ISSN No. 0976-5697

Volume 5, No. 1, Jan-Feb 2014

International Journal of Advanced Research in Computer Science

RESEARCH ARTICLE

Available Online at www.ijarcs.info

An Overview of VSM-Based Text Clustering Approaches

Francis Musembi Kwale
Lecturer, Department of Mathematics & Computer Science,
University of Eldoret, Eldoret, Kenya.
fmkwale@yahoo.com, fmkwale@gmail.com

Abstract: Huge digital data is typical nowadays. Consequently, text clustering has become a crucial text mining technique nowadays. Organizing search information into groups (or clusters) not only makes the search results more meaningful, but makes the system more efficient since the user views only the relevant information and ignores the rest. An example is clustering web search engine result into meaningful results.
Many text clustering approaches and their corresponding algorithms exist, but none has been found to be sufficient. There is also insufficient understanding of the algorithms as well as lack of agreed formal classification of the algorithms. There is thus, need for an in-depth study of the various algorithms.
In this paper, we describe the Vector Space Model (VSM) method of text representation. We also give an overview of the text clustering approaches that apply the VSM .These include distance based approach, feature extraction approach, density-based approach, grid-based approach, and neural networks approach. We describe the characteristics of each accompanied by a representative algorithm.
The paper thus informs researchers of text mining concerning the current state of affairs of text clustering algorithm.

Keywords: clusters, text clustering, text mining, vector space model.

1. INTRODUCTION

Data mining is a computer science area that can be defined as extraction of useful information from large (structured) data sets (specifically databases), e.g. detecting customers' preferences from a sales database. However, much of the information available for analysis is in text form. Thus, text mining is more applicable.

Text mining is an extension of data mining dealing only with text data i.e. unstructured data (e.g. word processing and web documents). It's thus the process of extracting useful patterns or knowledge from unstructured text.

Text clustering (TC) is a text mining technique used to group text documents into groups (or clusters) based on similarity of content. The clustering is so as to make documents more understandable, easier to search the relevant information, easier to process, etc.

2. THE VECTOR SPACE MODEL

2.1: Factors Distinguishing TC Approaches

There are different clustering algorithms, each using a particular approach. The approaches are distinguished by the following three factors.

- How a document is represented. The mostly used text representation method is VSM.
- The measure of similarity between documents.
- The logic/formulae of putting similar documents together (i.e. the clustering logic).

2.2: The Vector Space Model (VSM)

The VSM is the mostly used document representation method. Consequently, similarities between documents and the clustering logic are done on the VSM representation.

The VSM document representation

The VSM-based algorithms represent the text documents being clustered using a term-document matrix, i.e. the VSM (which is structured). The unstructured text documents are converted into the VSM structured form, since structured data is easier to cluster. Similarity measures and the clustering logic are then applied on the VSM.

Here, **n** documents containing **m** terms (words) are represented using a matrix of **m** rows and **n** columns (i.e. the **term-document matrix**). The rows (vectors) represent the terms (or words). The columns (vectors) represent the documents. The **ij**th entry in the matrix stores the number of occurrences of **i**th term in the **j**th document.

In mathematical space-based view, a term will be an axis of the space, while a document is a data point in the space.

Example

Assume the three documents;

- D1: Eating fruit improves health.
- D2: Give your infant fruit regularly, for the infant to have good health.
- D3: Regular exercise improves your health.

(Underlined are identifiable key terms – to form basis of clustering)

We form a term-dictionary as T1: fruit, T2: health, T3: infant, T4: exercise. We then form our term-document matrix (or the VSM) to be

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Note that;

- The first row is the vector (1, 1, 0) for first term (fruit), showing that the term occurs in the first and the second document, but not in the third.
- Similarly, vector (1, 1, 0, 0) represents the first document (that contains the first and the second terms, but not the third and the fourth terms).
- And entry $\mathbf{A_{42}}$ is 0, i.e. the fourth term (exercise) is not present in the second document.

In space-based view, this is a 4-dimensional view (because of 4 terms). A document is a point on the view. So the three documents will be the data points D1: (1, 1, 0, 0), D2: (1, 1, 2, 0), and D3: (0, 1, 0, 1).

An obvious limitation of VSM is the possible huge dimensions. For example, 1,000 documents with 500,000 terms will result to a matrix of 500,000 rows and 1,000 columns in the computer's memory! Thus, dimension reduction is a major need for text clustering algorithms.

Documents similarity measure and the clustering logic

The document similarity measurement and the actual logic of deciding which document goes to which cluster based on the similarity measurements varies from one algorithm to another. However, a common way to measure the similarity between two text documents is the length of the straight line between the corresponding two data points in the VSM, i.e. the **Euclidean measure.** For example, the distance between D1 and D2 in the immediate above example is gotten as $(1-1)^2+(1-1)^2+(0-2)^2+(0-0)^2)^{1/2} = 2$

3. THE VSM-BASED APPROACHES

3.1: Distance Based Approaches

These algorithms use the Euclidean distance between data points in VSM as the similarity between two documents. The nearest points represent similar documents. The logic differs from one algorithm to another.

A representative distance-based algorithm is the K Means algorithm, though there are variants.

The K Means algorithm

The K-Means algorithm is among the few most popular clustering algorithms, and was developed by J. MacQueen in 1967. It's a distance-based algorithm. It's a flat-type (or partitioning) clustering algorithm, meaning that the produced clusters are one-level (i.e. un-hierarchical). The above approach of K Means is;

- **Document representation**: K Means converts the text documents into a VSM (structured) form.
- **Definition of similarity measure**: It measures similarity between two documents as the Euclidean measure or the cosine measure of their two points in the VSM.
- **The clustering logic (or algorithm)**:
  1. Choose the number of clusters, k.
  2. Randomly generate k clusters and determine the cluster centers (centroids), where a cluster's centroid is the mean of all points in the cluster.
  3. Repeat the following until no object moves (i.e. no object changes its cluster)
     (i) Determine the Euclidean distance of each object to all centroids.
     (ii) Assign each point to the nearest centroid.
     (iii) Re-compute the new cluster centroids.

Thus, in each loop of step 3 above, the algorithm aims at minimizing the following function for **k** clusters and **n** data points.

$$J=\sum_{j=1}^{j=k}\sum_{i=1}^{i=n} \|x_i-c_j\|^2$$

where $\|x_i-c_j\|$ is a choosen distance measure (e.g. Euclidean measure) between data point x_i from cluster c_j.

Distance-based algorithms have an advantage of ease and simplicity to understand and implement. They use straight forward Euclidean calculations and elementary data storage using matrices. They are also efficient enough in memory requirements since the system only needs to store the data points and their distances from centroids using matrices.

However, some of their limitations are that the user must specify the initial number of clusters before clustering, and that they don't include dimension reduction. They are also sensitive to outliers (usual data sizes).

3.2: Feature Extraction Approach

These are algorithm used to reduce the dimensions of the term-document matrix by obtaining (selecting or extracting) only some important features from the original matrix and clustering them.

The feature extraction actually does feature transformation, i.e. defines new features to represent the original features (or data set). Here, the correlations among the words in the data set are leveraged in order to create features, which correspond to the concepts in the data. [1] says that 'The idea behind these methods is to extract the key/hidden features from the original feature-document matrix and to apply clustering algorithms only on these key features'.

A representative example of feature extraction algorithm is the LSA.

One advantage of these methods is that the dimension of the documents is reduced drastically, thus improving efficiency (but their inefficiency is as a result of their computations). Secondly, such algorithms are able to discover deep relations between terms and documents, thus ensuring high clustering accuracy.

However, they have expensive computations which lower their efficiency a lot. Also, the generated features are hard to interpret.

The Latent Semantic Analysis (LSA)

LSA projects an original vector space or term-document matrix into a small factor space [2]. It is a feature extraction clustering technique that applies Singular Value Decomposition (SVD) to reduce the dimension of the term-document matrix. SVD applies a mathematical rule that specifies that an **m** by **n** rectangular matrix say **A**, can be broken down into the product of three matrices say **U**, **S** and $\mathbf{V^T}$ i.e. $\mathbf{A_{mn}=U_{mm}S_{mn}V^T_{nn}}$, whereby

- **U** is a **m** by **m** orthogonal matrix whose columns are orthonormal eigenvectors of $\mathbf{AA^T}$,
- **S** is a **m** by **n** diagonal matrix containing the square roots of eigen values from **U** and **V** in descending order, and

- **V** is an **n** by **n** orthogonal matrix whose columns are orthonormal eigenvectors of $\mathbf{A}^T\mathbf{A}$.

In LSA, we decompose a weighted (e.g. with term frequencies) term-document matrix **A** into the three matrices **U**, **S**, and $\mathbf{V}^T$. Here, **U** is the document matrix while $\mathbf{V}^T$ is the term matrix. In other words, matrix **U** describes the original column entities as vectors of derived orthogonal factor values, while matrix $\mathbf{V}^T$ describes the original row entities as vectors of derived orthogonal factor values. Another characteristic of vectors of **U** and $\mathbf{V}^T$ is that they have mixed signs (i.e. some values are positive and others negative). Also, the set of rows (representing terms) usually repeals some **semantic** relations among the terms (representing synonyms). LSA is able to reveal deep/hidden (i.e. **latent**) relations among data items, thus the name Latent Semantic Analysis. And the set of columns (representing documents) repeals some clusters. Also, topics are represented by the factors of the matrix, and consequently, the relationships between documents are represented by the orthogonal characteristics of the factors. Thus, words in a factor have little relations with words in other factors, but words in a factor have high relations with words in that factor.

We then reconstruct the original matrix **A** by multiplying **U**, **S** and $\mathbf{V}^T$ but using only the vectors of **U** and $\mathbf{V}^T$ that are associated with higher eigen values of **S** (we ignore others). We can specify a particular threshold of these eigen values to massively reduce **A**'s dimension. In this way, **A**'s rank is reduced and so the dimension reduction need of text mining clustering is accomplished. SVD thus gives a reduced representation of the original text data.

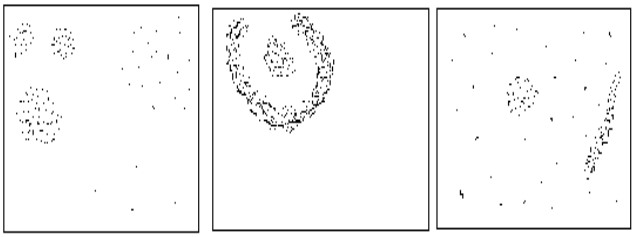
3.3: Density-based Approach

Rather than measuring distances between data points (as in distance-based algorithms), density-based algorithms find clusters by differentiating regions in terms of the relative density (or compactness/concentration/number of objects per unit area) of points in them. Thus, regions adjacent to a cluster contain data points of either less concentration or higher concentration. It was stated that

> An area is said to be of high density if it contains a large number of data points per area unit; otherwise, it is of low density. Under this understanding of space, a cluster is an area of density exceeding the required threshold value or greater than the density of the enclosing space. The areas that do not constitute clusters are considered to be noise [3, p. 20].

Some clusters are illustrated below.

Illustration 1: Some clusters – easy to identify using density approach



5 clusters **2 clusters** **3clusters**

Density-based algorithms are partitioning type of algorithms. A representative example is DBSCAN.

An advantage of density-based algorithms is that they can discover clusters of irregular shapes because a cluster is represented as a connected dense component that can grow in any direction that density leads. According to [4], clusters which are formed based on the density are easy to understand and the clustering is not limited to the shapes of clusters. Consequently and secondly, density-based algorithms are less sensitive to outliers/noise (because of the arbitrary shapes discovering ability). For example, the clusters in illustration 1 above (some with arbitrary shapes) will be a big problem to other algorithms like the distance-based algorithms, but not to density-based algorithms. 'Density-based algorithms are capable of discovering clusters of arbitrary shapes. Also this provides a natural protection against outliers' [5, p. 4]. Also, you don't specify the number of clusters unlike in K Means family of algorithms.

A limitation of density-based algorithms is that there are initial parameters that need to be set. And it's hard to specify the most appropriate setting.

The DBSCAN (Density Based Spatial Clustering of Applications with Noise) algorithm

This algorithm was presented by Ester, Kriegel, Sander, and Xu in 1996. According [6], the DBSCAN algorithm takes in two inputs, i.e. the radius of the cluster (*Eps*) and minimum points required inside the cluster (*Minpts*). Consequently, two types of points of a cluster are the **core** (a point which is inside the cluster, and so should lie within *Eps* and *Minpts*), **border** (a point on the border of the cluster, but within the neighborhood of the core), and **noise** (a point which is neither a core point nor a border point). And point **q** is **directly density-reachable** from a point *p* if it is not farther away from from *p* than distance *Eps*. And *q* is called **density-reachable** from *p* if there exists a sequence $p_1, p_2, \ldots, p_n$ of points with $p_1=p$ and $p_n=q$ where each $p_{i+1}$is directly density-reachable from p_i.

According to Mumtaz & Duraiswamy, DBSCAN clusters by arbitrary selecting a starting point *p* that has not been visited, and then finding all neighbor points within distance *Eps* of *p*. Then,

- If the number of the neighbors is at least *Minpts*, then a cluster is formed. The point *p* and its neighbors are added to this cluster and then *p* is marked as visited. The algorithm then repeats the evaluation process for all neighbors recursively.
- If the number of neighbors is less than *Minpts*, then *p* is marked as noise.

- If a cluster is fully expanded (i.e. all points within reach are visited) then the algorithm proceeds to iterate through the remaining unvisited points in the dataset.

Experiments have shown that DBSCAN is faster and more precise than many other algorithms. It has a time complexity of O(n log n), which is quite low. 'It holds good for large spatial databases' [4]. DBSCAN can even find a cluster that is entirely surrounded by a different cluster, but not linked to it. This is obviously because of the *Minpts* and *Eps* requirement.

According to [3, p. 18], the limitation of *DBSCAN* is that it can't distinguish dense clusters adhering to sparse clusters. In otherwords, DBSCAN (and other density-based algorithms) doesn't work well with data with different densities. Obviously, this may cause merging of two clusters or considering less dense neighboring clusters as noise. Secondly, the speed of the algorithm depends on the setting of the density parameters (*Eps* and *Minpts*). And it's hard to determine the most appropriate setting. 'For a complex data set, the appropriate parameters are hard to specify' [7].

3.4: Grid-based Approach

Grid-based clustering methods quantize the space of the data points into a finite number of cells to form a grid structure. The cells containing at least the minimum number of points are considered dense. Then the dense cells are connected to form clusters. Thus, there are no distance computations on the data points, and shapes are restricted to the union of the grid cells. The clustering is based not on the data points, but on the value space surrounding the data points.

In short, grid clustering concerns forming clusters with contiguous dense cells. Thus, grid-based approach is related to (or applies) density-based approach.

Grid-based clustering algorithms are usually hierarchical. A representative example of such algorithms is the CLIQUE.

According to [8], the basic steps of a grid algorithm are;

(i) Creating the grid structure, i.e. dividing the data space into a finite number of cells.
(ii) Calculating the cell density for each cell.
(iii) Sorting of the cells according to their densities.
(iv) Identifying cluster centers.
(v) Traversal of neighbor cells.

The advantage of the grid based methods is that they are much efficient. The processing time is much less even with large data sets. According to [3], in high dimensional space, this approach is more efficient than density-based approach. The processing time is fast and dependent only on the number of cells, i.e. they have a complexity of O(number of cells) rather than O(number of documents i.e. *n*).

An obvious limitation of many grid algorithms is that they require some input parameters, e.g. number of cells (intervals) and density threshold (for CLIQUE). Secondly, the efficiency of the algorithms is obviously seriously determined by the size of the cells. This is because a small cell size will clearly lead to unnecessary computation in regions with sparse points. But having a large cell size of course will lead to inaccuracy in regions with dense points. Thirdly, the clustering of cells is done horizontally and vertically, but never diagonally. This is unlike other approaches like distance approach and density approach whereby finding nearby points is not limited to horizontal and vertical directions, but is done in any direction. This obviously might greatly affect the performance.

The CLIQUE (CLustering In QUEst) algorithm

The CLIQUE algorithm was proposed by Agrawal, Gehrke, Gunopulos, and Raghavan in 1998. It applies both density and grid methods. It operates as explained by [9, p. 1] below.

> It makes use of concepts of density and grid based methods. In the first step, CLIQUE partitions the **n** dimensional data space **S** into non overlapping rectangular units (grids). The units are obtained by partitioning every dimension into ξ intervals of equal length. ξ is an input parameter, **selectivity** of a unit is defined as the total data points contained in it. A unit **u** is dense if selectivity (**u**) is greater than γ, where the density threshold γ is another input parameter. A unit in the subspace is the intersection of an interval from each of the **K** attributes. A cluster is a maximal set of connected dense units. Two K-dimensional units **u1**, **u2** are connected if they have a common face. The dense units are then connected to form clusters.

3.5: Neural Networks Approach

In (artificial) neural networks, each cluster as well as the input data is represented by a neuron. The neurons are connected, and each connection has a weight, which is learned adaptively during learning. The typical method used is the SOM.

The Self Organizing Maps (SOM)

The typical type of the neural networks approach is the Self Organizing Maps (SOM). This method was invented by Teuvo Kohonen in 1982, and is also known as Kohonen maps. A SOM is a two-dimensional array of neurons (or a Kohonen map) meant to store neighborhood relations (i.e. relations between neurons). It's a partitioning method.

A SOM is obtained by training the system using unsupervised method to map a high dimensional data space into a lower dimensional (two dimensional) one, which is a grid of neurons. Each neuron has an associated weight vector the same size of input vectors, as well as a position in the map space. Thus, SOMs preserve the topological properties of the input data space, i.e. the mapping preserves the relative distance between the points by the use of a neighborhood function. This means that points that are near to each other in the input data space are mapped to nearby map units. This preservation makes SOMs unique from other artificial neural networks, and this enables automatic clustering of the input data.

According to [10, p. 15], SOM apply neural networks to produce a low dimensional representation of a training sample, whereby a SOM is a single layer feed-forward network, i.e. a network without direct cyclic paths. Neurons are arranged in a low dimensional grid (typically two-dimensional). It was stated that

> Each input data item is mapped into one of the neurons in the map. SOM plays also the role of a clustering

method, so that similar data items – represented as vectors of numerical attribute values – tend to be mapped into nearby neurons [11, p. 9].

A SOM is also known as a SOFM (a Self Organizing Feature Map) because it can be used to detect features inherent to the problem.

Structure of neurons

The SOM is actually an arrangement of **k neurons** (m_1, m_2, m_3, … m_k), whereby a neuron is a vector called the **codebook vector**. For example, $m_1=(m_{11}, m_{12}, m_{13}, \ldots, m_{1n})$. Thus, this arrangement is a two-dimensional array of neurons. The number of neurons (i.e. **k**) is the number of the required clusters. The array's dimension is the same as the input vectors (i.e. k*n). Each neuron represents a cluster and has a weight associated with it. Neurons are connected to each other by a neighborhood relation, and this is the **topology** (or the structure) of the map. The two usual connection types are rectangular (a neuron linked to four neighbors i.e. the left, the right, the top, and the bottom neurons) or hexagonal (a neuron connected to six neighbors) topologies.

Operation modes

Just like other artificial neural networks, SOMs operate in two modes, i.e. **training** (i.e. building the map using some input examples), and **mapping** (i.e. receiving an input document vector from the data space and classifying it - placing it onto the map).

(i) Training

The SOM is trained to automatically change its behavior/functions depending on the input data (i.e. is a self organizing map) .The training is done in unsupervised method using a competitive method, i.e. the neurons compete with one another. Training involves getting sample data and "training" it to the SOM. This is a repetitive process which takes a considerably long time. Initially, the topological relations and the number of neurons are fixed from the beginning. Then during training, a sample vector is obtained randomly from the data, fed to each neuron in the map, and a measure of similarity between the input vector and a neuron computed. The similarity measure is usually the Euclidean measure between two vectors (input vector and the codebook vector). The neuron with the highest measure is selected as the winner neuron. Then the vectors of the winning neuron and its neighbors are updated to make them more nearer to the sample vector. The parameters of this updating operation are the learning rate (i.e. the rate of changing of the SOM) and the neighborhood radius (that determines how far the neighboring neurons should be). Thus, in each training step, the SOM changes (or learns). This process of training is repeated but the learning rate decreases with time.

(ii) **Mapping**: This mode receives an input document vector (from the data space) and classifies it (i.e. places it onto the map). This is by finding the neuron with the closest weight vector to the document vector (which becomes the winning neuron). Then the weights of the winning neuron and those of its neighbors are adjusted appropriately. As a result, the most similar clusters are always the neighboring cluster.

An advantage of neural networks-based algorithms is that they are more accurate than many other algorithms. Secondly, SOM just like other neural networks, are flexible in that they can learn from examples and adapt to situations based on their findings, so can handle unknown situations in future (i.e. text documents of different sizes, dimensions, and applications). This also makes them handle complex data (documents).

One limitation of SOM is that it is sensitive to the initial selection of the weight vector and the different parameters including the learning rate and the neighborhood radius. Secondly, it requires a lot of training of the SOM which is great computational burden, thus inefficient. Also, knowing how much training is enough for a desired output is completely dependent on the trainer itself. Consequently, increasing the number of documents requires retraining of the SOM, and this lowers the system's scalability.

4. REFERENCES

[1] Ning, W, "Textmining and Organization in Large Corpus", MSc Thesis, Technical University of Denmark (DTU), 2005.

[2] Lee, S, Song, J & Kim, Y, "An Empirical Comparison of Four Text Mining Methods", http://iacis.org/jcis/articles/Lee_Song_Kim_51_1.pdf, Journal of Computer Information Systems, Retrieved from, 2010.

[3] Lasek, P, "Efficient Density-Based Clustering", PhD Thesis, Warsaw University of Technology, 2011.

[4] Parimala, M, Lopez, D & Senthilkumar N, "A Survey on Density Based Clustering Algorithms for Mining Large Spatial Databases", International Journal of Advanced Science and Technology, vol. 31. 2011.

[5] Rai, P, "A Survey of Clustering Techniques", International Journal of Computer Applications, vol. 7, no. 12, 2010.

[6] Nagpal, P, Mann, P, "Comparative Study of Density Based Clustering Algorithms", International Journal of Computer Applications, vol. 27, no.11, 2011.

[7] Jiang, D, Pei, J & Zhang, A, "DHC: A Density-based Hierarchical Clustering Method for Time Series Gene Expression Data", Proceedings of Third IEEE Symposium on Bioinformatics and Bioengineering 10-12 March 2003, 2003, pp. 393 – 400, print ISBN: 0-7695-1907-5.

[8] Mann, A & Kaur, N, "Grid Density Based Clustering Algorithm", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), vol. 2, no. 6, 2013.

[9] Ilango & Mohan, V, "A Survey of Grid Based Clustering", International Journal of Engineering Science and Technology, vol. 2, no. 8, 2010.

[10] Geraci, F, "Fast Clustering for Web Information Retrieval", PhD Thesis, Universit. A Degli Studi Di Siena, 2008.

[11] Chifu, E, "Self Organizing Maps in Web Mining and Semantic Web", PhD Thesis, Technical University of Cluj-Napoca. 2010.