0.6119

n-simple distance, we use values of $n = 2$ and $n = 5$ (i.e., 2-distance, 2-simple distance, 5-distance, and 5-simple distance) in these experiments.

For the F-series, standard was the best performing representation, achieving the best value for Rand index and mutual information, while for the Dunn index, 5-distance was the best representation. For the J-series, normalized frequency was the best for Rand index and mutual information, with 5-distance again being best for the Dunn index. It is not a surprising result that Rand and mutual information should perform similarly to each other and differently than Dunn, as both Rand and mutual information are based on comparison with ground truth while Dunn is a measure of compactness and separation of the clusters with no regard to "accuracy."

For the K-series, the best performing graph representation was standard. However, the graph-based method in this case did not outperform the Jaccard distance-based vector approach. The K-series is a highly homogeneous data set; all the pages

Table 6. Representation comparison for the K-series data set using MCS distance and 70 nodes per graph maximum

representation	Rand index	mutual information	Dunn index
Cosine (vector-based)	0.8537	0.2266	0.0348
Jaccard (vector-based)	0.8998	0.2441	0.0730
Standard	0.8957	0.1174	0.0284
Simple	0.8870	0.0972	0.0274
2-distance	0.8753	0.0832	0.0229
5-distance	0.8813	0.1013	0.0206
2-simple distance	0.8813	0.0947	0.0218
5-simple distance	0.8663	0.0773	0.0234
Raw frequency	0.8770	0.0957	0.0335
Normalized frequency	0.8707	0.0992	0.0283

Table 7. Statistical analysis of experimental results

data set	performance measure	confidence $(1-P)$	significant?
F-series	Rand	0.9998	yes (better)
F-series	MI	1.0000	yes (better)
J-series	Rand	0.9255	no (same)
J-series	MI	0.4767	no (same)
K-series	Rand	0.3597	no (same)
K-series	MI	1.0000	yes (worse)

have a similar format and some of the same terms appear on every document. To improve the performance of the graph method in this case, we should look at either removing the common terms (nodes) from all graphs (which is often done with the vector model and can also be applied to our approach), or greatly increase the size of the graphs to capture more terms. In our experiments, Rand increases to 0.9053 and mutual information to 0.1618 for the standard representation and MCS distance when using 200 nodes maximum per graph.

In Table 7 we give a statistical analysis of some of the experimental results. Six comparisons are listed in the table, which represent comparing the Jaccard and graph methods for Rand index and mutual information for all three data sets. The graph experiments represented in Table 7 use the standard graph representation, MCS distance, and either 50 nodes per graph (F and J) or 70 nodes per graph (K). The *Confidence* column in the table represents the probability that the means of the results for the vector and graph methods are statistically different, as determined by a two-tailed t-test. Values higher than 0.95 are considered significant, as shown in the last column of the table; we also show whether the graph method was considered better, the same, or worse than the vector method.

6 Conclusions

In this paper we have examined the problem of clustering data which is represented by graphs instead of simpler feature vectors. To perform the clustering we have developed a graph-based version of the k-means clustering algorithm, substituting a suitable graph-theoretical distance measure in the place of the usual vector-related distance and median graphs in place of centroids.

The application we presented here was clustering of web documents. We implemented six different methods of representing web documents by graphs and three different graph distance measures. Our experiments compared the clustering performance of the various proposed methods with the usual vector model approach using cosine and Jaccard-based distance measures. Experimental results showed that the graph-based methods can outperform the traditional vector methods in terms of clustering performance under three different clustering performance measures. We saw that graph distance measures that were not normalized performed poorly, while those that were normalized to the interval $[0, 1]$ yielded good results. The standard representation produced the best results for one data set in terms of comparison with ground truth, while normalized frequency was better for another.

For future work we intend to extend our graph-based method to other classification and clustering methods, such as hierarchical agglomerative clustering and distance weighted k-nearest neighbors. We also wish to look for the optimal graph size and associated terms to represent each specific document. Further, we only examined using two values of n for the n-distance and n-simple distance representations in this paper. Finding the optimal value of n is another subject of ongoing research. Given the good results for the normalized frequency representation for one of the document collections, we will explore similar representations that incorporate more explicit term weighting components (i.e., a model similar to $tf \cdot idf$ but for graphs). However, such an extension is not immediately obvious, since we must deal with adjusting the weights of edges as well as terms (nodes). Finally, we can look at incorporating specific domain knowledge in the distance measure definitions.

Acknowledgement. This work was supported in part by the National Institute for Systems Test and Productivity at the University of South Florida under the US Space and Naval Warfare Systems Command Contract No. N00039–02–C–3244.

References

1. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999
2. G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison–Wesley, Reading, 1989
3. A. Schenker, M. Last, H. Bunke, and A. Kandel. Comparison of distance measures for graph-based clustering of documents. In *Proceedings of the Fourth IAPR-TC15 Workshop on Graph-based Representations*, pp. 202–213, 2003

4. A. Schenker, M. Last, H. Bunke, and A. Kandel. Graph representations for web document clustering. In *Proceedings of the First Iberian Conference on Pattern Recognition and Image Analysis*, pp. 935–942, 2003
5. J. Liang and D. Doermann. Logical labeling of document images using layout graph matching with adaptive learning. In D. Lopresti, J. Hu, and R. Kashi, editors, *Document Analysis Systems V*, Volume 2423 of *Lecture Notes in Computer Science*, pp. 224–235. Springer, Berlin Heidelberg New York, 2002
6. D. Lopresti and G. Wilfong. A fast technique for comparing graph representations with applications to performance evaluation. *International Journal on Document Analysis and Recognition*, 6(4):219–229, 2004
7. J. Tomita, H. Nakawatase, and M. Ishii. Graph-based text database for knowledge discovery. In *World Wide Web Conference Series*, pp. 454–455, 2004
8. J. F. Sowa. Conceptual graphs for a database interface. *IBM Journal of Research and Development*, 20(4):336–357, 1976
9. M. Crochemore and R. Vérin. Direct construction of compact directed acyclic word graphs. In A. Apostolico and J. Hein, editors, *CPM97*, Volume 1264 of *Lecture Notes in Computer Science*, pp. 116–129. Springer, Berlin Heidelberg New York, 1997
10. C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt structures. *IEEE Transactions on Computers*, C-20:68–86, 1971
11. S. Günter and H. Bunke. Self-organizing map for clustering in the graph domain. *Pattern Recognition Letters*, 23:405–417, 2002
12. B. Luo, A. Robles-Kelly, A. Torsello, R. C. Wilson, and E. R. Hancock. Clustering shock trees. In *Proceedings of the Third IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pp. 217–228, 2001
13. A. Sanfeliu, F. Serratosa, and R. Alquézar. Clustering of attributed graphs and unsupervised synthesis of function-described graphs. In *Proceedings of the 15th International Conference on Pattern Recognition*, Volume 2, pp. 1026–1029, 2000
14. M.-L. Fernández and G. Valiente. A graph distance metric combining maximum common subgraph and minimum common supergraph. *Pattern Recognition Letters*, 22:753–758, 2001
15. T. M. Mitchell. *Machine Learning*. McGraw–Hill, Boston, 1997
16. H. Bunke and K. Shearer. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters*, 19:225–259, 1998
17. W. D. Wallis, P. Shoubridge, M. Kraetz, and D. Ray. Graph distances using graph union. *Pattern Recognition Letters*, 22:701–704, 2001
18. G. Levi. A note on the derivation of maximal common subgraphs of two directed or undirected graphs. *Calcolo*, 9:341–354, 1972
19. J. J. McGregor. Backtrack search algorithms and the maximal common subgraph problem. *Software Practice and Experience*, 12:23–34, 1982
20. P. Dickinson, H. Bunke, A. Dadej, and M. Kraetzl. Matching graphs with unique node labels. *Pattern Analysis and Applications*, 7(3):243–254, 2004
21. H. Bunke, X. Jiang, and A. Kandel. On the minimum common supergraph of two graphs. *Computing*, 65:13–25, 2000
22. X. Jiang, A. Muenger, and H. Bunke. On median graphs: properties, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1144–1151, 2001
23. L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York, 1990
24. C.-M. Tan, Y.-F. Wang, and C.-D. Lee. The use of bigrams to enhance text categorization. *Information Processing and Management*, 38:529–546, 2002

25. D. L. Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2(4):325–344, 1998
26. A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *AAAI-2000: Workshop of Artificial Intelligence for Web Search*, pp. 58–64, 2000
27. W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, 1971
28. T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991
29. J. Dunn. Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4:95–104, 1974