Computer Engineering Department
Faculty of Engineering
Deanery of Higher Studies
The Islamic University-Gaza
Palestine

SOMvisua: a framework for clustering and visualization , visual graph-based SOM and GHSOM

Khalid M. Kahloot

Supervisor:

PROF. Mohammad A. Mikki

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering

GAZA, PALESTINE
(DECEMBER, 2013) 1435 H

Dedication

I would like to dedicate this work

To my beloved parents,

To my family , Ayah, Yosra and Moyad

The Researcher

Khalid M. Kahloot

Acknowledgement

I would like to acknowledge this work to Professor Mohammad Mikki, my supervisor, who guided me through research period. He was instructing me to do things in the right ways. Also, Thanks for thesis examiners Professor Aiman Abu Samra and Professor Sami Abu Naser for their valuable comments.

Table of Content

List of Figures

List of Tables

الملخص

SOMvisua: إطار للعنقدة والتصور، تصوير بصري للأساس الرسومي على الخريطة ذاتية التنظيمSOM و على الخريطة ذاتية التنظيم المتنامية هرمياً GHSOM

نحن نقدم إطاراً برمجي لعنقدة البيانات و التصور يسمى SOMvisua . الأطروحة الحالية تسلط الضوء على تمثيل رسومي لإدخال البيانات على الخريطة ذاتية التنظيم (SOM) و على الخريطة ذاتية التنظيم المتنامية هرمياً(GHSOM). وستمثل جمل من مقال كنموذج رسومي بدلا من نموذج متجه الفضاء. و خوارزمية العنقدة SOM و GHSOM ستكون نموذج معرفة حول هذا المقال. أظهرنا كيفية استخدام ميزات تشابه النص بشكل أصيل و صحيح في خوارزميات العنقدة SOM و GHSOM، قمنا بتطوير طريقة للتخفيف من حدة المشكلة تشتت الحلول، فقمنا بخلق حل تصويري بعد مراجعة لفئة خوارزميات تشابه النص. يوفر SOMvisua رسوم متحركة مصورة لتنفيذ ثمانية خوارزميات الرسم البياني الشهيرة مع التحكم في سرعة الحركة. SOMvisua يقدم ستة أنواع من التصور. لتصور قوائم التشابه استخدمنا أساليب معروفة تستطيع أن تأخذ قائمة التشابه كمدخل وفقا لعدة مقياس التشابه مختارة و قابل للتعديل و بالتالي ستظهر الجمل المتشابهة مرتبة في شكل تصوير بصري.

Abstract

We introduce a framework for data clustering and visualization called SOMvisua. The current thesis highlights a graph representation of data input for self-organizing map (SOM) and growing hierarchically self-organizing map (GHSOM). Sentences from an article will be represented as graph model instead of vector space model. The SOM and GHSOM clustering algorithm form knowledge about this article. We showed how to use the text similarity features natively and correctly in SOM and GHSOM clustering algorithms, developed a method to alleviate the hub problem and created a visualizing solution for the reviewed class of text similarity algorithms. SOMvisua provides a visual animation for eight famous graph algorithms execution with animation speed control. SOMvisua presents six types of visualization. For visualization for similarity Lists we used well-known methods that takes a similarity list as input and according to the used similarity measure an adjustable number of most similar sentences are arranged in visual form.

Chapter 1
Introduction

1.1 Overview of the thesis

Text as stipulated from distributors or in webpages is actually based on expert opinion of a large number of people including the views of the author but have different cultural or community which make extract information from very difficult and hard to gathering knowledge. Text search derived from the information on the main aspects of the access to information and presentation. Search in text usually works to find text similarities between of the text or paragraphs among articles of used to get correct results, which works with a wide variety of search text. SOMvisua supports one of the most important tasks in text information retrieval that is extracting similarities and building a structure for data representation alongside with visualization.

In the past few years, the importance of research for search in textual information become very obvious. Many researches in Text Information Retrieval has been carried out recently and their main concern is extraction and analysis of the text describes the different aspects of the view of information. This information is contained mainly of three types of sources: first, reference text file from digital documents. second, metadata provided by the distributors of the web site. Third, information extracted from the Internet. The analysis is based on the descriptive function at a high level on the context, like the matrix structure like in [42] and [53] or graphical representation based on the text of the attributes of the metadata for example like in [60] and [62]. Another example also a generally describe the properties as in Wikipedia.

In chapter 2, the thesis overviews the state of the art about previous work. The former studies tried to compose clustering and visualization algorithms. Chapter 3 illustrates phase one of our framework that is features extraction using Google PageRank algorithm. SOM and GHSOM clustering algorithms are discussed in Chapter 4. Visualization of Input and output Graphs are presented in Chapter 5. Chapter 6 shows the configurations of experiment and reviews the results. Finally, a conclusion is stated to summarize our experiment.

1.2 Challenges

A distance-based similarities in neighborhoods are the represented in most of the proposed visualization schemes. Some of those schemes are U-matrix [94] and its variants [5], [89]. The size and shape of the cells to represent the prototypes are adapted in [85], [57]. Alternatively, some methods use Euclidean distances to update the grid positions of the prototypes for visual inspection as adapted lately in [82] and double SOM [66] and visualization-induced SOM (ViSOM) [55]. Size of receptive fields [57] and smoothed histograms [59] are another methods that uses density-based visualizations. however density-based representations are less helpful compared to the distance-based visualizations unless density representation has a higher resolution than the receptive field size.

A drawback in all varieties of GHSOM is working directly on the observation vector. Observation vector does not take the order of coordinates into account. In some cases, such as processing a time series, it might prove inappropriate to the specific nature of data. The original GHSOM did not define visual distribution on the input space. However, the advantages of introducing a graphical methodologies into SOM models were soon evident. This has led to a wide range of proposals which reflect the importance of graphical approaches to data clustering.

The underlying combinative theories behind them derive from two main ideas: the transforming GHSOM to accept graph as an input and capturing the output topology as an output. The current thesis presents a comprehensive view of the state of the art, with a perspective of the involved theatrical frameworks. SOMvisua examines the most commonly used graph model representation.

Building a text information retrieval application from the scratch takes a very long time. The current available applications based on clustering systems or data features visualizations that may present text sentences to the document currently visualized, but separated apart. this give us a great motivation to build a toolkit easy to extend project. SOMvisua is produces as an object-oriented design concept and implemented in Java, which also facilitates its extensibility. Intended to be as a tool to build and customize applications as a basis can be used to cluster and visualize text to provide a system to operate at full capacity.

1.3 Contribution

SOMvisua provides a File Input and output mechanism to load and save data graphs as ASCII-text files. It generates meta-data files as ASCII-text files and offer the ability to preserve a complete workspace a set of data graphs and meta-data. we present a full implementation of the a graph input Self Organizing Map (SOM) and offering more options for initialization methods those are random, linear, gradient or SLC. To cover all potential needs of farther development, we prepare SOM with two training methods sequential and batch job. We also enhance the implementation of SOM with functions such as calculate SOM, show SOM-grid optionally, with labels from meta-data and load and save SOM-objects from previously saved process. As an extension of SOM we present an implementation of the Growing Hierarchical Self Organizing Map (GHSOM) as well.

SOMvisua provides four major capabilities those are File input output, Data Processing, Text processing and visualization. First, File input output provides capabilities to manipulates data files. to facilitate preserving our project's status, we provides tools to load and save data graphs generated by our project provided as ASCII-text files. another tools to load and save meta-data files provided as ASCII-text files are provided. Tools to load and save the complete workspace are provides as well. Second, Data Processing for constructing data structures to organize data in SOMvisua. moreover, assigning names to the data graphs and meta-data lists with the ability to rename data structures. We provides a data graph decomposition into single lists by sources or by destinations and normalization of data graphs. Third, For text Processing we use Google PageRank algorithm. Extraction of text features Google PageRank algorithm provides a graph based algorithm for text summarization. A GUI for a very simple text viewer. open test files and add them to a list which is used by the text viewer.

Forth, this thesis presents four types of visualization those are Circled Bars, Circled Fans, Probabilistic Network and Sunburst. For visualization of similarity Lists Circled Bars visualization is used. That is a well-known method to that takes a similarity list as input and given a seed sentence, according to the used similarity measure an adjustable number of most similar sentences are arranged in a circle. The sentences are ordered by their similarity to the seed sentence. The Circled Fans visualization is a conceptual extension of the simple Circled Bars. While the Circled Bars only take the nearest neighbors of a given seed sentence (or any other entity) into account, the Circled Fans incorporate similarities in a transitive manner. The Probabilistic Network visualization for similarity graphs uses a graph-based model for illustrating similarities between graphs by using one prototype for each of a number of given classes of sentences. Sunburst visualization for term co-occurrences start with the whole set of the terms with the highest document frequency and are selected and visualized as filled arcs around a centered circle called the root node that represents the entire document collection.

Finally, SOMvisua provides tools to store visualizations in graphics file and export visualizations to Encapsulated PostScript file and export SOM to HTML file. User can adjust some global preferences for the visualization area such as background color, border size and font size for labels.

1.4 Methodology

The method of this thesis is based on graph visualization of the topology of the neutral network resulting documents clustering using a modified version of SOM. This method combines the advantages of graph representation in both inputting and outputting to the clustering process. Well illustrated relationships representing the semantic in documents as a graph is the input and illustrated visual representation of topology of the resulting clustering. User can explore the documents pool either in fast way or in precise way.

SOMvisua is automated hierarchical growing due to the nature of GHSOM and the hieratical advantage of Google PageRank similarity graph. The framework is able to capture different types of clusters by using appropriate similarity criterion. The number of clusters will be determined by prior knowledge on data sets using a recent cluster validity index derived from Google PageRank. This thesis shows that a graph input output clustering can produce better partitioning than other types of clustering.

The main contribution is the mixing of input graph model with the visual output graph model with SOM in an automatic way. The researcher proves that using graph from the beginning to represent documents the performing clustering process will result a robust, fact, and precise clustering. This thesis carries out performance test on the SOMvisua framework on large document dataset. Then compare the performance with that of hierarchal clustering with automated topology based SOM and GHSOM clustering to prove the superiority.

Exploration of sentences Pool On figure 1. our basic concept is shown. Set of sentences is the sentence space wanted to be explored. SOMvisua will generate the similarity Graph of this document set via Google PageRank algorithm. Similarity Graph shows sentences on vertices and edges represented with a graph data structure. SOMvisua provides multiple methods for visualization similarity graph. SOMvisua provides the capability of executing eight well-known graph algorithms with visualization. The similarity graph is considered as an input to clustering algorithm. SOMvisua passes similarity graph to SOM clustering algorithm to generate the map. Similar sentences will be placed close to each other on the map. Beside this topological structure, sentences are grouped into clusters. Sentences that are very similar and best described with the same unit map are contained in the same cluster.

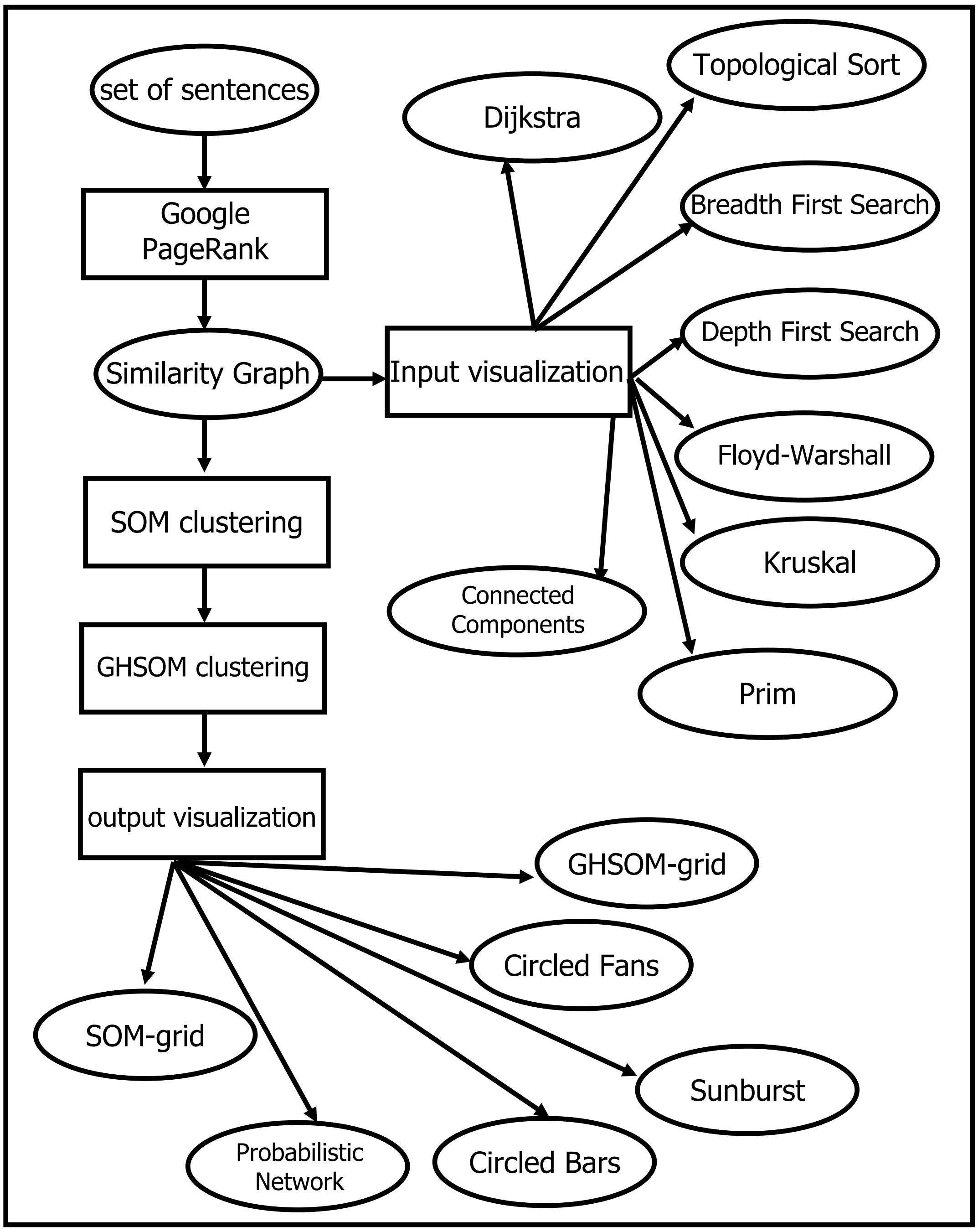


Figure 1: basic concept of SOMvisua

The important in this concept is decoupled visualization using SOM and GHSOM clustering. In the first phase, feature extraction phase, most important features of the sentences were found. In the second phase, clustering phase, sentences are mapped to 2d map using SOM map and then categorized hierarchal using GHSOM. Finally , SOMvisua provides six visualization tools to represent clustering.

Chapter 2

Related Work

2.1 Graph-based Feature Extraction

Much early work on document graph representations for text classification was directed at web documents. *Geibel et al.* in [33] demonstrated that it is possible to classify web documents using document structure alone and demonstrate that a much more powerful approach is to combine structure with linguistic and semantic information. For example *Schenker* [39] proposed a number of methods to represent web documents as graphs so as to include the structural information of the web documents. The typical approach is to conduct classification using some similarity-based algorithm. However, approaches that operate using graph similarity measures are computationally expensive. For example computing the "maximum common subgraph" between two graphs is a NP hard problem [100].

Hybrid representations have been introduced to resolve the computational overhead associated with pure graph representations, see for example [37]. Such hybrid representations are reported to have better performance than pure graph based methods. However, the computational resources required to process these hybrid model are still very high due to: (i) the extremely high number of nodes and edges, low number of edge labels and high repetition of structural node labels, encountered; and (ii) the consequent exponential complexity of the search space. The use of graphs for representing text has a very long history in Natural Language Processing (NLP). However the work in NLP has focused on language understanding techniques such as Part Of Speech (POS) tagging, rather than text classification.

Previous work [35] and [41] has looked at the collocation of terms and their frequencies as graphs, rather than the linguistic structure of the sentence. One other study [43] has represented linguistic information as well as word order in a graph for text classification, however the work was limited to very small texts of between 8 to 13 tokens such as the titles of works. As such, adopting the usage of linguistic information, structure and semantics in a graph for text classification at a full text scale.

In order to achieve this scale of processing, the use of frequent sub-graph mining is essential. Frequent sub-graph (and sub-tree) mining, using various approaches, has been extensively studied [73], [69], [65], [54] and [49]. However, the main bottleneck is the number of unnecessary candidate frequent sub-graphs generated. A substantial amount of work has been undertaken focusing on developing efficient graph mining algorithms using elegant search strategies, data structures or their combinations. Some authors have suggested the use of constraint based frequent sub-graph mining to remove unwanted patterns. The weighted sub-graph mining approach advocated integrates the weight constraints into the frequent sub-graph mining process to reduce the search space by generating only the most significant (interesting) patterns.

The frequent sub-graph mining approach is influenced by work on weighted pattern mining, especially Weighted Association Rules Mining (WARM), see for example the work of [101], [51], [38], [30] and [28]. A significant issue in WARM is that the "Downward Closure" (DC) property of items sets, on which many ARM algorithms are based, no longer holds. One solution (for example [101]) is to handle the weights as a post-processing step after mining frequent item sets, however the weights are then not integrated into the ARM process.

Tao et al. [51] proposed a model of weighted support, which satisfies a weighted graph property. *Yun et al.* [38], [102] and [28] introduced a series of concepts such as "weight range", "weight confidence", and "support confidence" for WARM in order to maintain the graph property and push the weight constraint deeply into the mining process. Although the ideas espoused by WARM cannot be directly applied to weighted frequent sub-graph mining.

2.2 Graph-based modeling over SOM

some proposed studies are previewed which tried to solve the weaknesses of VSM by graph-based models. Most of these studies improved successfully the quality of the resultant clusters. Semantic graph is used for document clustering as in [68], [93]. In this algorithm, a semantic graph is used to represent semantic relationships in documents then convert those graphs into vectors. Vectors then enters classical SOM algorithm. An improvement in the quality of document clustering is shown but they did not propose a direct technique to use the semantic graphs directly with SOM. *H. Chim* and *X. Deng* [27] proposed Suffix Tree Clustering (STC) algorithm which is a phrase-based document clustering approach. The performance achieves **`nlog(n)`** with a high quality resultant clustering.

The major disadvantage of STC is that a high number of redundancies words in the nodes of the suffix tree. STC was extended in [48] by proposing a Document Index Graph (DIG) model to represent the collection of documents. The redundancy, the major limitation in STC, of data in the nodes is avoided. In [48] DIG is adapted to use it in a suitable way with GGHSOM. *B. Choudhary* and *P. Bhattacharyya* [5] proposed a new suffix tree similarity measure for document clustering based on STC. This new method detects the overlap nodes between documents. Although, they did not solve the major disadvantage of STC which is the redundancy.

S. Zha [3] used Quantization errors (QE) and Topological errors (TE) as measurements of classification results. A new way to create the Multiple Spherical Self-Organizing Map (MSSOM) which is based on SSOM and avoids some ill-effects from the SOM. In order to explore a more effective method for classification, the distance between spheres is no longer a constant, but depends on the number of spheres and the size of the spheres. Due to the change of the distance between spheres, the neighborhood structure becomes more flexible. The results show that the method is feasible and worth further investigation.

After using Generic Meta-Data-Models for Clustering Medical Data and evaluating a number of clustering algorithms, *D. Girardi*, *M. Giretzlehner* and *J. Küng* had presented a data storage system for research, clinical studies or disease registers [1]. Their system is highly customizable and allows the user to set up a professional web-based data acquisition system including administration area, data input forms, overview tables and statistics within hours.

2.3 SOM visualization

Several studies that relates to using SOM for generation of topological maps of textual documents have been published. *A. Becks*, *S. Sklorz* and *M. Jarke* [71] use SOM as visualization method which allows easy access to enterprise document collection and they suggest a modularization of similarity definition. *K. Lagus*, *T. Honkela*, *S. Kaski* and *T. Kohonen* [87] developed WebSOM, visualization system for exploration of large collection of Internet Newsgroup e-mails. Documents were mapped from their n-dimensional document content space to 2d map of neurons with topology preservation. After this, each neuron is labeled with Newsgroup name that most documents mapped to this neuron belong to. *L. Soergel* and *Marchionini* [97] use SOM to construct a self-organizing map for information retrieval. They create a map of AI literature documents and later divide map into regions – word areas.

In these approaches, SOM is used as visualization method that maps documents to 2d map in a way that similar documents are mapped to neighboring positions. Clustering is here a side effect of this mapping, as in [71] and [87] where similar documents will be mapped to the same neuron. Problem with this clustering is that when map with many neurons, what is usually the case, too many clusters were produced. This problem is solved in [97] where all neurons are first labeled and then all neurons with the same label collected into one cluster. However, this clustering method does not offer interactivity with user defined categories and number of clusters. By decreasing the resolution of the SOM grid, the number of clusters will be decreases thereby the documents position will not be correct any more.

In [45] suggest a method that decouples visualization processing and clustering processing. SOM is used only as visualization method that maps document space to 2d space with topology preservation. Clustering is obtained in parallel processing what offer possibility for interactive clustering without losing precision of SOM visualization.

A visualization scheme CONNvis uses a topology representing graph that shows detailed local data distribution which brief proposes graph representation can be adapted to show local distances. *K. Tasdemir* [11] presented graph Based Representations of Density Distribution and Distances for Self-Organizing Maps. The proposed graphs provided tools to analyse the correlation between two pieces of information and achieved an advanced visualization by merging information in various ways. SOM produces a 2d spatially ordered quantization of a higher dimensional data space. To determine the optimal approximation of the density distribution of the data adapted a rigid lattice of visualization scheme to capture the data manifold.

A SOM is typically done interactively from various visualizations but it needs post processing to identify the clusters. *K. Taşdemir* and *E. Merényi* [22] showed that discovery through CONNvis clustering in a remote sensing spectral image from the Mars Exploration Rover Spirit. SOM has been used effectively for remote sensing spectral images which often have high-dimensional feature vectors (spectra) and many meaningful clusters with varying statistics and considered a powerful method for Cluster Analysis in Remote Sensing Spectral Imagery through Graph Representation and Advanced SOM Visualization. the SOM’s knowledge are presented by a visualization has great importance for cluster capture.

A graph based SOM visualization CONNvis is a suitable tool for represent the data topology on the SOM lattice despite the rigid grid structure. CONNvis   can show the topology preservation of the SOM and the extent of topology violations. *K. Tasdemir* [23] presented an application of CONNvis for Exploring Topology Preservation of SOMs with a Graph Based Visualization. SOM is a commonly used manifold learning algorithm and can exploits the underutilized knowledge of data topology. SOM can projects a (high-dimensional) data manifold onto a lower-dimensional (usually 2d) rigid lattice. existing visualization schemes are often designed to show the similarities local to the lattice without considering the data topology.

The urban surfaces represent a challenging environment for Knowledge discovery. The urban surface has a very large variety of materials concentrated in relatively small areas. Key components in the comprehensive analysis and understanding of Urban surfaces are co-registration, 3d surface reconstruction, object recognition, spatial reasoning, high-quality detailed and precise segmentation of remote sensing spectral images. Those challenges still with all the exciting advances in sensor fusion and data interpretation technologies in recent years.

2.4 Other SOM-based studies

In an early study, Lin formed a small map of scientific documents based on the words that occurred in the titles [95] and [96]. Later Lin has extended his method to full-text documents [80]. *Scholtes* has developed, based on the SOM, a neural document filter and a neural interest map for information retrieval [44]. In addition to encoding the documents based on their words, Scholtes has used character *n*-grams, sequences of *n* characters, in the encoding. This approach has also been adopted in [68]. *Merkl* [83] and [91] has used the SOM to cluster textual descriptions of software library components. Document maps have also been created by *Zavrel* [88]. The AI Group at the University of Arizona has used the SOM in their "ET-Map" in categorizing the content of Internet documents to aid in searching and exploration [84]. A system that is very similar to the webSOM has recently been used to organize collections of scientific articles on astronomy [80] and [77].

2.5 Music visualization

In Music visualization, there is so much work done on visualizing and exploring music libraries based on the semantic attributes of tracks. However, there is a relevant research community working on how to visualize and organize music based on signal processing techniques. In that sense, Islands of Music [70] and [40] organizes music libraries without requiring genre or other attribute classification because it uses psycho-acoustic models, and then tracks are visualized using a metaphor of geographical maps where islands resemble genres of music styles.

A different approach using a heuristic version of Multidimensional Scaling (MDS) named FastMap is described in [62]. With a similar goal to FastMap, Sonic Browser [70] uses sonic specialization for navigating into audio libraries and the Marsyas3D tool [6] proposes techniques based on principal component analysis for browsing audio libraries. Another related work is enlighten in [52] but it is more focused on how to identify any potential misfits between the designers' views of a product or system, embodied in the device itself, and those of its users.

B. Logan presented in [63] an approach to form playlists from a given seed song. Their technique is based on their own audio content similarity measure introduced in [67]. *Paws* and *Eggen* [64] propose to generate automatically playlists with an inductive learning algorithm considering different context-of-use of music consumers. All these previous works are based on audio information and use signal processing techniques. Another related work is the Variations2 project [58] at the Indiana University. They exploit music bibliographic data for providing visualization methods in order to assist music students and faculty members.

Chapter 3
Features Extraction

3.1 Overview of Feature Extraction

Feature Extraction is a branch of pattern recognition and image processing. Dimensionality reduction is the main purpose of feature extraction. For large input data set, there will be redundant suspected to an algorithm. Therefore, the input data will be transformed into a reduced representation set of features [34]. The transformation of input data into the set of features is called feature extraction. When features carefully selected, represented less carefully selected facilities rather than full resolution using input requested from relevant input data to extract information expected, actions are performed the desired task using this reduced representation instead of the full size input.

The selection of a large set of data needed to accurately describe the amount of resources involved. One of the major problems in conducting an integrated analysis of related data to the number of variables. Analysis with a large number of variables generally requires a large amount of memory and computation power or a classification algorithm which overfits the training sample and generalizes poorly to new samples requires a large amount. despite of these problems still describes the data with enough accuracy to a combination of variable construction methods for a generic term. Best results are achieved when an expert constructs a set of application-dependent features. Nevertheless, if no such expert knowledge is available general dimensionality reduction techniques may help.

Feature Extraction Algorithm Examples are: Principal component analysis, Semi-definite embedding, Multi-factor dimensionality reduction, multi-linear subspace learning, Nonlinear dimensionality reduction, ISO-map, kernel PCA, Multi-linear PCA, latent semantic analysis, partial least squares , independent component analysis and auto-encoder. Finally, Google's PageRank was chosen in this thesis.

3.2 Google's PageRank

Over the past few years, Google so far the most widely, used search engine in the world. Therefore, a crucial factor, high performance, and use the results simple addition of excellent quality to search for more than one country other search engines. A lot of the quality of the search results, the site, and complex method based on the rank of web documents. Make sure to conduct a comprehensive study of all aspects of the site the purpose of these pages [34]. Basically the content of those pages Google founders *Lawrence Page* and *Sergey Brin* at Stanford University as a graduate student at the time of the newspaper. It is often particularly Internet dynamics and a long time in view of the scientific work on PageRank has passed since, it is said that, and it still could be the basis for ranking search engine Google. In past years, there probably Google rating methods would have a lot of changes and modifications in that there is no doubt, but is crucial to the success of Google, PageRank for at least the second basic concept.

3.2.1 The PageRank Concept

It is in the early stages of the World Wide Web, search engines arrangement webpage in different ways. Today search for a phrase in the document, and in any case, a major factor in search engine ranking methods. webpages undergoes in measured phrase can thereby be weighted by the length of a document (in the classification of keywords density) and the persistence of long tag in the HTML document. Especially in order to improve the results of the search and content of specific search engine ranking criteria (doorway pages) automatically on the basis of the analysis of resistance to create, webpages has been developing the concept of link popularity [6]. Following this concept, the total value to determine the number of external links to the document. Thus, the webpage is usually more important if it is associated with many other webpages through the creation of webpages , as well as to avoid trivial.

Unlike the concept of link popularity, is not simply based on the total number of inbound links. The basic approach of the site and links to other documents, are considered more important in the document, but these external links do not count on an equal footing. If all the references to other high-level documents, paper, site, rank higher than the mattress. Thus, under the concept of the site, which connects in the ranks of those documents by e- document. Word documents in their ranks again, since they indicate. So always the site definition site document repeatedly other documents [34]. Since then, even if the limit and has a lot of connections - any document through the ranks will affect other rank, page rank, link structure of the web site . Although this approach seems to be very broad and complex, *Lawrence Page* and *Sergey Brin* were able to put it into practice by a relatively trivial algorithm.

3.2.2 The PageRank Algorithm

The original PageRank algorithm was introduced by *Lawrence Page* and *Sergey Brin* in several publications. given:

`PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))`

where

- **`PR(A)`** is the PageRank of page **`A`**,
- **`PR(Ti)`** is the PageRank of pages **`Ti`**, which link to page **`A`**,
- **`C(Ti)`** is the number of outbound links on page **`Ti`** and
- **`d`** is a damping factor which can be set between **`0`** and **`1`**.

First and foremost, the site does not take web sites in general, which is defined individually for each page [66]. Moreover, Page Ranks pages is determined by the link to the page from the pages of Transparency International **`(Ti)`** which refers to the webpage on the site page is not on an equal footing. In the algorithm, location is always the amount of outgoing links on any page with a weight of means that more than the former, but will reduce the use of **`T`**. link on this page likely location of the pages is then added **`Ti`**. As a result of this that the external links on the page will always increase the webpage **`A`**. Finally, the damping coefficient is multiplied by "**`d`**", which can be set between **`0`**and **`1`** total weighted PageRank of all pages. Thus, it was connected to another page on the site for less page for the distribution of benefits.

3.2.3 The Random Surfer Model

In publications, and *Lawrence Page* and *Sergey Brin* in a very simple, intuitive evidence for Page Rank algorithm. Where are they random connection, respectful behavior of materials surfer clicks site as a model to consider [6]. PageRank of a page run random surfer visited webpages with a certain probability [18]. Server is completely random, and probably clicks on the link to the number of links on this page. Therefore, site of the pages completely linked to it, but it is divided by the number of links on the page does not move.

So, is the probability of a random surfer casual surfer to access the page , click the link to the page Total possibilities. Now, this capability reduces **`d`**. damping coefficient of the random surfer model, internal justification, as the surfer link, click an infinite number, but sometimes gets bored and moves on to another page in random order [18]. The possibility of stop random surfer clicking on a link to the damping coefficient **`d`**, which depends on the degree of probability , therefore, between **`0`** and **`1`** is shown. **`d`** the highest, and most likely keep pressing random surfer on the link. It stops after clicking on a random link surfer moves to another page, so the algorithm continuously probability **`(1 - d)`**, as implemented. Regardless of external links , which is a random surfer, there is always the possibility of jumping the top of the page （**1 - d**）, the minimum webpage always [16].

3.3 A Different Notation of the PageRank Algorithm

Lawrence Page and *Sergey Brin* have published two different versions of their PageRank algorithm in different papers [16]. In the second version of the algorithm, the PageRank of page **A** is given as

```
PR(A) = (1-d) / N + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))
```

Where **N** is the total number of pages tested by PageRank. Algorithm, "second edition", in fact, essentially no different from the first. As for the random surfer model, it is likely that the actual number to access this page after clicking on a link to the latest version of Site Server page. PageRank then the probability distribution for a webpage, all pages in total PageRank will be one. In contrast, the probability that the first version of the algorithm surfer random access to the page is likely the total number of webpages. If you double, in the case of run 100 times 100 pages and web Site Value Page 2, Surfer casual access to an average page [16]. As mentioned above, the most important of the two versions of the algorithm does not differ much from each other.

They claimed the first version of the algorithm as the connection. The reason is that the site through the algorithm calculated easily because the total number of webpages cannot be ignored. As mentioned above, one of the two versions of the algorithm is not significantly different from each other. According to the second version of the algorithm PageRank of the web must be multiplied by the total number of pages that have been calculated using the first version of the classification of pages calculated first order. All pages with a PageRank of webpages to generate a probability distribution to claim first version of the algorithm so far as that *Lawrence Page* and *Sergey Brin*, broadband Internet, a hypertext search engine Anatomy" algorithm in two versions of their most popular.

3.3.1 The Characteristics of PageRank

Here is a small example to explain the site with features [34]. It is supposed to have the three pages. Page **A** contains links to page **B** and page **C**. Page **B** links page **A**. Page **C** contains links to page **A**. where **A, B** and **C** consisting of a small network. *Lawrence Page* and *Sergey Brin*, the damping factor **d** is generally （**0.85**）, but keep a simple counting value of **0.5**. The true value of the damping factor **d** is an impact on website, but the page does not affect the classification of basic principles [16]. Thus, the following equation is used to calculate the PageRank:

```
PR(A) = 0.5 + 0.5 PR(C)
PR(B) = 0.5 + 0.5 (PR(A) / 2)
PR(C) = 0.5 + 0.5 (PR(A) / 2 + PR(B))
```

These equations can easily be solved. Where following PageRank values for the single pages are resulted:

```
PR(A) = 14/13 = 1.07692308
PR(B) = 10/13 = 0.76923077
PR(C) = 15/13 = 1.15384615
```

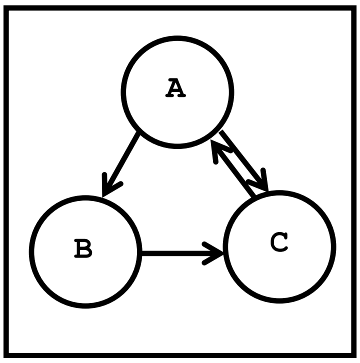


Figure 2: Connecting A ,B and C pages

Of course, all page (3) total number of webpages and the amount of PageRank. As mentioned above, this is not a specific simple example results. Page rank values which used to determine a system of equations to solve, are easy examples of three simple pages.

3.3.2 The Iterative Computation of PageRank

Because the actual size of the web search engine, Google uses an approximate calculation run page rank values [6]. It first and then each page PageRank multiple circles calculation equations all pages, the algorithm calculated from the start is set to what that means. it has assigned a PageRank of each page 1 value as illustrated by the example account should be three pages.

After a few iterations the real PageRank values us a good approximation. Get a good approximation of the value of the network to 100 iterations about *Lawrence Page* and *Sergey Brin* publications [16]. In addition, through iterative computation of PageRank for all pages still surpassed the total number of pages on the Internet. Thus, the average location of the webpage. At least the page server（**1-d**）was provided. The total number of webpages. Thus **dN +(1-d)** Specify a limit to one page PageRank, where **N** there is only one page to all webpages linked to, theoretically, it could be, and that the pages itself.

3.4 The Implementation of PageRank

The ranking of webpages by the Google search engine was determined by three factors Page specific factors, Anchor text of inbound links and Page Rank. Page specific factors are Tag names or URL address of the document such as body next page specific factors [6]. As *Lawrence Page and Sergey Brin* content search engine Google in the first rank among the most likely to be

Iteration	PR(A)	PR(B)	PR(C)
0	1	1	1
1	1	0.75	1.125
2	1.0625	0.765625	1.1484375
3	1.07421875	0.76855469	1.15283203
4	1.07641602	0.769104	1.15365601
5	1.076828	0.769207	1.1538105
6	1.07690525	0.76922631	1.15383947
7	1.07691973	0.76922993	1.1538449
8	1.07692245	0.76923061	1.15384592
9	1.07692296	0.76923074	1.15384611
10	1.07692305	0.76923076	1.15384615
11	1.07692307	0.76923077	1.15384615
12	1.07692308	0.76923077	1.15384615

Table 1:iterative calculations of PageRank

associated. This, however, is not of interest here. In order to provide search results, Google is most likely from IR page inside and incoming links, counts the number of anchor text of the page and the document search. Thus, the importance of a document is requested. Then the overall importance of the page as an indicator of the outcome of a combination of IR site. Results in a combination of two IR multiplier value.

If pages prepared high page rank, search request well optimized result because it is essential for good rankings to have a high PageRank. In particular, two or more search terms and site effects in the first place is a term for non-specific questions, while content classification standards very high, there are questions. Target webmasters find them two or more words phrases classic search engine optimization to get higher on the best layout pages, it is possible [18]. Search terms are very competitive page is customized, so a high page rank is important for good ranking, a page that is optimized as a classical search engine optimization. So why to avoid repeating the word infrared, spam the document or anchor text of inbound links more often reduces the words degree increased by it. Thus a classic search engine optimization and PageRank for potentially competitive areas become the determining factors.

3.4.1 The Distribution of PageRank

Now the number of pages and respectively the incoming and outgoing links, described as the number of the PageRank. This internal structure of linked website's search engine optimization can be affected, should be discussed in a page rank. As the example shows, in most cases, sites, to some extent, are arranged in a hierarchical form, the web site pages, including **A**, **B** and **C** are usually the root page optimization execution of search query. In this example, page of outgoing links and PageRank **10** link **X**, someone which has been added to the page. **B** and **C** each reference page is a link to it. A dabble in the pages of PageRank values set **d** to **0.5** equations is shown below:

```
PR(A) = 0.5 + 0.5 (10 + PR(B) + PR (C))
PR(B) = 0.5 + 0.5 (PR(A) / 2)
PR(C) = 0.5 + 0.5 (PR(A) / 2)
```

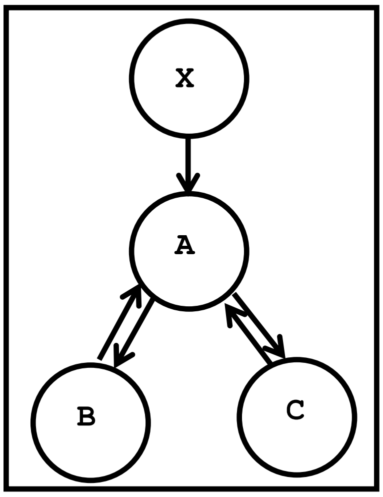


Figure 3: X connected only to A

Solving the equations gives us the following PageRank values:

```
PR(A) = 8
PR(B) = 2.5
PR(C) = 2.5
```

This is generally the root of the only places to search customization page to work on is not recommended. In fact, in most cases, more efficiently separate the search terms to improve the every page of the site. An example at the root of the webpages provides satisfactory results for inter, but the site's other pages, where the structure of the binding site have been changed **B**, **C**. Page and site examples of hierarchical structure is linked to the first vice versa. Again, no outgoing links and PageRank page is a page **X**, a new link. Damping **d** values of PageRank **(0.5)**, pages one factor equation for:

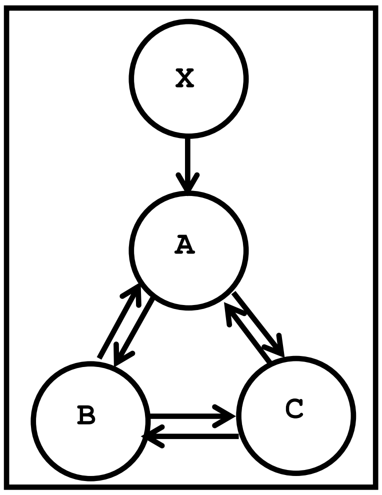


Figure 4: connecting A , B and C

```
PR(A) = 0.5 + 0.5 (10 + PR(B) / 2 + PR(C) / 2)
PR(B) = 0.5 + 0.5 (PR(A) / 2 + PR(C) / 2)
PR(C) = 0.5 + 0.5 (PR(A) / 2 + PR(B) / 2)
```

Solving the equations gives us the following PageRank values:

```
PR(A) = 7
PR(B) = 3
PR(C) = 3
```

As a result of targeted keywords to add internal links to pages of search engine results can develop in which pages **B** and **C**, to rise in value. On the other hand, of course, probably due to a lack of page at your page rank. Generally speaking, for the purpose of search engine optimization is more evenly between pages, and will deliver more cross-categorize pages PageRank.

3.4.2 Well Directed PageRank Distribution

This outgoing external links on the online webpages are proven to have a negative impact. This is the mechanism for regular external links search optimization can be reduced in order to have an distribution. We take a look at another site, for example, **A, B, C** and **D,** are an organized hierarchy of pages **B, C** and **D** each page, in addition to a reference to a page **B, C** and **D** an outgoing link. None of these external pages to link to those pages, for example our clients associated with **B, C** and **D**. We assume a damping factor **d** of **0.5**, the PageRank of page to calculate the value of:

```
PR(A) = 0.5 + 0.5 (PR(B) / 2 + PR(C) / 2 + PR(D) / 2)
PR(B) = PR(C) = PR(D) = 0.5 + 0.5 (PR(A) / 3)
```

Solving the equations gives us the following PageRank values:

```
PR(A) = 1
PR(B) = 2/3
PR(C) = 2/3
PR(D) = 2/3
```

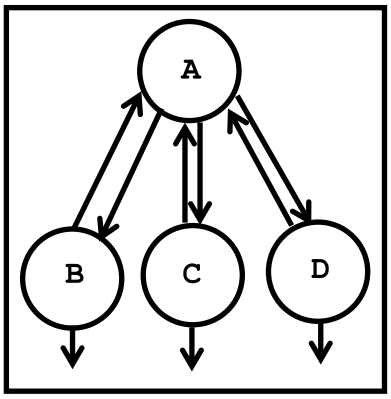


Figure 5: D,B and C have links to external pages

Now **D,B** and **C** do not have any contact with outside of all three links and change our example page to external pages. The way ahead for the General conditions are the same as above. A link back to this example pages getting a link from any external pages. Once again, damping factor **d** of **0.5**, the equations for the single pages of Page Rank values appear:

```
PR(A) = 0.5 + 0.5 (PR(B) + PR(C) + PR(D) / 4)
PR(B) = PR(C) = PR(D) = 0.5 + 0.5 (PR(A) / 3)
```

Solving these equations gives us the following PageRank values:

```
PR(A) = 17/13
PR(B) = 28/39
PR(D) = 28/39
```

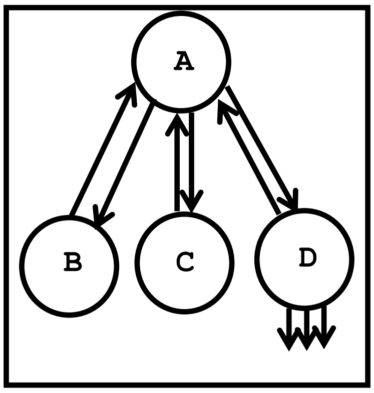


Figure 6: D has more than one external link

As a result of the modifications, we have every page of the website increase the PageRank for that opinion. About search engine optimization as long as it does not reduce the usefulness of the site as possible, as the concentrate, therefore appropriate. Search engine optimization and to increase their popularity with many websites link to other people. As a closed system of webpages do not affect the accumulated PageRank links in pages. Link exchanges all positive rating page, thus it is doubtful.

Effects of link exchanges, a structured hierarchy of pages which has two sites, **A, B, C, D, E** and **F**, respectively, as an example. The site has a link to page **B** and **C** and they have a link back to the page on the first page. Values need not be computed explicitly pages PageRank so that web design II, respectively. Damping **d** values of PageRank 0.5, pages one factor equation for:

```
PR(A) = 0.5 + 0.5 (PR(B) + PR(C))
PR(B) = PR(C) = 0.5 + 0.5 (PR(A) / 2)
```

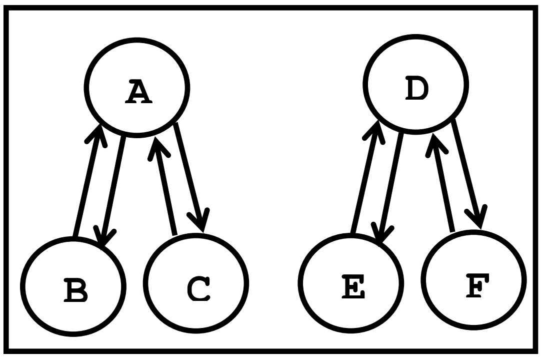


Figure 7: two different hierarchies

Solving the equations gives us the following PageRank values for the first site

```
PR(A) = 4/3
PR(B) = 5/6
PR(C) = 5/6
```

and accordingly for the second site

```
PR(D) = 4/3
PR(E) = 5/6
PR(F) = 5/6
```

Two pages of sample sharing sites now link to get started. Page **D** link to the page and vice versa. The same has been done as above example under normal circumstances, then, the damping factor **d** of 0.5, the equations for the value of PageRank with pages set:

```
PR(A) = 0.5 + 0.5 (PR(B) + PR(C) + PR(D) / 3)
PR(B) = PR(C) = 0.5 + 0.5 (PR(A) / 3)
PR(D) = 0.5 + 0.5 (PR(E) + PR(F) + PR(A) / 3)
PR(E) = PR(F) = 0.5 + 0.5 (PR(D) / 3)
```

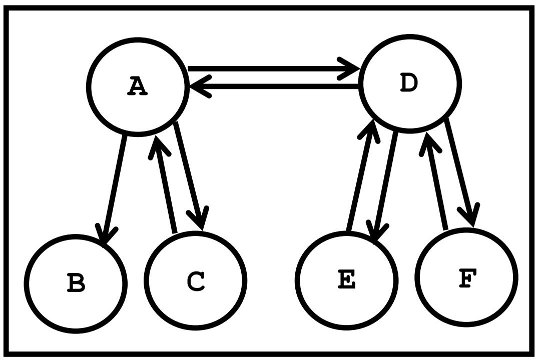


Figure 8: linking two hierarchies

Solving these equations gives us the following PageRank values:

```
PR(A) = 3/2
PR(B) = 3/4
PR(C) = 3/4
PR(D) = 3/2
PR(E) = 3/4
PR(F) = 3/4
```

All other pages lose PageRank while link exchange page, PageRank in case **A** and **D** that uses. Search engine optimization as binding takes place opposite effect within low hierarchical pages that means. Exchange links, and therefore recommends that you do the same sentence a page (such as the website root page). Link exchange is one of the basic principles of the positive effects of both pages have a similar amount of the location of a second deployment.

One of the pages or abroad at least a high page rank links, it is likely to lose all the pages PageRank. Here's an important factor is the size of the site.

More pages to participate in link exchanges, regardless of the number of outgoing links on the page, other pages on the site have a PageRank of website inbound links are distributed throughout. This reduced the advantages of sharing relevant link exchange pages and episode pages have multiple link exchanges posted as to rank pages. This is a link to a second deals Beaver affecting factors to weigh. In the end, the other site also participates in link exchange page rank of the website using the link exchange pages does not lose all PageRank so it is possible that should be noted. Page already links to a site that has a number of outgoing links, included in the exchange complex.

Chapter 4

SOM and GHSOM clustering Algorithms

4.1 Self-Organizing Map

4.1.1 Classic SOM

Self-Organizing Map (SOM) has been developed by *T. Kohonen* since 1980 and it has been used for clustering [20],[43]. SOM is a powerful method for data clustering. SOM can analyze data sets with varying statistics such as sizes, shapes, density distribution, overlaps, etc. Moreover, SOM has the capability of data clustering without defining number of clusters [75].Traditional clustering algorithms such as k-means does not provide such important capabilities which makes SOM better. *Do Phuc* and *Mai Xuan* Hung have developed a system for clustering the graphs [19]. They use SOM neural network for clustering the graphs and extracting the main ideas from the documents. They make SOM put the documents on a document map and help to access the content of similar documents.

Initialization problem is the most challenging issue of the learning clustering algorithms. Preservation of relationships of the topology is another big limitation to learning clustering algorithms. Using SOM to perform clustering will overcome this drawback because of its well-known neural network model [2], [78]. Clustering process preserved or visualized the topological relationships among data clusters on the network. However, SOM suffers from some drawbacks such as the cost of objective function, a general proof of convergence, and a probabilistic framework [76].

In training phase, A vector for each input data representing all features enters training process resulting the map of SOM in which similar input data classified onto clusters in according to similarity measurements. The predefined fixed architecture of SOM, growing input data will produce huge maps that are hard to handle. This is really considered a drawback of SOM.

A limitation of SOM and all other variants models receives Vector Space Model (VSM) as an input. A vector is structure composed by scalar and context. The scalar, mostly a number, is used as an index to the context. The context mostly be a text. Mathematically, a vector space is used as a collection of vectors. Vector space model is an algebraic model for representing text documents. In VSM, all words in the documents are structured as vectors. The facilitates indexing, filtering, information retrieval and ranking.

A distance-based Similarities in neighborhoods are the represented in Most of the proposed visualization schemes. Some of those schemes are U-matrix [94] and its variants [6], [89]. The size and shape of the cells to represent the prototypes are adapted in [85], [57]. Alternatively, some methods use Euclidean distances to update the grid positions of the prototypes for visual inspection as adapted lately in [82] and double SOM [66] and visualization-induced SOM (ViSOM) [55].

Size of receptive fields [57] and smoothed histograms [59] are another methods that uses density-based visualizations. However, density-based representations are less helpful compared to the distance-based visualizations unless density representation has a higher resolution than the receptive field size.

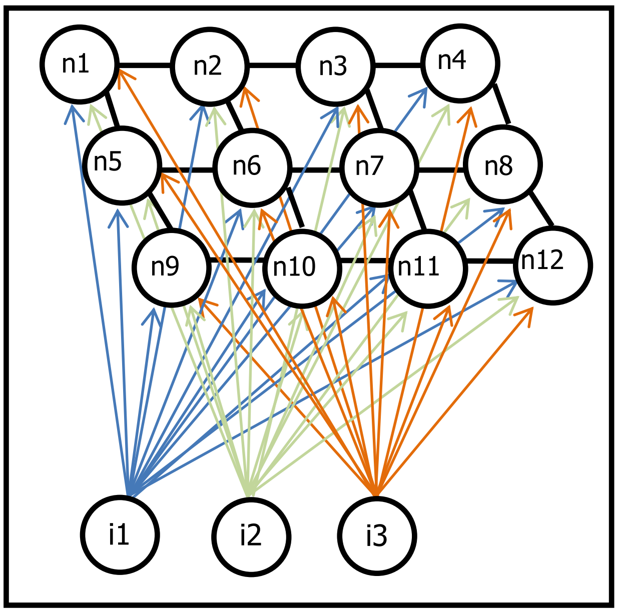


Figure 9: SOM composing illustration

The SOM is a neural network that consists of two layers of processing units (Figure 9): the first is an input layer containing processing units for each element in the input vector, the second is an output layer of processing units which are fully connected with those at the input layer.

The SOM algorithm is based on unsupervised, competitive learning. It maps the high-dimensional space to a two dimensional grid and has the property of topology preservation which means that the SOM groups similar input data vectors at similar neurons: points that are near each other in the input space are mapped to nearby map units in the SOM.

4.1.2 Modified Graph-based SOM

Modified SOM is fed with the similarity graph that resulted from text pre-processing with Google PageRank algorithm. Each sentence is represented with one graph node. Here is important to say that we choose a graph data structure to get a dynamic dimensions of SOM grid so big that there was approximately no limit number of sentences mapped to the neurons. This is important because we want clustering at the level of neuron and use SOM only as visualization method. After learning the SOM, each sentence graph is mapped to one of the map neuron that is the for this sentence. The result is illustrated on Figure 24 Each point represents a sentence and each line crosses a neuron.

4.2 Growing Hierarchically Self-Organizing Map

Growing Hierarchically Self-Organizing Map (GHSOM) comes with a growing hierarchical architecture. The growing developing according to data distribution, allows a hierarchical decomposition and navigation in the map and subsections horizontally and vertically [66]. The structure and the size of the map are dynamically adapted due to newly coming input data. GHSOM structure looks like central library. It is composed with floors. Each floor is composed with sections. Each section is composed with subsections. Each subsection is composed with shelves.

GHSOM is one of most outstanding models of SOM [66]. Self-organizing neural networks with a dynamic topology have been proposed [17], [50] although they do not feature a hierarchical structure. GHSOM model tries to solve growing and navigation problems of SOM. It consists of several independent growing SOMs [90] arranged in layers. The unsupervised learning process determines the number of layers, maps and neurons. This neural model gives hierarchal relations between input data. The resulting fixed topology leads to a number of drawbacks later on. One drawback that the topology is always must be a 2d rectangular grid [9]. Not all of input data will fit in this topology leading to a rigid topology. Another drawback appears with very large maps. The map should preserve a rectangular topology [90]. This means when the map grows, a row or a column of neurons must be inserted whether they are needed or not. As a result, additional neurons are used as a padding to preserve topology.

In spite of the mentioned drawbacks of GHSOM, it is still the most efficient model among the variants of SOM. GHSOM has many different applications varies from pattern recognition to document clustering. A drawback in all varieties of GHSOM is working directly on the observation vector. Observation vector does not take the order of coordinates into account.

In some cases, such as processing a time series, it might prove inappropriate to the specific nature of data. The original GHSOM did not define visual distribution on the input space. However, the advantages of introducing a graphical methodologies into SOM models were soon evident. This has led to a wide range of proposals which reflect the importance of graphical approaches to data clustering.

The underlying combinative theories behind them derive from two main ideas: the transforming GHSOM to accept graph as an input and capturing the output topology as an output. The current study presents a comprehensive view of the state of the art, with a perspective of the involved theatrical frameworks. The study examines the most commonly used graph model representation.

4.3 Graph Representation

Classification is a complex process due to enormous size and the lack of effective methods to classify. The classification algorithms must meet the needs of classifications. Some of the already existing algorithms are efficient for specific types of information but not so for other types. It is quite impossible to find a general algorithm sufficient enough for all types of information.

Concerning the types of info information, take documents as an example of data sets. Each document contains words, sentences and paragraphs. A single sentence represents a structure of correlated words where each word has its weight and function with respect to other words in the same sentence bearing in mind that a word alone has no function consequently has it is weightless. A group of coherent, related and sequent sentences compose a meaningful paragraph where each sentence has its own function. Concerning the meaning, a document has a topic where the paragraph revolved around it. In another word, the paragraph holds sub-ideas working together to form the main idea of the document.

A graph is an abstract representation of a set of words where some pairs of the words are connected by links. The interconnected words are represented by mathematical abstractions called vertices, and the links that connect some pairs of vertices are called edges. Idiomatically, the graph is a diagram consists of a set of dots for vertices, joined by lines for edges.

In the classification algorithm based on graphs, the words within the document are represented as a graph of that document. The weights of the words are estimated with the help of co-occurrence of the word as a measure of the relationship between them. The weight of vertices calculated according to its function in the sentence and with respect to its global function to the whole document. Graph does not ignore the important information about words such as position, semantic relationship. There are some researches that works with a graphs to represent the document [15] , [46], [50].A model to represent the data set as an input to clustering algorithm is needed, as well as a model to show the output clustering. Graph model seems to be the best option because it is the computation is fast and scalable and its structure is flexible to incorporate many performance enhancement techniques.

In [20], [27], [48], [6] and [68] graph-based clustering algorithms have proven an improvement in the quality of clustering. A combination of GHSOM model with graph model improves the quality of resulting clusters more than algorithms use GHSOM model with VSM [27].

4.4 Graph Input for SOM and GHSOM

GHSOM is extended by *F. Hussin*, *M. Farra* and *Y. Sonbaty* [20] to work in the graph domain by using graphs to represent documents then cluster those documents by modified GHSOM called Graph-based Growing Hierarchal SOM (GGHSOM). GGHSOM enhances the quality of clustering. After testing GGHSOM on two different document collections, using three different measures for evaluating clustering quality, the experimental results show an improvement in words clustering compared to classical GHSOM.

Using VSM for document representations does not represent any relationships between the words thus VSM does not build any semantic or form any context of meaning to the document. VSM breaks sentences into their individual components regardless to its grammatical function or significant meaning. In contrast, Graph model capture the salient features of data [6]. In addition to index words in vertices, a weight relationship is stated in edges. Besides the magnitude of the edge to represent the relations, the edge has a direction. For example, a word "A" has 5 points as weight to its relation towards the word "B". But the word "B" has 1 points as weight to its relation towards word "A". VSM uses phases of document collection as features which is required a huge space while using the graph, required one graph. This decreases the complexity of the GGHSOM algorithm.

4.5 Clustering Visualization

In general, clustering is exploratory data analysis tool. However, clustering tools produce static results by nature. There are many visual systems for data exploration and understanding. These systems interact with clustering results of large multidimensional data sets easier [74].

Visualization of clustering results highlights the interrelationships between documents by employing graph capabilities of illustrating relations. Unlike ranked list presentation of documents, 2D and 3D graph presentations allow the clusters to identify and assess rapidly. A trained SOM forms knowledge about the input data sets. Visualization tools can extract any information from SOM by the help of cluster detailed provided. Therefore, visualization of learned SOM are widely used for information extraction and knowledge discovery, especially from large data sets such as remote sensing imagery.

CONN uses the recent cluster validity index to determine the number of clusters in datasets [29]. The similarities of SOM will be determined by CONN by defining local data distribution within the receptive fields of the prototypes using distance-based similarities [29].

Tools for clustering visualization are necessary to capture the topology of the resultant cluster. In the current thesis, circled bars , circled fans , Probabilistic Network and sunburst visualization will be used to visualize the learned GHSOM knowledge. These tools are recent, powerful and very effective with interactive clustering. For more detail see chapter 5.

Chapter 5

Visualization

5.1 Dijkstra's Algorithm

Dijkstra's algorithm commonly known as shortest path algorithm, solves the problem of identifying the shortest path in a weighted graph from an initial vertex to the other vertices [24]. The central idea behind the algorithm is that each sub-path of the minimal path is also a minimum cost path.

An upper bound of the running time of Dijkstra's algorithm on a graph with edges *E* and vertices *V* can be expressed as a function of E and V using big-O notation [8]. For any implementation of vertex set Q the running time is in $O(E * dk_Q + V * em_Q)$, where dk_Q and em_Q are times needed to perform decrease key and extract minimum operations in set Q, respectively. Dijkstra's algorithm can be implemented more efficiently by storing the graph in the form of adjacency lists and using a self-balancing binary search tree, binary heap, pairing heap, or Fibonacci heap as a priority queue to implement extracting minimum efficiently.

SOMvisua provides Dijkstra's algorithm application that runs in Input Panel. The Panel is split into two areas: graph, messages areas (see Figure 16). The first one is a panel where the graph is displayed with user control over the graph and edit nodes and weighted edges. The second one is the messages area. A table, presenting the current state of the algorithm structures (vertex nodes, cost and path), is displayed in this panel.

The messages area displays the animated execution , showing in blue the current step and in red the values of cost calculated, while the SOMvisua provides Animation Speed slide bar to adjust speed of execution. Active vertex node is highlighted when calculating cost and path. SOMvisua provides an input field to choose start vertex node to find the indicates the first step to be done. This panel also offers useful hints when the graph is being edited.

5.2 Topological Sort Algorithm

Topological sort algorithm is an ordering of the vertices in a directed acyclic graph, such that If there is a path from *u* to *v*, then *v* appears after *u* in the ordering. The graphs should be directed or otherwise for any edge (*u*,*v*) there would be a path from *u* to *v* and also from *v* to *u*, and hence they cannot be ordered [81]. The graphs should be acyclic or otherwise for any two vertices u and v on a cycle *u* would precede *v* and *v* would precede *u*. topological sort algorithm defines two terms, first, Degree of a vertex *u* is the number of edges (*u*, *v*) to the outgoing edges. Second, Indegree of a vertex *u* is the number of edges (*v*, *u*) to the incoming edges. The algorithm for topological sort uses "indegrees" of vertices.

Topological sort algorithm computes the indegrees of all vertices then finds a vertex *u* with indegree*0* and print it (store it in the ordering) [25]. If there is no such vertex then there is a cycle and the vertices cannot be ordered then stop. Topological sort algorithm removes *u* and all its edges (*u*,*v*) from the graph. The remaining vertices will be updated. Steps will be repeated while there are vertices to be processed [25].

SOMvisua provides Topological sort algorithm application that runs in Input Panel. The Panel is split into two areas: graph, messages areas (see Figure 17). The first one is a panel where the graph is displayed with user control over the graph and edit nodes and weighted edges. The second one is the messages area. A table, presenting the indegrees, is displayed in this panel. A final result of topological sort of vertices node is displayed.

5.3 Breadth First Search Algorithm

Breadth First Search (BFS) is a graph search algorithm that begins at the root node and explores all the neighboring nodes. Then for each of those nearest nodes, it explores their unexplored neighbor nodes, and so on, until it finds the goal [14].

BFS is an uninformed search method that aims to expand and examine all nodes of a graph or combination of sequences by systematically searching through every solution. In other words, it exhaustively searches the entire graph or sequence without considering the goal until it finds it. It does not use a heuristic algorithm [32]. From the standpoint of the algorithm, all child nodes obtained by expanding a node are added to a FIFO (i.e., First In, First Out) queue. In typical implementations, nodes that have not yet been examined for their neighbors are placed in some container (such as a queue or linked list) called "open" and then once examined are placed in the container "closed".

SOMvisua provides Breadth First Search algorithm application that runs in Input Panel. The Panel is split into two areas: graph, messages areas (see Figure 18). The first one is a panel where the graph is displayed with user control over the graph and edit nodes and weighted edges. The second one is the messages area. A list, presenting the queue, is displayed in this panel. A final result of Breadth First Search of vertices node is displayed.

5.4 Depth First Search Algorithm

Depth First Search (DFS) is an algorithm for traversing a graph. DFS needs to address building graph problems. For an unweighted graph, DFS traversal of the graph produces the minimum spanning tree and all pair shortest path tree [7]. A graph has cycle if and only if it see a back edge during DFS that made a cycle in a graph detection.

The DFS algorithm can find a path between two given vertices *u* and *z*. first, call DFS(G, *u*) with *u* as the start vertex. Use a Queue Q to keep track of the path between the start vertex and the current vertex. As soon as destination vertex *z* is encountered, return the path as the contents of the stack. To test if a graph is bipartite, first discover a new vertex, color it opposite its parents, and for each other edge, check it doesn't link two vertices of the same color. The first vertex in any connected component can be red or blue. DFS can be adapted to find all solutions to a maze by only including nodes on the current path in the visited set [4].

SOMvisua provides Depth First Search algorithm application that runs in Input Panel. The Panel is split into two areas: graph, messages areas (see Figure 19). The first one is a panel where the graph is displayed with user control over the graph and edit nodes and weighted edges. The second one is the messages area. A list, presenting the queue, is displayed in this panel. A final result of Depth First Search of vertices node is displayed.

5.5 Floyd-Warshall Algorithm

Floyd-Warshall algorithm is a simple and widely used algorithm to compute shortest paths between all pairs of vertices in an edge weighted directed graph [12]. This algorithm has a worst-case runtime of $O(n^3)$ for graphs with n vertices. This is remarkable considering that there may be up to $O(n^2)$ edges in the graph, and every combination of edges is tested. It does so by incrementally improving an estimate on the shortest path between two vertices, until the estimate is optimal. The Floyd–Warshall algorithm compares all possible paths through the graph between each pair of vertices.

The Floyd-Warshall algorithm outputs the correct result as long as no negative cycles exist in the input graph. In case that a negative cycle exists, computing a shortest path is an NP-hard problem and the Floyd-Warshall algorithm will not output the correct result. Rather, it will detect the presence of a negative cycle by checking that there is a negative entry in the diagonal of the matrix M.

Consider a graph G with vertices v numbered 1 through N. Further consider a function shortestPath(i, j, k) that returns the shortest possible path from i to j using vertices only from the set $\{1,2,...,k\}$ as intermediate points along the way. Now, given this function, our goal is to find the shortest path from each i to each j using only vertices 1 to $k + 1$.

For each of these pairs of vertices, the true shortest path could be either (1) a path that only uses vertices in the set $\{1, ..., k\}$ or (2) a path that goes from i to $k + 1$ and then from $k + 1$ to j [13]. It is known that the best path from i to j that only uses vertices 1 through k is defined by shortestPath(i, j, k), and it is clear that if there were a better path from i to $k + 1$ to j, then the length of this path would be the concatenation of the shortest path from i to $k + 1$ (using vertices in $\{1, ..., k\}$) and the shortest path from $k + 1$ to j (also using vertices in $\{1, ..., k\}$).

SOMvisua provides Floyd-Warshall algorithm application that runs in Input Panel. The Panel is split into three areas: graph, matrix Distance and matrix Path areas (see Figure 20). The first one is a panel where the graph is displayed with user control over the graph and edit nodes and weighted edges. The second one is the matrix Distance area where graph shortest path is stored during calculation time. The third one is the matrix Path area where final paths is represented.

5.6 Kruskal Algorithm

Kruskal algorithm is an algorithm in graph theory that finds a minimum spanning tree for a connected weighted graph. This means it finds a subset of the edges that forms a tree that includes every vertex where the total weight of all the edges in the tree is minimized. The algorithm consists in arranging by order of weight all of edges of the graph, then to withdraw one by one, test if the last arc does not form a cycle, and add this arc to the final tree [13].

Kruskal algorithm creates a forest F (a set of trees), where each vertex in the graph is a separate tree. Then it creates a set S containing all the edges in the graph while S is nonempty and F is not yet spanning. Finally, it removes an edge with minimum weight from S . If that edge connects two different trees, then add it to the forest, combining two trees into a single tree otherwise discard that edge. At the termination of the algorithm, the forest forms a minimum spanning forest of the graph. If the graph is connected, the forest has a single component and forms a minimum spanning tree.

For E is the number of edges in the graph and V is the number of vertices, Kruskal's algorithm can be shown to run in $O(E \log(E))$ time, or equivalently, $O(E \log(V))$ time, all with simple data structures. This bound can be achieve as follows: first sort the edges by weight using a comparison sort in $O(E \log(E))$ time; this allows the step "remove an edge with minimum weight from S" to operate in constant time. Next, A disjoint-set data structure is used to keep track of which vertices are in which components [10]. It is needed to perform $O(E)$ operations, two 'find' operations and possibly one union for each edge. Even a simple disjoint-set data structure such as disjoint-set forests with union by rank can perform $O(E)$ operations in $O(E \log(V))$ time. Thus the total time is $O(E \log(E)) = O(E \log(V))$.

SOMvisua provides Kruskal algorithm application that runs in Input Panel. The Panel is split into two areas: graph, messages areas (see Figure 20). The first one is a panel where the graph is displayed with user control over the graph and edit nodes and weighted edges. The second one is the messages area. A table, presenting the tree order, is displayed in this panel. A final result of Kruskal algorithm of vertices node is displayed.

5.7 Prim's algorithm

Like Kruskal algorithm, Prim algorithm is an algorithm in graph theory that finds a minimum spanning tree for a connected weighted graph. This means it finds a subset of the edges that forms a tree that includes every vertex where the total weight of all the edges in the tree is minimized [21]. The first step consists in choosing a vertex randomly and gathering an ensemble containing 1 vertex and 0 edges. Then, the minimal tree is built recursively in the following way: At stage n, edges have been already built a tree containing n vertex and n-1, we draw up the list of all the edges binding a vertex of the ensemble to a vertex which is not on the ensemble. Then we choose an edge of minimal weight, which add it in the tree. The tree contains now n+1 vertices and n edges. The algorithm finishes when all vertices of the graph are contained in the ensemble.

The implementation using an adjacency matrix graph representation and searching an array of weights to find the minimum weight edge to add requires $O(V^2)$ running time. Using a simple binary heap data structure and an adjacency list representation, Prim's algorithm can be shown to run in time $O(E \log(V))$ where E is the number of edges and V is the number of vertices. Using a more sophisticated Fibonacci heap, this can be brought down to `O(E + V log(V) )`, which is asymptotically faster [56].

SOMvisua provides Prim's algorithm application that runs in Input Panel. The Panel is split into two areas: graph, messages areas (see Figure 22). The first one is a panel where the graph is displayed with user control over the graph and edit nodes and weighted edges. The second one is the messages area. A table, presenting cost and path for each vertices, is displayed in this panel. A final result of Prim's algorithm of vertices node is displayed.

5.8 Connected Components algorithm

Connected component algorithm of an undirected graph is a sub-graph in which any two vertices are connected to each other by paths, and which is connected to no additional vertices in the super-graph [26]. For example, the graph shown in the illustration on the right has three connected components. A graph that is itself connected has exactly one connected component, consisting of the whole graph.

There are also efficient algorithms to dynamically track the connected components of a graph as vertices and edges are added, as a straightforward application of disjoint-set data structures. These algorithms require amortized `O(f(n))` time per operation, where adding vertices and edges and determining the connected component in which a vertex falls are both operations, and f(n) is a very slow-growing inverse of the very quickly growing Ackermann function [92]. A related problem is tracking connected components as all edges are deleted from a graph, one by one; an algorithm exists to solve this with constant time per query, and `O(V * E)` time to maintain the data structure; this is an amortized cost of `O(V)` per edge deletion. For forests, the cost can be reduced to `O(q + V log(V))`, or `O(log V)` amortized cost per edge deletion.

SOMvisua provides Connected Components algorithm application that runs in Input Panel. The Panel is split into two areas: graph, messages areas (see Figure 23). The first one is a panel where the graph is displayed with user control over the graph and edit nodes and weighted edges. The second one is the messages area. A list, presenting visiting report for each vertices, is displayed in this panel. A final result of Connected Components of vertices node is displayed.

5.9 Circled Bars Visualization

The Circled Bars visualization approach offers a simple method to answer questions like: “Which sentence produce similar concept to that of a selected sentence **A**?”. It thus takes a similarity list as input. Given a seed sentence **A**, an adjustable number of most similar sentences (according to the used similarity measure) are arranged in a circle. The sentences are ordered by their similarity to sentence **A**. The similarity values are visualized by filled arcs that vary in length and color corresponding to the applied color map. A Circled Bars visualization is generated from Google PageRank algorithm and the color map Fire is applied. Hence, the values in parentheses after the sentence names indicate the probability for the respective sentence to be found on a webpage that is known to mention the seed sentence **A**.

Since the Circled Bars visualization does not require high computing or graphics capabilities, it may serve as a user interface for small devices with limited screen size, like mobile phones or personal digital assistants.

5.10 Circled Fans Visualization

The Circled Fans visualization is a conceptual extension of the simple Circled Bars. While the Circled Bars only take the nearest neighbors of a given seed sentence (or any other entity) into account, the Circled Fans incorporate similarities in a transitive manner. Given a seed sentence **A** whose name is displayed in the center of the visualization, an adjustable number of most similar sentences are arranged in a circle around **A** and connected to **A** by edges whose thickness and color correspond to the similarities given by the similarity graph and the chosen colormap. The thicker the connecting edge, the more similar two sentences are. Subsequently, for each of the similar sentences of **A**, again, the most similar ones are selected, arranged in a circular arc whose center is the respective parent node, and connected to this parent node by an edge.

The user can adjust the maximum edge thickness, the maximum number of data items on the inner circle and on the outer circular arcs (which we call fans), as well as the angular extent of the fans. Moreover, the Circled Fans support user interaction by redrawing the visualization with a new seed sentence **B** whenever the user clicks on the label of an arbitrary sentence **B**. the Circled Fans visualization with the seed sentence Evanescence is depicted. as an example, an asymmetric similarity graph derived from Google PageRank is used to define sentence similarity, and the color-map Colorful is applied. Visualizing such asymmetric similarities is an important area of application of the Circled Fans. For example, the Circled Fans can reveal that specific sentence is mentioned on 53% of the article containing a given word, whereas the given word can only be found on 21% of the webpages that mention the specific sentence. Such information about similarity asymmetries can be used for measuring the popularity of an sentence and further for determining which sentences are prototypical for a specific context.

the Circled Fans visualization technique developed in [42] and integrated in the SOMvisua framework uses standard Euclidean geometry to visualize tree-like data structures as well as networks. In addition, the Circled Fans also support illustrating a similarity graph. In this case, the root element of the visualization corresponds to a sentence, represented by a source or destination of the similarity graph, the elements one level deeper in the hierarchy are the entities most similar to the root element, and the elements another level further down in the hierarchy are the nearest sentences to the sentences on the second level. When using a similarity graph as data source, this layout provides an easy means of differentiating between entities whose similarity relation is of a rather symmetric nature and entities with a rather unidirectional relationship.

Figure 27 illustrates the Circled Fans technique by depicting a visualization based on a similarity graph that has been created from Google PageRank algorithms of sentences in article. The root node (7), corresponding to a sentences in an article, is surrounded by its most similar sentence according to the probability-based similarity measure. Those sentence s are connected to the root via straight lines of different thickness and color according to the similarity values. Using size and color as a means of information encoding, in addition to the pure textual similarity values, further helps easily discerning the most important connections, i.e., those with highest similarity values.

From Figure 27 it can be seen that the most similar sentence to sentence node(7) is sentence node(6) with 0.3 similarity measure then followed by sentence node(5) came with 0.295. Moreover, the figure reveals some interesting information about the asymmetry of the similarity relations. Indeed, considering the third hierarchy level shows that all sentence nodes have a rather symmetric similarity relation to the root sentence node (7) as all have a variant similarity measure to the root in their set of most similar sentences. All neighbors of root sentence node(7) do link back to the root sentence node (7). The most apparent explanation for this fact is that all the sentences form a cluster of sentences in the same context article.

A shortcoming of the Circled Fans technique is the restriction of the visualization capabilities to three hierarchy levels due to the space limitations imposed by the use of Euclidean geometry. This is to some extent alleviated by the easy browsing facilities provided. The user can create a new view on the data simply by clicking on any label to bring the selected entity in the center position and thus use it as root node of the illustration.

5.11 Probabilistic Network Visualization

A Probabilistic Network visualization based on a similarity graph of concept sentences. Using this method, first, the vertices representing the data items are placed randomly on the screen. Then, an adaptation process that moves similar data items closer to each other is performed iteratively. Finally, edges between data items are drawn with a probability that is proportional to their similarity. The size of each vertex is calculated using the summed similarities between the data item represented by the vertex and all other data items. The label of a vertex is displayed when the mouse is moved over it.

Probabilistic Network visualization is a Graph-based visualization approach and also called node-link diagrams. Probabilistic Network visualization is scarcely used to illustrate relationships between text sentences entities. In [68] model similarity relations between artists as a similarity network and analyze some of its properties. In particular, they construct artist similarity graphs from two sources: expert opinions on similar artists extracted from playlist co-occurrences. this approach is taken and applied over text clustering visualization to utilize visual representation of relationships.

Another graph-based model is used in [47] to visualize an artist similarity matrix in a two-dimensional space. Each artist is represented by a vertex, and an edge between two vertices is drawn if the corresponding artist similarity is greater than a certain threshold. To position the vertices in a visually appealing manner, [47] use a physical spring model, where vertices are interpreted as metal rings and edges as springs. The respective distances are interpreted as repelling or attracting forces. The aim is then to bring the system in a state of low energy by minimizing an energy function.

Given a similarity graph as input, the Probabilistic Network algorithm first places the data items on random positions in the two-dimensional visualization space. To reorganize the positions of the artists in the output space, a simulated annealing approach [98], [99] is used. To this end, an iterative algorithm randomly selects two nodes and adjusts the distance between them in the output space to better fit the corresponding distance in the high-dimensional feature space. A time-dependent, linearly decreasing rate of adaptation is further used to gradually decrease the maximum distance correction. The reason for this is that high adaptations are necessary in the beginning due to the random initial placement of the nodes, whereas lower adaptations towards the end of the algorithm are intended to stabilize the system and should therefore only fine-tune the positions.

Figure 29 illustrates Probabilistic Network visualizations using as input an similarity Graph created via Google PageRank algorithm , see chapter 3 for more details .

5.12 Sunburst Visualization

The Sunburst as proposed in [79] and [72] is a circular, space-filling visualization technique for illustrating hierarchical data. It is sometimes also referred to as InterRing [61]. The center of the visualization represents the highest element in the hierarchy, whereas elements on deeper levels are illustrated by arcs further away from the center. Child elements are drawn within the angular borders of their parent, but at a more distant position from the center as shown in *Figure 30*.

Term occurrence graph is used to create a user interface that makes use of the well-established Sunburst visualization technique. Starting with the whole set of documents, more precisely, the graph that describes this set, the terms with the highest document frequency are selected and visualized as filled arcs around a centered circle (the root node) that represents the entire document collection. The size (angular extent) of each individual arc is proportional to the document frequency of the associated term, i.e. to the number of documents containing the term. Performing the term selection with respect to document frequencies recursively for all arcs eventually yields a complete Sunburst visualization. Internally, our project stores the Sunburst as a tree. Every arc A represents a set of documents that contain the term associated with A and the terms associated with all arcs that must be traversed on the shortest way to the root node.

The user can define a number of stop criteria to limit the calculation time, the size of the Sunburst, and the number of recursions. SOMvisua project provides the following constraints: maximum sub nodes per node, maximum depth of the tree, minimum angular extent of an individual arc. The font size of the labels, i.e. the terms, is automatically adapted to the angular extent of the arc. However, minimum and maximum values for the font size can be defined by the user.

Since the Sunburst interface is intended to be used for document search, users interaction is provided in two ways. First, clicking with the left mouse button on an arbitrary arc generates a new Sunburst visualization with this arc as root node, i.e. only the documents that are represented by the selected arc are used. Second, a right mouse click on any arc displays a pop-up menu with the locations of the documents represented by the selected arc. The user can then view a document by selecting it from the pop-up menu. the Sunburst interface generated from an graph of documents about the concept of a selected sentence. The values in parentheses indicate the document frequency. This sample visualization reveals which terms occur in a collection of documents about on concept. If the user wants to know, for example, in which documents two sentences are mentioned together, user can easily display a list of these documents by clicking on the respective arc.

In almost all scientific publications related to the Sunburst, its usual application scenario is browsing the hierarchical tree structure of a file system. In this scenario, directories and files are represented by arcs whose sizes are proportional to the sizes of the respective directories/files. In [36], the authors use the Sunburst technique to visualize frequent patterns for the purpose of analyzing the content of shopping carts. This application area is quite similar to our purpose of using this tool, in that the underlying co-occurrence data, from which the most important sets of terms according to a term weighting measure are extracted, can also be interpreted as frequent patterns.

The Sunburst offers the advantage of more clearly revealing the structure of the hierarchy as it displays all elements that reside on the same hierarchy level on the same torus, whereas all elements that reside on different hierarchy levels are depicted clearly visually separated on different trousers. As a result, the Sunburst frequently performs better in user studies on typical file/directory search and analysis tasks, both in time and correctness as in [67]. However, the principal weakness of the Sunburst is the very small arc size that makes small elements hardly perceivable.

Chapter 6
Experiment and Results

6.1 Introduction

In chapter 1 we identified the challenges related to building a graph representation for large scale text and visualization system using any of the content-based text similarity measures used in this thesis. Chapter 3, 4 and 5 subsequently presented the theoretical base to each of the component of SOMvisua. We showed how to use the text similarity features natively and correctly in SOM and GHSOM clustering algorithms, developed a method to alleviate the hub problem and created an visualizing solution for the reviewed class of text similarity algorithms. Now everything is in place to build a truly scalable text representation system. The framework developed alongside with the data we use for evaluation, is described in the next sections of this final chapter.

6.2 Selected text data set

To build the framework real text data is used. The data is fetched from an on-line Wikipedia web site, which offers free articles of texts for surfing purposes. Wikipedia's 30 million articles in 287 languages, including over 4.3 million in the English Wikipedia were estimated 365 million readers worldwide. Wikipedia seeks to create a summary of all human knowledge in the form of an online encyclopedia, with each topic of knowledge covered encyclopedically in one article.

Each article is assigned to one or more topics, which enables us to perform a large scale topic-evaluation experiment. In total there are 11 different text topic in the website. The distribution of the topics in the Wikipedia is shown in Figure 10 Almost one thirds of the albums are assigned to the Culture and the arts. A 2008 study conducted by researchers at Carnegie Mellon University and Palo Alto Research Center gave a distribution of topics as well as growth (from July 2006 to January 2008) in each field.

6.3 The framework: SOMvisua

To build the framework we first select the text similarity algorithm to use for extracting features, choose the parameters for the representing data as graph method to achieve best clustering results and evaluate the configuration according to the measures we have used throughout this thesis. After that we show how we implemented the framework, report query performance and show the graphical interface we built to allow easy visualizing of the selected text with the “SOMvisua”.

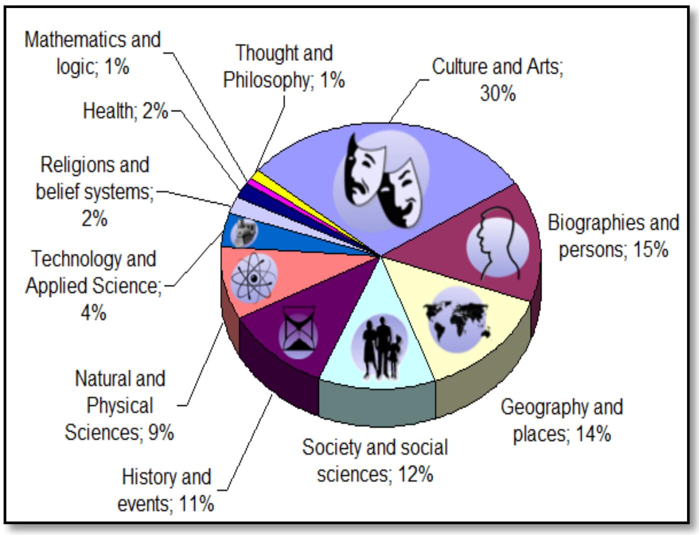


Figure 10: Pie chart of Wikipedia content by topic as of January 2008

6.3.1 Text Features Extraction

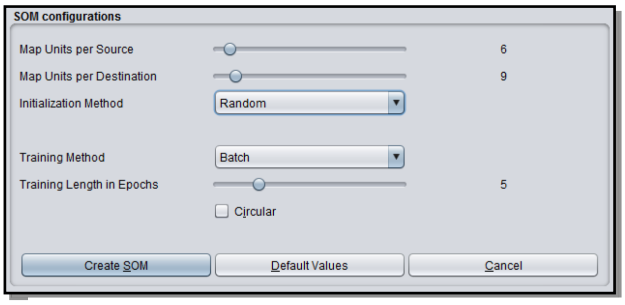
Duo to the nature of human language, throughout this thesis we have used a content-based text similarity algorithms to demonstrate the method and develop a mechanism to form graph data representation to be used as an input to SOM and GHSOM clustering algorithms (see Chapter 4 for their review). For our framework we decide to use Google pageRank algorithm. It is currently one of the text similarity algorithms with the best qualitative results.

6.3.2 Clustering Algorithm

The Google PageRank similarity function produces full representative similarity hierarchical graph. Chapter 3 shows that When PageRank is applied to all of the documents and the data is analyzed, the implications become much more interesting, for example PageRank values are an exceedingly accurate map of user behavior probability. Additionally SOM and GHSOM requires the similarity Matrix technique to linearly combine its two words and concept similarity measures. However with the Google pageRank method presented in Chapter 3 we have shown a way to use the complex similarity functions while still retrieving a very high percentage of concept representation. In Chapter 4, we showed how a SOM and GHSOM clustering can be done for the text similarity features. In Chapter 5 we have also presented similarity graph visualization with the ability to apply 8 famous graph algorithms. All of that enables us to finally use the SOMvisua with a wide variation.

6.3.3 Clustering Parameters

To use the SOM and GHSOM clustering algorithms two parameters need to be determined: the map units per Source and the map units per Destination for the approximate graph sources and destinations in SOM. Both parameters have a direct impact on the clustering quality and the speed of the system.



SOM configurations

Map Units per Source		6
Map Units per Destination		9
Initialization Method	Random	
Training Method	Batch	
Training Length in Epochs		5
	☐ Circular	

Figure 11: SOM configuration

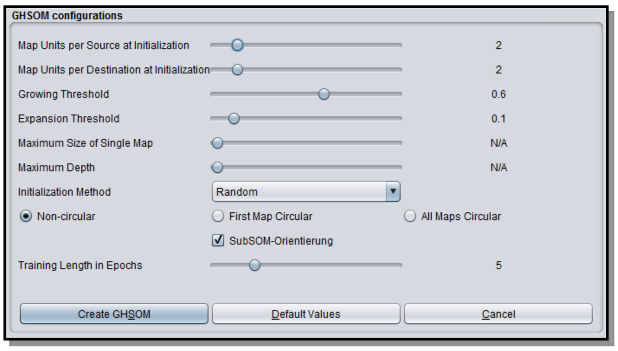
SOM configurations can be chosen via dialog shown in figure 11. In an experiment we evaluate how the approximate clustering method performs on the one million sentences using different parameter configurations. This experiment is the basis to choose the optimum set of parameters for the visualization system. To perform the experiment we randomly select 1000 sentences from the one million sentences, compute their exact 1–10 nearest neighbors in the whole article (of one million articles) and use the SOM and GHSOM methods with different parameter settings to measure the impact on the clustering quality, comparing it to a manual exact clustering. A larger part of the article could not be tested as a single computation of an exact arrangement takes about 15 seconds to complete. In the experiment the nearest neighbor is computed, measuring the percentage of true nearest neighbors compared to the exact solution.

6.3.4 Implementation and Specification

The implementation of the actual clustering and visualization system is now straightforward. In an initial step text document is analyzed and its similarity model is computed. All similarity models are allocated in memory. Graph data structure is built by Google pageRank. The number of node per graph depends on the actual number of words in the document. Google pageRank form and stores ranking object to link between nodes. It is good thing that English language has a limited set of words and Google pageRank comprehensively evaluate words. After the document has been analyzed, two additional preprocessing steps are required before the system can visualize:

- Creating SOM cluster by selecting SOM parameter manual. Then SOM enters a training phase to calculate epochs. Finally SOM-Grid is displayed.
- Creating GHSOM cluster over the previously formed SOM.

GHSOM configurations dialog will pop up as shown in figure 12 In this step more parameters has to be set. Alongside with map units per source and map units per destination, Growing and Extension Threshold should be adjusted.


GHSOM configurations

Map Units per Source at Initialization		2
Map Units per Destination at Initialization		2
Growing Threshold		0.6
Expansion Threshold		0.1
Maximum Size of Single Map		N/A
Maximum Depth		N/A
Initialization Method	Random	
[✓ Non-circular]	First Map Circular	All Maps Circular
	[✓ SubSOM-Orientierung]	
Training Length in Epochs		5

Figure 12: GHSOM configuration

6.4 Performance

As high performance is a key objective, and finally the actual cluster training times of the system is measured. The SOMvisua server system is implemented in Java programming Language and requires all similarity models and visualization to be loaded to Memory. In the case of the one million sentences 8 megabytes of memory are required to hold all text features in memory.

To run the system we used a desktop PC with a core i7 CPU and 32GB of RAM memory. Figure 13 compares the clustering execution times of five different SOM and GHSOM configurations. In table 2 we showed those configurations on the one million sentences article. From that plot we can see that the final system is capable of visualizing text clustering in maximum 0.29 seconds on a one million sentences article while returning about 94% of the correct nearest neighbors.

6.5 User Interface

We used the desktop application which we developed to build a compound text clustering and visualization framework on top of it. We integrated four functionalities on top of the SOMvisua:

- Text feature extraction method into similarity Graph data structure.
- Visualizing the similarity Graph with eight variant graph navigation algorithms.
- Perform SOM and GHSOM clustering for the similarity Graph with visualization of SOM-grid.
- Six graphical visualization of SOM and GHSOM clustering.

The desktop application starts by displaying a menu which is used to select a text document. A file chooser for selecting text document. Figure 14 shows a screenshot of the application using file chooser ready. Select text document and the Google PageRank algorithm will start to build similarity graph. Nodes construction will appear in the black messaging area. The text document that you chose will be displayed at right side text viewer area.

	Config 1	Config 2	Config 3	Config 4	Config 5
Map units per Source	2	4	6	8	10
Map units per destination	2	4	6	8	10
Growing threshold	0.2	0.4	0.6	0.8	1.0
Expanding threshold	0.2	0.4	0.6	0.8	1.0
Max size of single Map	20	40	60	80	100
Max depth	20	40	60	80	100
circularity	Non-circular	All maps	first	All maps	first

Table 2: five different SOM and GHSOM configuration for testing SOMvisua

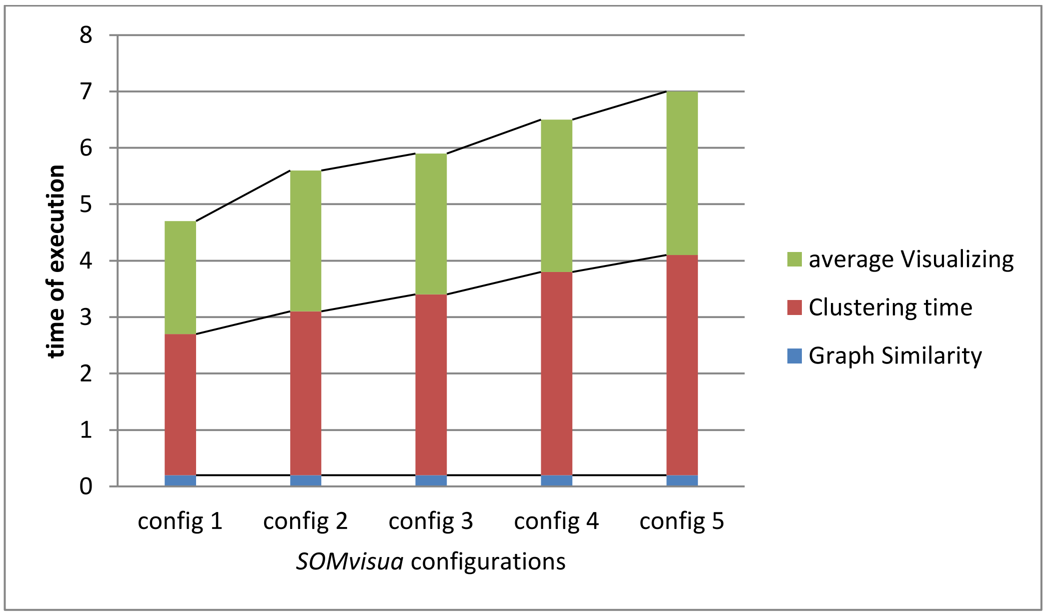
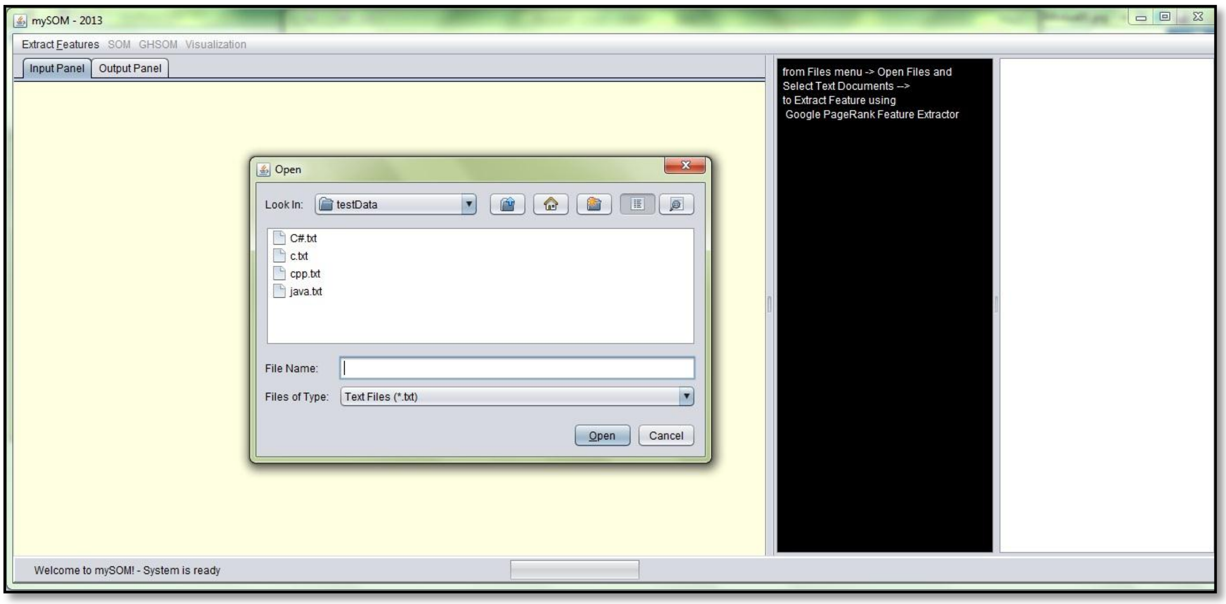


Figure 13: Comparison of the time it takes to execute the one million sentences article for nearest neighbors using five variant configurations.



from Files menu -> Open Files and
Select Text Documents -->
to Extract Feature using
Google PageRank Feature Extractor

Welcome to mySOM! - System is ready

Figure 14: Screenshot of the SOMvisua framework text clustering and visualization. The file chooser is ready to select text document and start.

6.5.1 Similarity Graph Visualization

Similarity Graph can be viewed in three forms. First , in view logical in which graph is drown as vertices and edges like in figure 15 (a). another way to represent similarity graph is adjacency List as shown in figure 15 (b). the last representation for similarity graph is adjacency matrix as shown in figure 15 (c). From Extract Features menu you can choose view input Graph to start constructing eight tab panels. Figure 16 shows similarity graph visualized as nodes and vertices and ready to be test. SOMvisua provides a visual animation for algorithms execution with animation speed control. In the bottom of figure 16 a slide bar starting with slow to fast to control animation speed.

Figure 16 shows Dijksrta's algorithm running to compute the shortest path to Node 5. In *Figure 17* Topological sort algorithm is sorting the similarity graph. Breath First algorithm is illustrated in *figure 18*. you can choose between directed graph or undirected graph. The results of the algorithm will be viewed on the graph. In Depth First Search algorithm, graph nodes will be sorted according to its depth as shown in Figure 19. For a graph analysis algorithm for finding shortest paths in a weighted graph with positive or negative edge weights like Floyd-Warsall algorithms shown in Figure 20. Finding a minimum spanning tree for a connected weighted graph like in kruskal's algorithm as Figure 21 and prim's algorithm as figure 22. the Connected component graph algorithm for connecting and labeling is shown in figure 23.

6.5.2 SOM and GHSOM clustering

To execute SOM clustering algorithm, the user can select "create SOM" submenu from SOM menu. SOM configurations dialog will appear for the user to choose preferred configurations. SOM-Grid will be appeared in "SOM" tab as a sub tab of "output panel" tab as shown in figure 24. "SOM" menu offers "save SOM" submenu to save SOM in a file to be loaded later on from "load SOM" submenu. Another submenu "Export SOM to HTML" is to generate HTML file contains SOM-Grid. All previously mentioned options about SOM can be applied to GHSOM clustering algorithm using "GHSOM" menu. GHSOM-Grid we look like figure 25.

6.5.3 Clustering Visualization

"visualization" menu contains five submenu for clustering visualization. circled bar basic and advance , circled Fans , Probabilistic network and Sun Burst visualization are shown in figures 26 ,27,28 ,29 and 30 respectively.

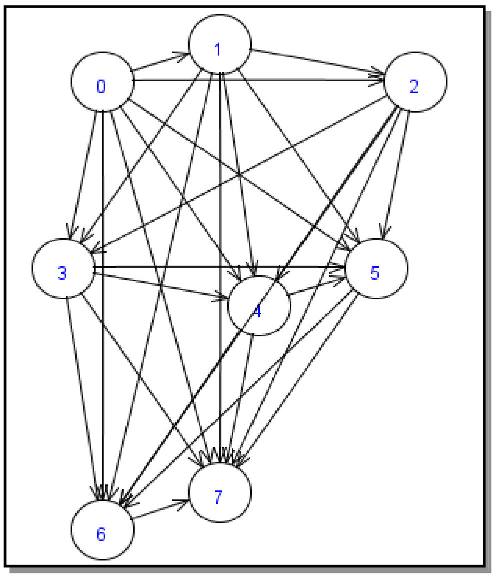


Figure 15 (a):logical graph representation.

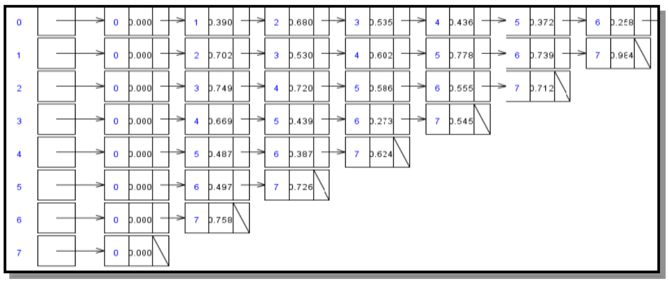
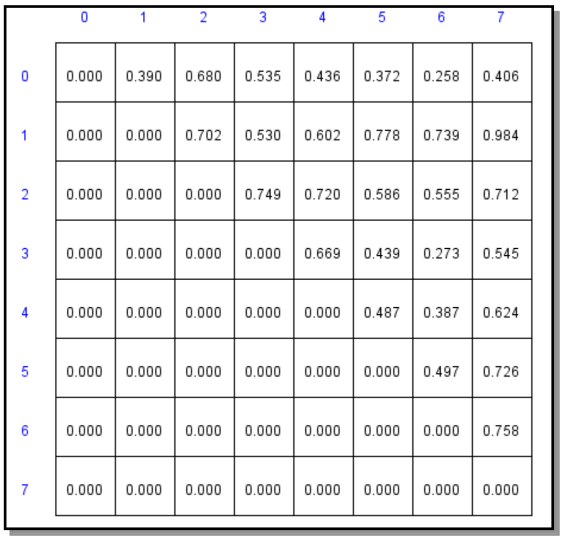
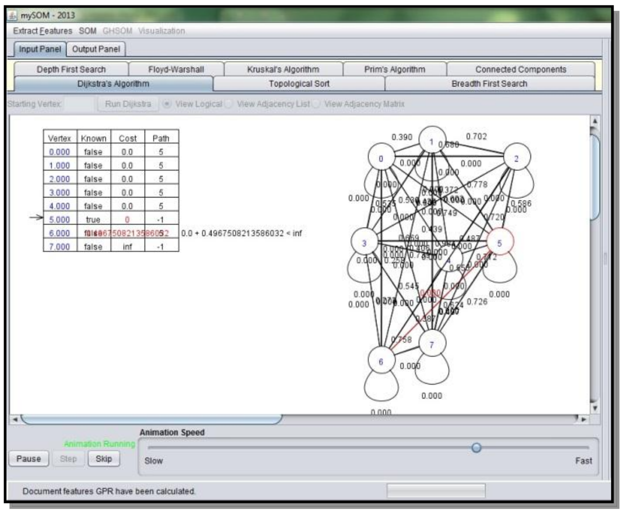


Figure 15 (b):adjacency List representation.



	0	1	2	3	4	5	6	7
0	0.000	0.390	0.680	0.535	0.436	0.372	0.258	0.406
1	0.000	0.000	0.702	0.530	0.602	0.778	0.739	0.984
2	0.000	0.000	0.000	0.749	0.720	0.586	0.555	0.712
3	0.000	0.000	0.000	0.000	0.669	0.439	0.273	0.545
4	0.000	0.000	0.000	0.000	0.000	0.487	0.387	0.624
5	0.000	0.000	0.000	0.000	0.000	0.000	0.497	0.726
6	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.758
7	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Figure 15 (c):adjacency Matrix representation.

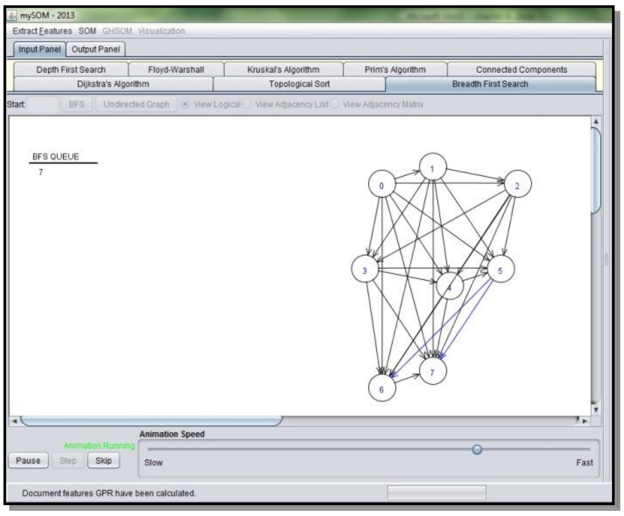

mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
Depth First Search Floyd-Warshall Kruskal's Algorithm Prim's Algorithm Connected Components
Dijkstra's Algorithm Topological Sort Breadth First Search
Starting Vertex: Run Dijkstra View Logical View Adjacency List View Adjacency Matrix
Animation Running
Animation Speed
Pause Step Skip
Slow Fast

Vertex	Known	Cost	Path
0.000	false	0.0	5
1.000	false	0.0	5
2.000	false	0.0	5
3.000	false	0.0	5
4.000	false	0.0	5
5.000	true	0	-1
6.000	false	0.4967508213586032	5
7.000	false	inf	-1

0.0 + 0.4967508213586032 < inf

Document features GPR have been calculated.

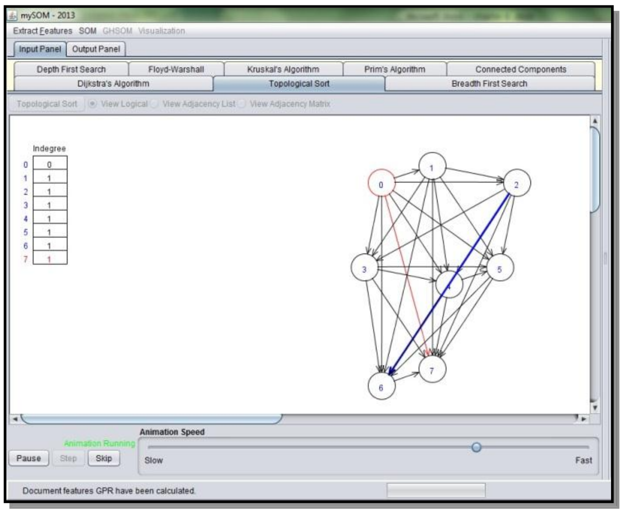
Figure 16:Dijksrta's algorithm


mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
Depth First Search Floyd-Warshall Kruskal's Algorithm Prim's Algorithm Connected Components
Dijkstra's Algorithm Topological Sort Breadth First Search
Start: BFS Undirected Graph View Logical View Adjacency List View Adjacency Matrix
Animation Running
Animation Speed
Pause Step Skip
Slow Fast

BFS QUEUE
7

Document features GPR have been calculated.

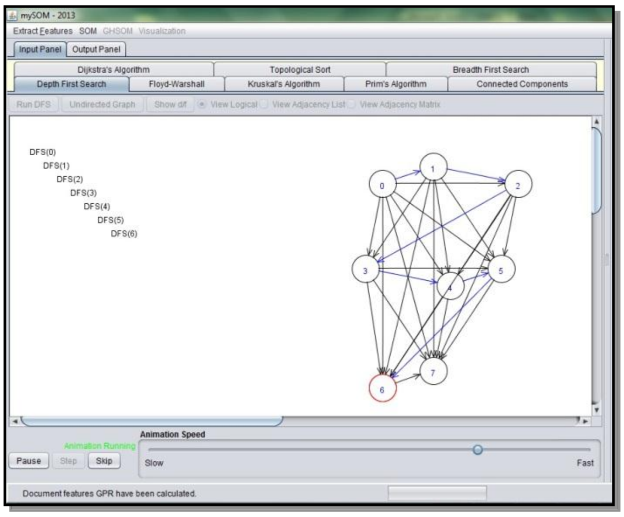
Figure 18:Breath First Search algorithm


mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
Depth First Search Floyd-Warshall Kruskal's Algorithm Prim's Algorithm Connected Components
Dijkstra's Algorithm Topological Sort Breadth First Search
Topological Sort View Logical View Adjacency List View Adjacency Matrix
Animation Running
Animation Speed
Pause Step Skip
Slow Fast

	Indegree
0	0
1	1
2	1
3	1
4	1
5	1
6	1
7	1

Document features GPR have been calculated.

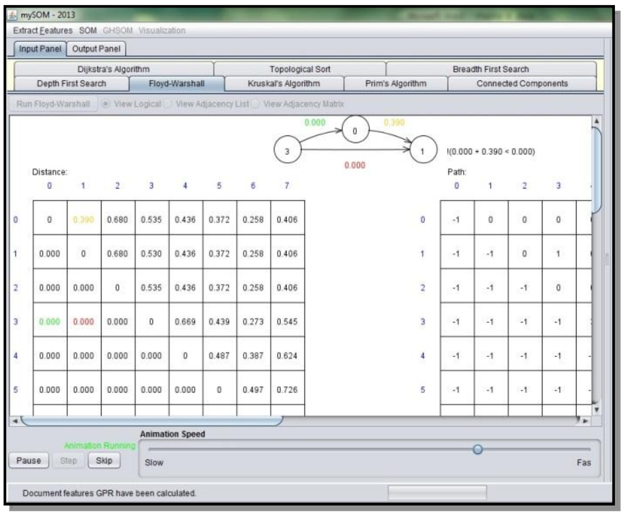
Figure 17:Topological sort algorithm


mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
Dijkstra's Algorithm Topological Sort Breadth First Search
Depth First Search Floyd-Warshall Kruskal's Algorithm Prim's Algorithm Connected Components
Run DFS Undirected Graph Show dif View Logical View Adjacency List View Adjacency Matrix
Animation Speed Animation Running Pause Step Skip Slow Fast

```
DFS(0)
  DFS(1)
    DFS(2)
      DFS(3)
        DFS(4)
          DFS(5)
            DFS(6)
```

Document features GPR have been calculated.

Figure 19:Depth First Search algorithm


mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
Run Floyd-Warshall View Logical View Adjacency List View Adjacency Matrix
0.000
0.390
0.000
Animation Speed Animation Running Pause Step Skip Slow Fas

!(0.000 + 0.390 < 0.000)

Distance:

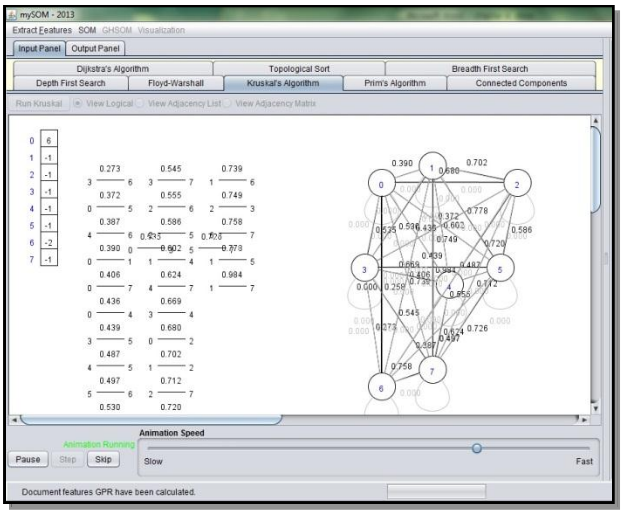
	0	1	2	3	4	5	6	7
0	0	0.390	0.680	0.535	0.436	0.372	0.258	0.406
1	0.000	0	0.680	0.530	0.436	0.372	0.258	0.406
2	0.000	0.000	0	0.535	0.436	0.372	0.258	0.406
3	0.000	0.000	0.000	0	0.669	0.439	0.273	0.545
4	0.000	0.000	0.000	0.000	0	0.487	0.387	0.624
5	0.000	0.000	0.000	0.000	0.000	0	0.497	0.726

Path:

	0	1	2	3
0	-1	0	0	0
1	-1	-1	0	1
2	-1	-1	-1	0
3	-1	-1	-1	-1
4	-1	-1	-1	-1
5	-1	-1	-1	-1

Document features GPR have been calculated.

Figure 20:Floyd-Warsall algorithm

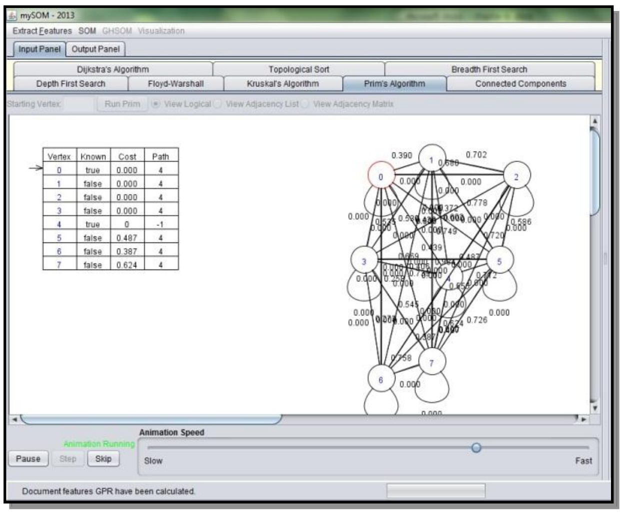

mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
Run Kruskal View Logical View Adjacency List View Adjacency Matrix
Animation Speed Animation Running Pause Step Skip Slow Fast

0	6
1	-1
2	-1
3	-1
4	-1
5	-1
6	-2
7	-1

```
   0.273        0.545        0.739
3 ——— 6     3 ——— 7     1 ——— 6
   0.372        0.555        0.749
0 ——— 5     2 ——— 6     2 ——— 3
   0.387        0.586        0.758
4 ——— 6     2 ——— 5     6 ——— 7
   0.390        0.602        0.778
0 ——— 1     1 ——— 4     1 ——— 5
   0.406        0.624        0.984
0 ——— 7     4 ——— 7     1 ——— 7
   0.436        0.669
0 ——— 4     3 ——— 4
   0.439        0.680
3 ——— 5     0 ——— 2
   0.487        0.702
4 ——— 5     1 ——— 2
   0.497        0.712
5 ——— 6     2 ——— 7
   0.530        0.720
```

Document features GPR have been calculated.

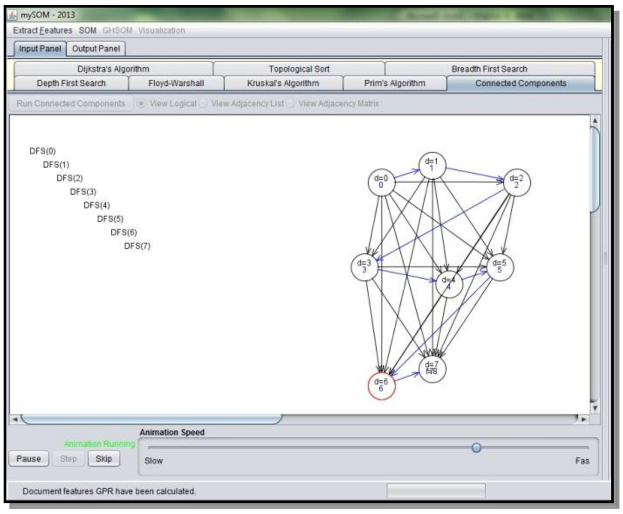
Figure 21:kruskal's algorithm


mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
Starting Vertex: Run Prim View Logical View Adjacency List View Adjacency Matrix
0.390
0.702
Animation Speed Animation Running Pause Step Skip Slow Fast

Vertex	Known	Cost	Path
0	true	0.000	4
1	false	0.000	4
2	false	0.000	4
3	false	0.000	4
4	true	0	-1
5	false	0.487	4
6	false	0.387	4
7	false	0.624	4

Document features GPR have been calculated.

Figure 22:Prim's algorithm


mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
Run Connected Components View Logical View Adjacency List View Adjacency Matrix
d=0
d=1
d=2
d=3
d=4
d=5
d=6
d=7
Animation Speed Animation Running Pause Step Skip Slow Fas

```
DFS(0)
  DFS(1)
    DFS(2)
      DFS(3)
        DFS(4)
          DFS(5)
            DFS(6)
              DFS(7)
```

Document features GPR have been calculated.

Figure 23: Connected Components algorithm

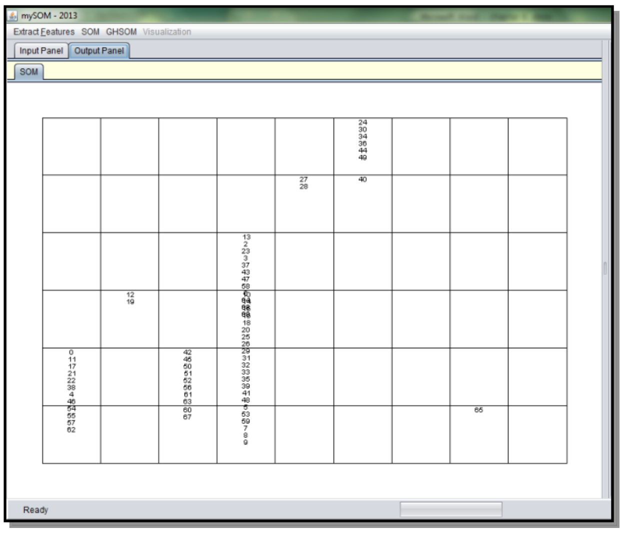
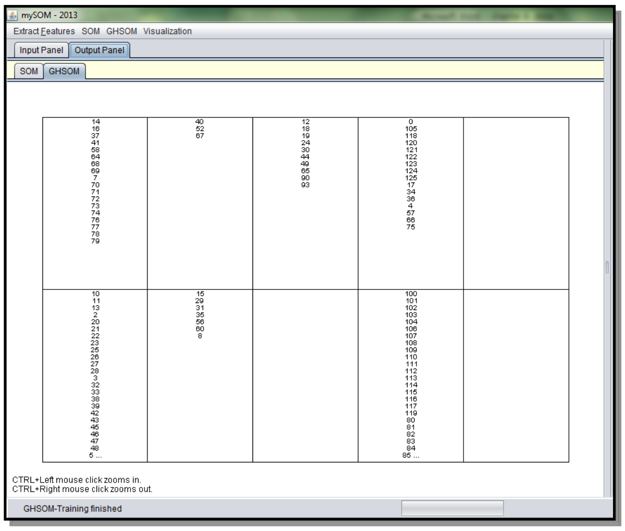


Figure 24: SOM-Grid preview

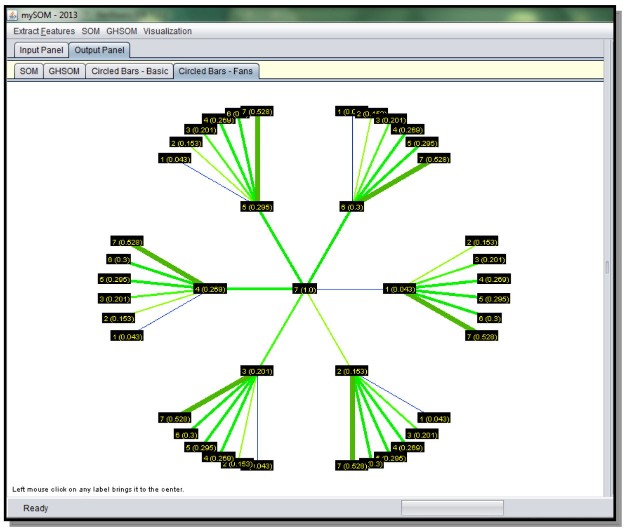


14 16 37 41 58 64 68 69 7 70 71 72 73 74 76 77 78 79	40 52 67	12 18 19 24 30 44 49 65 90 93	0 105 118 120 121 122 123 124 125 17 34 36 4 57 66 75	
10 11 13 2 20 21 22 23 25 26 27 28 3 32 33 38 39 42 43 45 46 47 48 5 ...	15 29 31 35 56 60 8		100 101 102 103 104 106 107 108 109 110 111 112 113 114 115 116 117 119 80 81 82 83 84 85 ...	

CTRL+Left mouse click zooms in.
CTRL+Right mouse click zooms out.

GHSOM-Training finished

Figure 25:GHSOM-Grid preview


mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
SOM GHSOM Circled Bars - Basic Circled Bars - Fans
7 (1.0)
Ready

Left mouse click on any label brings it to the center.

Figure 28: Circled Fans visualization

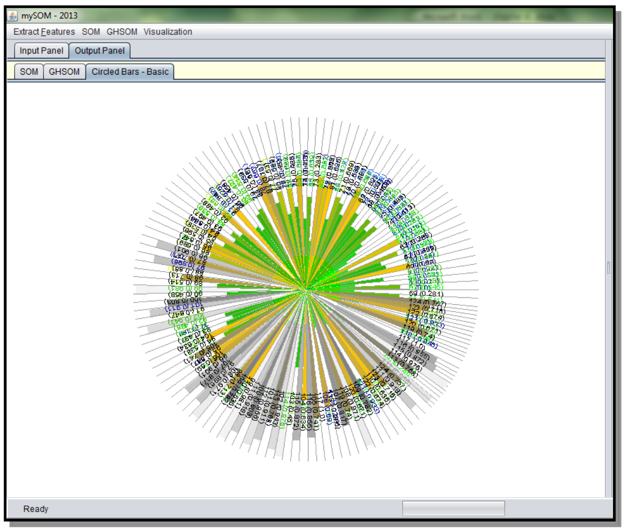
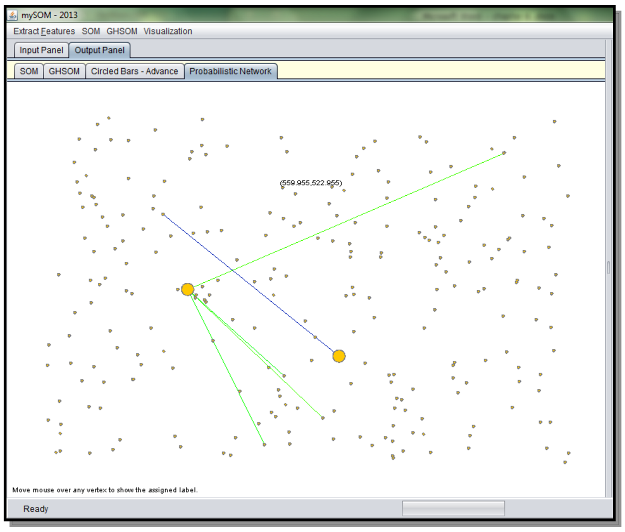

mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
SOM GHSOM Circled Bars - Basic
Ready

Figure 26:Circled Bars – Basic visualization


mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
SOM GHSOM Circled Bars - Advance Probabilistic Network
(559.955,522.955)
Ready

Move mouse over any vertex to show the assigned label.

Figure 29: Probabilistic network visualization

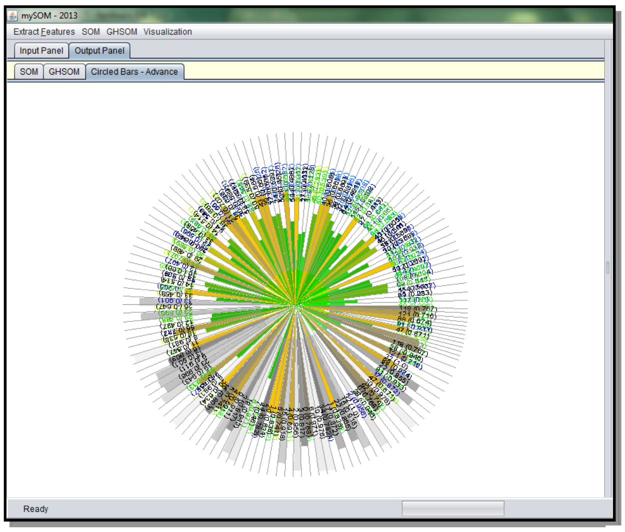

mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
SOM GHSOM Circled Bars - Advance
Ready

Figure 27:Circled Bars – advance visualization

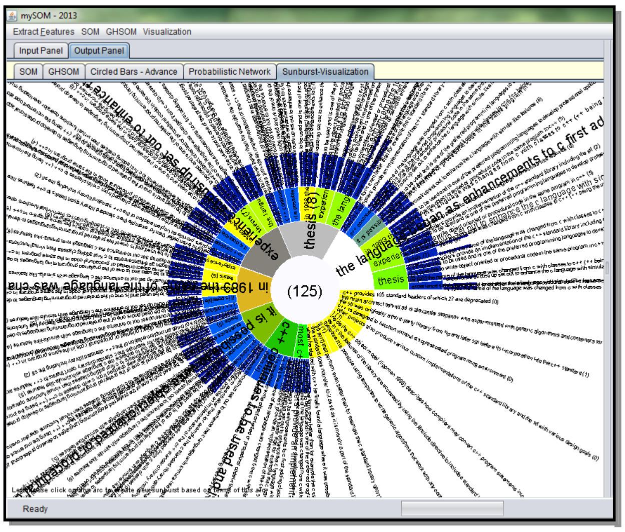

mySOM - 2013
Extract Features SOM GHSOM Visualization
Input Panel Output Panel
SOM GHSOM Circled Bars - Advance Probabilistic Network Sunburst-Visualization
(125)
thesis
the language
Ready

Figure 30: SunBurst visualization

Conclusion

visualization usually allows a faster data exploration and often provides better results, especially in cases where automatic algorithms fail. In addition, visualization techniques provide a much higher degree of confidence in the findings of the exploration. This fact leads to a high demand for visualization techniques and makes them indispensable in conjunction with automatic clustering techniques.

In the age of information explosion and knowledge revolution, information surrounds people everywhere. Pieces of information of the same context should be categorized into groups. By classifying information into groups, four major benefits could be acquired. Classification sums up the ideas and summarizes the context to form organized knowledge structure. This is what is called data clustering.

We have designed and implemented a new framework called the SOMvisua for clustering sets of sentences of full-text article that is available in electronic form. The SOMvisua appears to be especially suitable for visualization tasks in which the user has a vague idea of the contents of the text article being examined. With the SOMvisua, the sentences are ordered meaningfully on a graph map according to their contents. Graph representation helps the visualization by giving an overall view of what the information space looks like.

We showed how to use the text similarity features natively and correctly in SOM and GHSOM clustering algorithms, developed a method to alleviate the hub problem and created an visualizing solution for the reviewed class of text similarity algorithms. SOMvisua provides a visual animation for eight famous graph algorithms execution with animation speed control. SOMvisua provides a visual animation for eight famous graph algorithms execution with animation speed control. SOMvisua presents six types of visualization.

References

[1] D. Girardi, M. Giretzlehner and J. Küng, "Using Generic Meta-Data-Models for Clustering Medical Data", Lecture Notes in Computer Science, Springer Berlin Heidelberg ISBN 978-3-642-32394-2,Vol. 7451,page(s) 40-53, March 2012.

[2] M. Zribi, Y. Boujelbene, I. Abdelkafi and R. Feki, , "The self-organizing maps of Kohonen in the medical classification", International Conference Sciences of Electronics, Technologies of Information and Telecommunications ,Page(s) 852 - 856, March 2012

[3] S. Zha,"Multiple Spheres SOM", Australian National University annual IT conference ,May 2012.

[4] L. Zhao, H. Xu, J. Li and Q. Cai,"A Kind of Map Coloring Algorithm Considering of Depth First Search", the 2012 International Conference on Industrial Control and Electronics Engineering, Page(s) 648-651, August 2012

[5] A. Shklovets and N. Axak, "Visualization of high-dimensional data using two-dimensional self-organizing piecewise-smooth Kohonen maps", Optical Memory and Neural Networks Journal, vol. 21, Page(s) 227-232 ,October 2012.

[6] Q. Yu, Z. Miao, G. Wu and Y. Wei, "Lumping algorithms for computing Google's PageRank and its derivative, with attention to unreferenced nodes", Information Retrieval Journal, Pages 503-526,vol. 15 Issue 6, December 2012

[7] M. Naoghare and V. Deshmukh,"Comparison of parallel genetic algorithm with depth first search algorithm for solving verbal arithmetic problems", the International Conference & Workshop on Emerging Trends in Technology, Pages 324-329, February 2011

[8] G. Vaira and O. Kurasova, "Parallel Bidirectional Dijkstra's Shortest Path Algorithm", the 2011 conference on Databases and Information Systems VI: Selected Papers from the Ninth International Baltic Conference, Page(s): 422-435 ,August 2011

[9] Urs Bergmann and C. von der Malsburg,"Self-Organization of Topographic Bilinear Networks for Invariant Recognition", Neural Computation, vol. 23,Page(s) 2770-2797,November 2011

[10] N. Broutin, L. Devroye and E. McLeish,"Note on the Structure of Kruskal's Algorithm",Springer-Verlag New York, Inc.,Volume 56 Issue 2,Pages 141-159 ,February 2010

[11] K. Tasdemir , "Graph Based Representations of Density Distribution and Distances for Self-Organizing Maps", Neural Networks, IEEE Transactions , March 2010.

[12] S. Hougardy,"The Floyd-Warshall algorithm on graphs with negative cycles", Information Processing Letters , Volume 110, Issue 8-9,April 2010

[13] C. Mouratidis, G. Majchrowska, D. Zissopoulos and N. Asimopoulos, "Modified Floyd-Warshall algorithm for risk arbitrage", the 14th WSEAS international conference on Systems: part of the 14th WSEAS CSCC multiconference - Volume I, July 2010

[14] M. Hassaan, M. Burtscher and KeshavPingali, "Ordered and unordered algorithms for parallel breadth first search", the 19th international conference on Parallel architectures and compilation techniques ,Pages 539-540 ,September 2010

[15] C. Jiang, F. Coenen and M. Zito,"Frequent sub-graph mining on edge weighted graphs", the 12th international conference on Data warehousing and knowledge discovery, Page(s) 77-88 ,August 2010

[16] M. Muskulus, "Applications of Page Ranking in P Systems",Book Membrane Computing, Page(s) 311 - 324 ,January 2009

[17] E. López-Rubio, J. M. Ortiz-de-Lazcano-Lobato, and M. C. Vargas- González, "Probabilistic self-organizing graphs", 10th International Work-Conference Artificial Neural Networks, vol. 1, Page(s) 180 – 187, October 2009.

[18] P. Chebolu, P. Melsted, "PageRank and the random surfer model", the 9th annual ACM-SIAM symposium on Discrete algorithms, Page(s) 1010-1018,January 2008

[19] D. Phuc and M. X. Hung, "Using SOM based Graph Clustering for Extracting Main Ideas from Documents", Research, Innovation and Vision for the Future International IEEE Conference Page(s) 209 - 214,July 2008

[20] F. Hussin, M. Farra and Y. Sonbaty, "Extending the Growing Hierarchal SOM for Clustering Documents in Graphs domain", International Joint Conference on Neural Networks, Pp. 4028–4035,June 2008.

[21] W. Chang, Y. Chiu and M. Li,"LearningKruskal's Algorithm, Prim's Algorithm and Dijkstra's Algorithm by Board Game",the 7th international conference on Advances in Web Based Learning, Pages 275 - 284,August 2008

[22] K. Taşdemir and E. Merényi ,"Cluster Analysis in Remote Sensing Spectral Imagery through Graph Representation and Advanced SOM Visualization", Springer Berlin Heidelberg ISBN: 978-3-540-88410-1 ,Lecture Notes in Computer Science, Vol. 5255,Page(s) 259 - 271 ,October 2008.

[23] K. Taşdemir ,"Exploring Topology Preservation of SOMs with a Graph Based Visualization", International Conference on Intelligent Data Engineering and Automated Learning, Vol. 5326,Page(s) 180 - 187, November 2008

[24] R. Möhring, H. Schilling, B. Schütz, D. Wagner and T. Willhalm,"Partitioning graphs to speedup Dijkstra's algorithm", Journal of Experimental Algorithmic (JEA) , vol. 11,February 2007

[25] D. Pearce and P. Kelly, "A dynamic topological sort algorithm for directed acyclic graphs", Journal of Experimental Algorithmic (JEA), Article No. 1.7 , Volume 11, February 2007

[26] R. Yapa and H. Koichi, "A connected component labeling algorithm for grayscale images and application of the algorithm on mammograms", the 2007 ACM symposium on Applied computing, Page(s) 146-152 ,March 2007

[27] H. Chim and X. Deng, "A New Suffix Tree Similarity Measure for Document Clustering", the 16th international conference on World Wide Web, Page(s) 121-130,May 2007

[28] U. Yun, "WIS: Weighted Interesting Sequential Pattern Mining with a Similar Level of Support and/or Weight". ETRI Journal, vol. 29, No. 3, Page(s): 336-352, June 2007.

[29] K. Tasdemir and E. Merényi, "A new cluster validity index for prototype based clustering algorithms based on inter- and intra-cluster density", International Joint Conference on Neural Networks, Page(s) 2205–2211, August 2007.

[30] U. Yun, and J. Leggett, "WIP: Mining Weighted Interesting Patterns with a Strong Weight and/or Support Affinity". The 6th SIAM International Conference on Data Mining, September, 2007

[31] P. Guttoski, M. Sunye and F. Silva, "Kruskal's Algorithm for Query Tree Optimization",the 11th International Database Engineering and Applications Symposium,Pages 296-302,September 2007

[32] B. Milic and M. Malek ,"Adaptation of the breadth first search algorithm for cut-edge detection in wireless multihopnetworks",the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems, Pages 377-386 ,October 2007

[33] P. Geibel, U. Krumnack, O. Pustylnikov, A. Mehler , H. Gust and K. Kühnberger, "Structure-Sensitive Learning of Text Types", vol. 4830, Page(s) 642-646 ,December 2007.

[34] A. Langville and C. Meyer,"Google's PageRank and Beyond: The Science of Search Engine Rankings", Princeton University Press Princeton, ISBN:0691122024,July 2006

[35] A. Markov, M. Last, and A. Kandel, "Fast Categorization of Web Documents represented by Graphs", the Advances in Web Mining and Web Usage Analysis, vol 4811, Page(s): 56-71, August 2006

[36] D. Keim, F. Mansmann, J. Schneidewind and T. Schreck, "Monitoring Network Traffic with Radial Traffic Analyzer", IEEE Symposium Visual Analytics Science And Technology, Page(s):123 - 128, October 2006

[37] A. Markov, M. Last, "Efficient Graph-based Representation of Web Documents", the 3rd International Workshop on Mining Graphs, Trees and Sequences, Page(s): 52-62, April 2005.

[38] U. Yun, and J. Leggett, "WFIM: Weighted Frequent Item set Mining with a Weight Range and a Minimum Weight", the 5th SIAM International Conference on Data Mining, Page(s): 636-640, April 2005.

[39] A. Schenker, H. Bunke, A. Kandel and M. Last , "Graph Theorectic Techniques for Web Content Mining", Series in Machine Perception and Artificial Intelligence, ISBN: 978-981-256-339-2 , May 2005.

[40] A. Rauber, E. Pampalk, and D. Merkl. "The SOM enhanced Juke Box: Organization and visualization of music collections based on perceptual models", Journal of New Music Research, vol. 32, Pages:193–210, May 2003.

[41] W. Wang, D. Do, and X. Lin, "Term GraphModel for Text classification", the Advanced Data Mining and Applications, Page(s): 19-30, July 2005

[42] M. Schedl, P. Knees, and G. Widmer." Using CoMIRVA for Visualizing Similarities Between Music Artists". the 16th IEEE Visualization 2005 Conference , October 2005.

[43] K. Gee, and D. Cook, "Text Classification Using Graph-Encoded Linguistic Elements", FLAIRS Conference, Page(s): 487-492, October 2005

[44] E. Koutsomitopoulou and D. Loritz, "A neural network model for the representation of natural language", Doctoral Dissertation - Georgetown University, January 2004

[45] K. Simunic, J. Novak, "Combining visualization and interactive clustering for exploring large document pools" , the 4th IASTED International Conference on Visualization, Imaging, and Image Processing , April 2004

[46] J. Tomita, H. Nakawatase and M. Ishii, "Graph based text database for Knowledge discovery", IEEE International Conference on World Wide Web (WWW),Page(s) 454-455, May 2004

[47] F. Vignoli, R. van Gulik and H. van de Wetering, "Mapping Music in the Palm of Your Hand, Explore and Discover Your Collection". the 5th International Symposium on Music Information Retrieval, September 2004.

[48] K. Hammouda and M. Kame,"Efficient phrase-based document indexing for web document clustering". IEEE Transactions on Knowledge and Data Engineering, Vol. 16,Page(s): 1279 - 1296 ,October. 2004

[49] Y. Chi, S. Nijssen, R. Muntz, and J. Kok, "Frequent Subgree Mining An Overview", Fundamental Informatics, Special Issue on Graph and Tree Mining, vol. 66, Page(s):161-198, November 2004

[50] A. Schenker, M. Last, H. Bunke and A. Kandel , "Classification of Web documents using a graph model", 7th International Conference of document analysis and Recognition ICDAR, vol. 1,Page(s) 240 , August 2003.

[51] F. Tao, F. Murtagh, and M. Farid,"Weighted Association Rule Mining using Weighted Support and Significance Framework" . the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 2003.

[52] I. Connell, T. Green, and A. Blandford, "Ontological Sketch Models: highlighting user-system misfits", Human Computer Interaction (HCI), Page(s): 163-178, September 2003

[53] S. Dixon, E. Pampalk, and G. Widmer. "Classification of Dance Music by Periodicity Patterns". the 4th International Conference on Music Information Retrieval , Vol. 29,pages 159–166,October 2003.

[54] J. Huan, W. Wang, and J. Prins, "Efficient Mining of Frequent Subgraph in the Presence of Isomorphism". the 3th IEEE International Conference on Data Mining, Page(s): 549 - 552 ,November 2003.

[55] H. Yin, "ViSOM- a novel method for multivariate data projection and structure visualization", IEEE Transactions on Neural Networks, vol. 13,Page(s). 237–243, January 2002.

[56] C. Martel, "The expected complexity of Prim's minimum spanning tree algorithm", Information Processing Letters , Volume 81, Issue 4,Page(s) 197 - 201,February 2002

[57] R. Gruen and T. Kubota, , "A neural network approach to system performance analysis", IEEE Digital Object Identifier , Page(s) 349 - 354,April 2002.

[58] M. Notess and N. Minibayeva, "Variations2: Toward Visual Interfaces for Digital Music Libraries",the 2nd International Workshop on Visual Interfaces to Digital Libraries, July 2002.

[59] E. Pampalk, A. Rauber, and D. Merkl, "Using smoothed data histograms for cluster visualization in self-organizing maps", International Work-Conference of Artificial Neural Networks, Page(s) 871–876, August 2002

[60] J. Aucouturier and F. Pachet. "Music Similarity Measures: What's the Use?". the 3rd International Symposium on Music Information Retrieval , pages 157-163 , October 2002.

[61] J. Yang , M. Ward and E. Rundensteiner."InterRing: An Interactive Tool for Visually Navigating and Manipulating Hierarchical Structures". the IEEE Symposium on Information Visualization. Page(s):77 - 84, October 2002

[62] P. Cano, M. Kaltenbrunner, F. Gouyon, and E. Batlle. "On the Use of FastMap for Audio

Information Retrieval and Browsing", the International Conference on Music Information Retrieval , October 2002.

[63] B. Logan, "Content-Based Playlist Generation: Exploratory Experiments", the International Conference on Music Information Retrieval (ISMIR 2002), October 2002.

[64] S. Pauws and B. Eggen. "PATS: Realization and User Evaluation of an Automatic Playlist Generator", the International Conference on Music Information Retrieval (ISMIR 2002), Paris, France, October 2002.

[65] X. Yan and J.Han, "gSpan: Graph-based Substructure Pattern Mining". The 2nd IEEE International Conference on Data Mining, Page(s): 721 - 724 ,December 2002

[66] M.-C. Su and H.-T. Chang, "A new model of self-organizing neural networks and its applications",IEEE Transactions on Neural Networks, vol. 12, Page(s) 153–158, January 2001.

[67] B. Logan and A. Salomon, "A Music Similarity Function Based on Signal Analysis", the IEEE International Conference on Multimedia and Expo, Page(s): 745 - 748 ,August 2001

[68] B. Choudhary and P. Bhattacharyya, "Text Clustering Using Universal Networking Language", Universal Networking Language Conference, vol. 16 , Page(s) 22-36, November 2001.

[69] M. Kuramochi, and G. Karypis, "Frequent Subgraph Discovery", IEEE International Conference on Data Mining, Page(s): 313-320 , November 2001

[70] G. Tzanetakis and P. Cook. "Marsyas3D: A Prototype Audio Browser-Editor using a Large Scale Immersive Visual Audio Display", the International Conference on Auditory Display, November 2001.

[71] A. Becks, S. Sklorz and M. Jarke, "A Modular Approach for Exploring the Semantic Structure of Technical Document Collection", the International Working Conference on Advanced Visual Interfaces, Page(s): 298-301,May 2000

[72] J. Stasko and E. Zhang. "Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations". the 6th IEEE Symposium on Information Visualization, pages 57–65, October 2000

[73] J. Huan, W. Wang, and J. Prins, "Efficient Mining of Frequent Subgraph in the Presence of Isomorphism". the 3th IEEE International Conference on Data Mining, Page(s): 549 - 552 ,November 2003.

[74] A. Ben-Dor, R. Shamir and Z. Yakhini, "Clustering gene expression patterns", Journal of Computational Biology,Page(s) 33-42 , April 1999

[75] S. Kaski , T. Honkela , K. Lagus , and L. Kohonen , "WEBSOM self-organizing maps of document collections",Neurocomputing, vol. 21, May 1998

[76] C. Bishop, M. Svensén and C. Williams, "The generative topographic mapping", Neurocomputing, vol. 10, Page(s) 33-42 ,May 1998.

[77] P. Poincot , S. Lesteven and F. Murtagh,"A spatial user interface to the astronomical literature", Astronomy and Astrophysics Supplement, vol. 130, Page(s): 183-191, May 1998

[78] T. Kohonen, "The self-organizing maps",Neurocomputing, vol. 21,Page(s) 1–6, September 1998.

[79] K. Andrews and H. Heidegger. "Information Slices: Visualising and Exploring Large Hierarchies using Cascading, Semi-Circular Discs". the 4th IEEE

Symposium on Information Visualization , page(s): 9–12,October 1998

[80] X. Lin, "Map displays for information retrieval", Journal of the American Society for Information Science, vol. 48,Page(s) 40-54 ,January 1997

[81] J. Ma, K. Iwama, T. Takaoka and Q. Gus, "Efficient Parallel and Distributed Topological Sort Algorithms", the 2nd AIZU International Symposium on Parallel Algorithms / Architecture Synthesis,Page 378 ,March 1997

[82] D. Merkl and A. Rauber, "Alternative ways for cluster visualization in self-organizing maps", 1st Workshop Self-Organizing Maps, Page(s) 106–111,May 1997.

[83] S. Lesteven, P. Poincot and F. Murtagh, "Neural networks and information extraction in astronomical information retrieval", Strategies and Techniques of Information for Astronomy,vol. 40, Issue 3, Page(s): 395–400 ,January 1996

[84] H. Chen, C. Schuffels and R. Orwig, "Internet categorization and search: a self-organizing approach", Journal of Visual Communication and Image Representation, Vol. 7, Page(s): 88-102,March 1996

[85] M. Cottrell and E. de Bodt, "A Kohonen map representation to avoid misleading interpretations", 4th Europe Symposium on Artificial Neural Networks, Bruges, Belgium, Page(s) 103–110, April 1996.

[86] H. Hyotyniemi, "Text document classification with self-organizing maps", Finnish Artificial Intelligence Conference, Page(s) 64-72,August 1996

[87] K. Lagus, T. Honkela, S. Kaski and T. Kohonen, "Self-organizing map of Document Collection, A New Approach to Interaktive Exploration", the 2nd International Conference on Knowledge Discovery and Data Mining, pages 238-243, August 1996

[88] J. Zavrel, "Neural navigation interfaces for information retrieval: are they more than an appealing idea?", Artificial Intelligence Review , vol. 10, Page(s) 477-504, October 1996

[89] M. Kraaijveld, J. Mao and A. Jain, "A nonlinear projection method based on Kohonen’s topology preserving maps", IEEE Transactions on Neural Networks, vol. 6, Page(s) 548–559, May 1995.

[90] B. Fritzke, "Growing grid A self-organizing network with constant neighborhood range and adaptation strength", Neural Process. Lett., vol.2, Page(s) 9–13, September 1995.

[91] D. Merkl, A.M. Tjoa and G. Kappel, "Application of self-organizing feature maps with lateral inhibition to structure a library of reusable software components", IEEE World Congress on Computational Intelligence Neural Networks, vol. 6, Page(s): 3905 - 3908 ,July 1994

[92] J. Greiner,"A comparison of parallel algorithms for connected components", the 6th annual ACM symposium on Parallel algorithms and architectures, Page(s): 16-25 ,August 1994

[93] L. Leinonen, T. Hiltunen, K. Torkkola and J. Kangas. "Self-organized acoustic feature map in detection of fricative-vowel co-articulation", Journal of the Acoustical Society of America ,vol 93, Page(s) 3468–3474,June 1993

[94] A. Ultsch, "Self-organizing neural networks for visualization and classification", Information and Classification-Concepts Methods and Applications, Page(s) 307–313,September 1993.

[95] X. Lin, "Visualization for the document space", the 3rd conference on Visualization , Page(s): 274 - 281 ,October 1992

[96] X. Lin, D. Soergel and G. Marchionini, "A self-organizing semantic map for information retrieval", the 14th annual international ACM SIGIR conference on Research and development in information retrieval, Page(s): 262-269 ,September 1991

[97] X. Lin, D. Soergel and G. Marchionini, "A self-organizing Semantic Map for Information Retrieval", the 14th annual international ACM SIGIR conference on Research and development in information retrieval, Illinois, United States, pages 262-269,September 1991

[98] S. Kirkpatrick, C. Gelatt and M. Vecchi, "Optimization by Simulated Annealing. Science", Doctoral Dissertation University of Illinois at Urbana-Champaign Champaign, Page(s):671–680.January 1987

[99] P. Laarhoven and E. Aarts, "Simulated annealing: theory and applications", Kluwer Academic Publishers Norwell ISBN:9-027-72513-6,January 1987

[100] M. Garey and D. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness", W. H. Freeman ISBN:0716710447, January 1979.

[101] W. Wang, J. Yang, and P. Yu, "Efficient Mining of Weighted Association Rules(WAR)".the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 2000.