Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2016, Article ID 5139574, 11 pages
http://dx.doi.org/10.1155/2016/5139574

Research Article

On the Use of Self-Organizing Map for Text Clustering in Engineering Change Process Analysis: A Case Study

Massimo Pacella, Antonio Grieco, and Marzia Blaco

Dipartimento di Ingegneria dell'Innovazione, Università del Salento, 73100 Lecce, Italy

Correspondence should be addressed to Marzia Blaco; marzia.blaco@unisalento.it

Received 17 March 2016; Accepted 30 October 2016

Academic Editor: Jens Christian Claussen

In modern industry, the development of complex products involves engineering changes that frequently require redesigning or altering the products or their components. In an engineering change process, engineering change requests (ECRs) are documents (forms) with parts written in natural language describing a suggested enhancement or a problem with a product or a component. ECRs initiate the change process and promote discussions within an organization to help to determine the impact of a change and the best possible solution. Although ECRs can contain important details, that is, recurring problems or examples of good practice repeated across a number of projects, they are often stored but not consulted, missing important opportunities to learn from previous projects. This paper explores the use of Self-Organizing Map (SOM) to the problem of unsupervised clustering of ECR texts. A case study is presented in which ECRs collected during the engineering change process of a railways industry are analyzed. The results show that SOM text clustering has a good potential to improve overall knowledge reuse and exploitation.

1. Introduction

The development of complex products, such as trains or automobiles, involves engineering changes that frequently require redesigning or altering the products and their components. As defined by Jarratt et al. [1] "engineering change is an alteration made to parts, drawings or software that have already been released during the design process. The change can be of any size or type, can involve any number of people and can take any length of time." A change may encompass any modification to the form, fit, and/or function of the product as a whole or in part, materials, and may alter the interactions and dependencies of the constituent elements of the product. A change can be needed to solve quality problems or to meet new customer requirements. Although engineering change management was historically seen as a typical design and manufacturing research field, several contributions highlighted the effect of engineering change on other business processes such as material requirement planning [2] and enterprise resource planning [3, 4]. An overview of the engineering change process and a big picture of literature on engineering change management are provided, respectively, by Jarratt et al. [5] and Hamraz et al. [6].

The engineering change request (ECR) is the document which initiates the engineering change process. ECR is used to describe a required change or a problem which may exist in a given product. After the ECR, the impact of a change is discussed among involved stakeholders and the best possible solution is identified.

Once the implementation of a change is completed, too often ECRs are no longer consulted by who could benefit from them. However, reviewing the ECR documents could offer a chance to improve both the design of a product and the engineering change process. A change may be a chance to both improve the product and do things "better next time" [9]. ECRs are documents containing structured and unstructured data, which, if analyzed, may be useful to discover information relating to recurring problems and solutions adopted in the past.

As described in Hamraz et al. [6], a lot of literature concerns the prechange stage of the process and proposes methods to prevent or to ease the implementation of engineering changes before they occur. In contrast, the

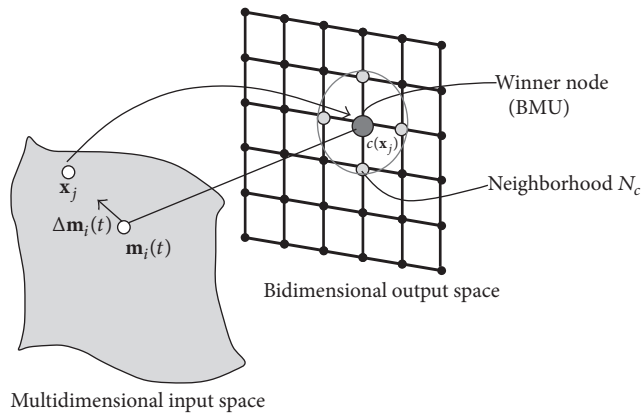

Figure 1: SOM training algorithm. Each neuron has a prototype vector $\mathbf{m}_i$, which corresponds to a point in the embedding input space. An input vector $\mathbf{x}_j$ will select that neuron with closest $\mathbf{m}_i$ to it. Adjustment of the weight vector for the winning output neuron and its neighbors through quantity $\Delta\mathbf{m}_i$. Adapted from Ritter et al. [7].

postchange stage involves less publication and deals with the ex post facto exploration of effect of implemented engineering changes. The analysis of engineering changes process belongs to the postchange stage and there are only few approaches concerning the analysis of engineering changes data in complex products industry. In this context, one of the main challenges is dealing with free-form text contained in engineering changes documents which makes the data more difficult to query, search, and extract. This paper focuses on unstructured data contained in ECRs and proposes the text clustering for the postchange analysis of engineering change process.

Text clustering is an unsupervised learning method where similar documents are grouped into clusters. The goal is to create clusters that are coherent internally, but clearly different from each other. Among the clustering methods proposed in the literature, Self-Organizing Map (SOM) has attracted many researchers in recent years. SOM is a neural-network model and algorithm that implements a characteristic nonlinear projection from the high-dimensional space of input signals onto a low-dimensional regular grid, which can be effectively utilized to visualize and explore properties of the data [10]. With respect to other text clustering methods, SOM allows visualizing the similarity between documents within the low-dimensional grid. Hence, similar documents may be found in neighboring regions of the grid.

In the literature, text mining methods have been proposed in support of the engineering change process by Sharafi et al. [11], Elezi et al. [12], and Sharafi [13]. In particular, Sharafi et al. [11] focused on the causes of changes contained in ECRs and calculated term occurrences for all ECRs in order to analyze occurrences of the keywords in different projects and to find pattern in the data. Elezi et al. [12] employed a semiautomatic text mining process to classify the causes of iteration in engineering changes. As a result, cost and technical categories of causes were identified as the main reasons for the occurrence of iterations. Sharafi [13] applied Knowledge Discovery in Database methods to analyze historical engineering changes data in order to gain insights in form of patterns within the database. In detail, a part of the study concerned the application of K-means, K-Medoids, DBSCAN, and Support Vector Clustering methods to cluster ECRs documents of an automotive manufacturer.

This paper explores the use of SOM to the problem of unsupervised clustering of ECR documents. A case study is presented and ECRs collected during the engineering change process of a railway industry are analyzed. The results show that SOM text clustering has great potential to improve overall knowledge reuse and exploitation in an engineering change process.

The reminder of the paper is organized as follows. In Section 2, the basic concepts of the SOM theory are introduced. In Section 3, the SOM text based clustering method is described. In Section 4, the engineering change process in industry is described. In Section 5, the case study and the experimental results are both discussed. In Section 6, conclusions are given.

2. The SOM Algorithm

The SOM, originally proposed by Kohonen [14], is based on the idea that systems can be designed to emulate the collective cooperation of the neurons in the human brain. It is an unsupervised machine learning method widely used in data mining, visualization of complex data, image processing, speech recognition, process control, diagnostics in industry and medicine, and natural language processing [15].

The algorithm of SOM consists in mapping M-dimensional input vectors $\mathbf{x}_j$ to two-dimensional *neurons* or *maps* according to their characteristic features. It reduces the dimensions of data to a map, helps to understand high-dimensional data, and groups similar data together. A simple SOM consists of two layers. The first includes nodes in the input space and the second the nodes in the output space. A representation of SOM with output nodes in a two-dimensional grid view is provided in Figure 1. SOM consists of P units; each unit of index i is associated with an M-dimensional prototype vector $\mathbf{m}_i$ in the input space and a

position vector on a low-dimensional regular grid, $\mathbf{r}_i$, in the output space. The steps of the SOM learning algorithm are as follows:

(1) *Initialization.* Start with initial values of prototype vectors $\mathbf{m}_i$. In the absence of any prior information, values of prototype vector $\mathbf{m}_i$ can be random or linear and are adjusted while the network learns.

(2) *Sampling.* Select a vector $\mathbf{x}_j$ from the training input space. The selection of $\mathbf{x}_j$ can be random.

(3) *Matching.* Determine the Best Matching Unit (BMU). Vector $\mathbf{x}_j$ is compared with all the prototype vectors and the index $c(\mathbf{x}_j)$ of the BMU; that is, the prototype vector $\mathbf{m}_c$ which is closest to $\mathbf{x}_j$ is chosen accordingly to the smallest Euclidian distance as follows:

$$\left\|\mathbf{x}_j - \mathbf{m}_c\right\| = \min_i \left\|\mathbf{x}_j - \mathbf{m}_i\right\|. \tag{1}$$

(4) *Updating.* Update the BMU and its neighbors. Adjustment of the prototype vector for the winning output neuron and its neighbors are updated as

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \Delta\mathbf{m}_i(t), \tag{2}$$

where $t = 0, 1, 2, \ldots$ is an index of the time. The value of $\Delta\mathbf{m}_i(t)$ in (2) is computed as follows:

$$\Delta\mathbf{m}_i(t) = \alpha(t)\, h_{ci}(t) \left(\mathbf{x}_j(t) - \mathbf{m}_i(t)\right), \tag{3}$$

where $\alpha(t)$ is the learning-rate factor and $h_{ci}(t)$ the neighborhood function. In particular, the learning-rate factor $\alpha(t)$ is comprised in $[0, 1]$ and is monotonically decreasing during the learning phase. The neighborhood function $h_{ci}(t)$ determines the distance between nodes of indexes c and i in the output layer grid. A widely applied neighborhood kernel can be written in terms of the Gaussian function:

$$h_{ci}(t) = \exp\left(-\frac{\left\|\mathbf{r}_c - \mathbf{r}_i\right\|^2}{2\sigma^2(t)}\right), \tag{4}$$

where $\mathbf{r}_c$ and $\mathbf{r}_i$ are the position vectors of nodes c and i and the parameter $\sigma(t)$ defines the width of the kernel which corresponds to the radius of the neighborhood $N_c(t)$. $N_c(t)$ refers to a neighborhood set of array points around node of index c (Figure 1). The value $h_{ci}(t)$ decreases during learning, from an initial value often comparable to the dimension of the output layer grid to a value equal to one.

During the learning of the SOM, phases 2–4 are repeated for a number of successive iterations until the prototype vectors $\mathbf{m}_i$ represent, as much as possible, the input patterns $\mathbf{x}_j$ that are closer to the neurons in the two-dimensional map. After initialization, the SOM can be trained in a sequential or batch manner [8]. Sequential training is repetitive as batch training but instead of sending all data vectors to the map for weight adjustment, one data vector at a time is sent to the network. Once the SOM is trained, each input vector is mapped to one neuron of the map, reducing high-dimensional input space to a low-dimensional output space. The map size depends on the type of application. The bigger size map reveals more details of information whereas a smaller map is being chosen to guarantee the generalization capability.

Before application, SOM method requires predefining the size and structure of the network, the neighborhood function, and the learning function. These parameters are generally selected on the basis of heuristic information [7, 8, 16, 17].

2.1. SOM Cluster Visualization. The SOM is an extremely versatile tool for visualizing high-dimensional data in low dimension. For visualization of SOM both the unified-distance matrix (U-matrix) [18] and Component Planes [19] are used. The U-matrix calculates distances between neighboring map units, and these distances can be visualized to represent clusters using a color scale on the map.

The U-matrix technique is a single plot that shows cluster borders according to dissimilarities between neighboring units. The distance ranges of U-matrix visualized on the map are represented by different colors (or grey shades). Red colors correspond to large distances; that is, large gaps exist between the prototype vector values in the input space; blue colors correspond to small distance; that is, map units are strongly clustered together. U-matrices are useful tools for visualizing clusters in input data without having any a priori information about the clusters.

Another important tool of visualization is Component Planes, that is, a grid whose cells contain the value of the pth dimension of a prototype vector displayed by variation of color. It helps to analyze the contribution of each variable to cluster structures and the correlation between the different variables in the dataset.

2.2. SOM Clustering Using K-Means Algorithm. One of the drawbacks of SOM analysis is that unlike other cluster methods, the SOM has no distinct cluster boundaries. When datasets become more complex it is not easy to distinguish the cluster by pure visualization. As described in Vesanto and Alhoniemi [10], in SOM the prototype nodes can be used for clustering instead of all input dataset. Let $\mathscr{C}_1, \ldots, \mathscr{C}_k, \ldots, \mathscr{C}_K$ denote a cluster partition composed of K clusters. The choice of the best clustering can be determined by applying the well-known K-means algorithm [20]. This algorithm minimizes an error function computed on the sum of squared distances of each data point in each cluster. The algorithm iteratively computes partitioning for the data and updates cluster centers based on the error function. In this approach, the number of clusters K has to be fixed a priori. Therefore K-means algorithm is run multiple times for each $K \in [2, \sqrt{N}]$, where N is number of samples. The best number of clusters K^* can be selected based on the Davies Bouldin Index (DBI) [21]. This index is based on a ratio of within-cluster and between-cluster distances and is calculated as

$$\mathrm{DBI}(K) = \frac{1}{K}\sum_{k=1}^{K} \max_{k \neq l} \left\{\frac{\Delta(\mathscr{C}_k) + \Delta(\mathscr{C}_l)}{\delta(\mathscr{C}_k, \mathscr{C}_l)}\right\}, \tag{5}$$

where K is the number of clusters, $\Delta(\mathscr{C}_k)$ and $\Delta(\mathscr{C}_l)$, and $\delta(\mathscr{C}_k, \mathscr{C}_l)$ the within-cluster and between-cluster distances, respectively. The optimum number of clusters K^* corresponds to the minimum value of DBI(K). SOM neural network combined with other clustering algorithms was used in Yorek et al. [22] for visualization of students' cognitive structural models.

3. SOM-Based Text Clustering

Text clustering is an unsupervised process used to separate a document collection into some clusters on the basis of the similarity relationship between documents in the collection [17]. Suppose $\mathscr{C} = \{d_1, \ldots, d_j, \ldots, d_N\}$ be a collection of N documents to be clustered. The purpose of text clustering is to divide $\mathscr{C}$ into $\mathscr{C}_1, \ldots, \mathscr{C}_k, \ldots, \mathscr{C}_K$ clusters, with $\mathscr{C}_1 \cdots \cup \mathscr{C}_k \cdots \cup \mathscr{C}_K = \mathscr{C}$.

SOM text clustering can be divided into two main phases [23, 24]. The first phase is *document preprocessing* which consists in using Vector Space Model (VSM) to generate output document vectors from input text documents. The second one is *document clustering* that applies SOM on the generated document vectors to obtain output clusters.

3.1. Document Preprocessing. An important preprocessing aspect for text clustering is to consider how the text content can be represented in the form of mathematical expression for further analysis and processing.

By means of VSM, each d_j ($j = 1, \ldots, N$) can be represented as vector in M-dimensional space. In detail, each document d_j can be represented by a numerical feature vector $\mathbf{x}_j$:

$$\mathbf{x}_j = \left[w_{1,j}, \ldots, w_{p,j}, \ldots, w_{M,j}\right]. \quad (6)$$

Each element $w_{p,j}$ of the vector usually represents a word (or a group of words) of the document collection; that is, the size of the vector is defined by the number of words (or groups of words) of the complete document collection.

The simplest approach is to assign to each document the *Term Frequency and Inverse Document Frequency* (TF-IDF) weighting scheme [25, 26]. The TF-IDF weighting scheme assigns to each term p in the jth document a weight $w_{p,j}$ computed as

$$w_{p,j} = tf_{p,j} \times \log \frac{N}{df_p}, \quad (7)$$

where $tf_{p,j}$ is the term frequency; that is, the number of times that term p appears in the document j and df_p is the number of documents in the collection which contains term p.

According to TF-IDF weighting scheme, $w_{p,j}$ is

(1) higher when the term p occurs many times within a small number of documents (thus lending high discriminating power to those documents),

(2) lower when the term p occurs fewer times in a document or occurs in many documents (thus offering a less pronounced relevance signal),

(3) lower when the term p occurs in virtually all documents.

Before preprocessing the documents by the TF-IDF weighting scheme, The size of the list of terms created from documents can be reduced using methods of *stop words removal* and *stemming* [23, 24].

In text based document, in fact, there are a great number of noninformative words, such as articles, prepositions, and conjunctions, called stop words. A stop-list is usually built with words that should be filtered in the document representation process. Words that are to be included in the stop-list are language and task dependent; however a set of general words can be considered stop words for almost all tasks, such as "and" and "or." Words that appear in very few documents are also filtered.

Another common phase in preprocessing is stemming, where the word stem is derived from the occurrence of a word by removing case and inflection information. For example, "computes," "computing," and "computer" are all mapped to the same stem "comput." Stemming does not alter significantly the information included in document representation, but it does avoid feature expansion.

3.2. SOM Text Clustering. Once obtaining the feature vector $\mathbf{x}_j$ in (6) associated with each text d_j, the SOM algorithm described in Section 2 can be applied for text clustering. The text clustering method explained above is known as "SOM plus VSM"; other variants to it have been proposed by Liu et al. [27, 28]. An overview of the application of SOM in text clustering is provided by Liu et al. [17]. This kind of clustering method was employed in domains such as patent [29], financial services [30], and public policy analysis [31].

4. The Engineering Change Process in Complex Products Industry

For complex products, such as trains, automobiles, or aircraft, engineering changes are unavoidable and products or components have to be redesigned and retrofitted to accommodate the new changes to new installations and products. In these environments, an engineering change can involve the risk of due time delay. Huang et al. [32] carried out a survey about the effects of engineering changes on four manufacturing industries and found that the time invested in processing an engineering change varies from 2 to 36 person days. In Angers [33] it is estimated that more than 35% of today's manufacturing resources are just devoted to managing changes to engineering drawings, manufacturing plans, and scheduling requirements. Engineering change processes in complex environments such as automobile, train, and aviation industry were also studied by Leng et al. [3] and Subrahmanian et al. [34].

The phases of a real engineering change process in complex products industry can be summarized as follows (Figure 2):

(1) A request of an engineering change is made and sent to an engineering manager. In this phase, standard ECR forms are used outlining the reason of the change, the type of the change, which components or

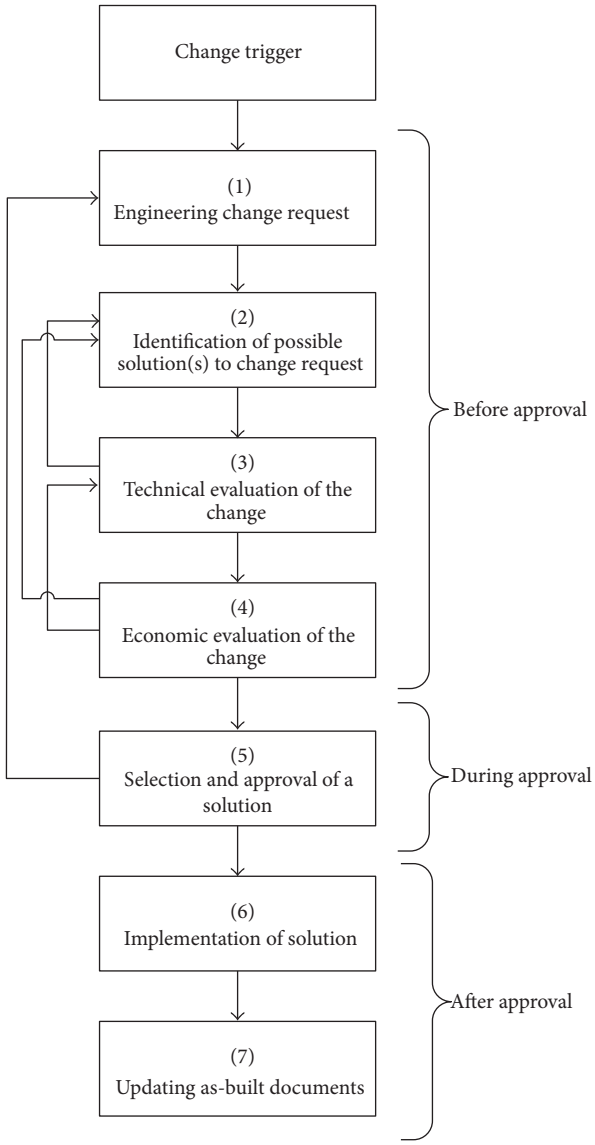

Figure 2: The real engineering change process in complex products environments. Adapted from Jarratt et al. [5].

systems are likely to be affected, the person and the department making the request, and so forth.

(2) Potential solutions to the request for change are identified.

(3) Technical evaluation of the change is carried out. In this phase, the technical impact of implementing each solution is assessed. Various factors are considered, for example, the impact upon design and product requirements, production schedule, and resources to be devoted.

(4) Economic evaluation of the change is performed. The economic risk of implementing each solution is assessed. In this phase, costs related to extra production times, replacements of materials, penalty for missed due date, and so forth are estimated.

(5) Once a particular solution is selected, it is approved or not approved. The change is reviewed and a cost benefit analysis is carried out. When a solution is approved, the engineering change order is prepared and issued.

(6) Implementation of the engineering change and identification of the documents, such as drawings, are to be updated.

(7) Update of the as-built documents occurs. As-built documents are usually the original design documents revised to reflect any changes made during the process, that is, design changes, material changes, and so forth.

Iterations of the process occur, for example, when a particular solution has negative impact on product requirements or is too risky to be implemented so the process returns to phase 2 and another solution is identified. Another iteration is possible when the costs of a solution are too high or more risk analysis is required or when the proposed solution is completely refused.

As shown in Figure 2, no review process of similar changes faced in the past is carried out during the process or at the end. This aspect is emphasized by Jarratt et al. [5] by highlighting that, after a period of time, the change should be reviewed to verify if it achieved what was initially intended and what lessons can be learned for future change process. Various factors can discourage examining the solutions adopted in the past to a particular change. First of all, there is the lack of opportune methods to analyze the documents collected during the process, that is, ECR. ECRs are often forms containing parts written in natural language. Analyzing these kinds of documents in the design phase of a product or a component or when a new change request occurs could be very time consuming without an appropriate solution.

In this context, SOM text clustering application can improve the process. When a new ECR occurs, in fact, ECRs managed in the past and similar to the current request could be analyzed in order to evaluate the best solution and to avoid repeating the same mistakes made in the past. In order to explore the existence of similarity between the different ECRs texts, the first step is to verify the potential clustering present in the analyzed dataset. The application of SOM text clustering to ECR documents is explored in the next section.

5. The Use of SOM Text Clustering for ECRs Analysis

In order to test SOM text clustering, we used a dataset of $N = 54$ ECRs representing some engineering changes managed during the engineering change process of a railway company. The dataset included the natural language written descriptions of the causes of changes contained in the ECRs forms. The documents were written in Italian language and the VSM in Section 3 was used to generate output vectors from input text documents. The number of terms, that is, the dimension of each vector $\mathbf{x}_j$ associated with each document

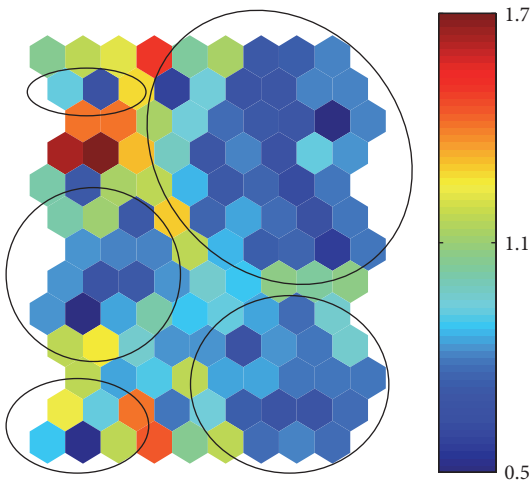

Figure 3: U-matrix of the 54 ERCs texts preprocessed through the VSM. Color scale is related to distances between map units. Red colors represent large distances and blue colors represent small distances.

in the dataset after the *stop word removal* and *stemming* processes, was equal to $M = 361$.

In our work, we used the MATLAB software. Specifically, Term to Matrix Generator (TMG) toolbox [35] for document preprocessing and SOM toolbox for SOM text clustering [8] were employed.

The map size of the SOM is computed through the heuristic formula in Vesanto et al. [8]. In detail, the number of neurons is computed as $5\sqrt{N}$, where N is the number of training samples. Map shape is a rectangular grid with hexagonal lattice. The neighborhood function is Gaussian and the map is trained using the batch version of SOM. After the map is trained, each data vector is mapped to the most similar prototype vector in the map, that is, the BMU which results from the matching step in the SOM algorithm. In our case, the network structure is a 2D-lattice of 7×5 hexagonal.

A first step in cluster analysis based on SOM is based on visual inspection through U-matrix and Component Planes.

The U-matrix obtained on the application of SOM to ECR dataset is shown in Figure 3. In order to represent additional information (i.e., distances), the SOM map size is augmented by inserting an additional neuron between each pair of neurons and reporting the distance between the two neurons. The U-matrix and the different colors linked to the distance between neurons on the map show that five clusters are present in the dataset. Each cluster has been highlighted by a circle.

Besides looking at the overall differences in the U-matrix, it is interesting as well to look at the differences between each component present in the input vectors, meaning that we look at differences regarding each component associated with a single "term" in the input dataset. The total number of Component Planes obtained from the SOM corresponds to the total number of terms in our dataset; that is, $M = 361$. For illustrative purpose, Figure 4 shows two Component Planes chosen as an example. The first Component Plane

Table 1: Class, label, and number of ECR texts.

Class	Label	Number of ECR texts
(1) Metal-Sheet	MS	12
(2) Carter	CR	8
(3) Antenna	AN	8
(4) Semi-Finished Round	SR	7
(5) Hydraulic Panel	HP	9
(6) Pneumatic System	PS	10

(Figure 4(a)) is associated with the term of index $p = 48$, that is, the term "Antenna"; the second one (Figure 4(b)) is related to the term of index $p = 195$, that is, the term "Metal-Sheet." The difference between the two terms in the dataset can be represented by considering, for example, the map unit in top left corner of the two figures. This map unit has high values for variable "Term $p = 48$" and low values for variable "Term $p = 195$." From the observation of Component Planes, we can conclude that there is no correlation between the two terms. As a matter of fact, these two terms were never used together into the same documents.

As shown above, the U-matrix and the Component Planes allow obtaining a rough cluster structure of the ECR dataset. To get a finer clustering result, the prototype vectors from the map were clustered using K-means algorithm. The best number of clusters in the SOM map grip can be determined by using the DBI values. Figure 5(a) shows the DBI values by varying the number of clusters K in $[2, 7]$. The elbow point in the graph shows that the optimum number of cluster K^* corresponding to the minimum value of DBI(K) is equal to 5. The clustering result of SOM obtained by using K-means with $K^* = 5$ clusters is shown in Figure 5(b). The BMUs belonging to each cluster are represented with a different color and in each hexagon the number of documents associated with each BMU is provided.

5.1. External Validation of SOM Text Clustering. In the reference case study, the true classification of each ECR in the dataset was provided by process operators. Therefore, this information was used in our study in order to perform an external validation of the SOM text clustering. In particular, each ECR text was analyzed and classified with reference to the main component involved in the engineering change (namely, "Metal-Sheet," "Carter," "Antenna," "Semi-Finished Round," "Hydraulic Panel," and "Pneumatic System"). Table 1 reports the number of ECRs related to each component, along with the labels used in order to classify the ECR documents (namely, "MS," "CR," "AN," "SR," "HP," and "PS," respectively). Although a classification is available in the specific case study of this paper, it is worth noting that often such information may be unavailable.

By superimposing the classification information, the map grid resulting by the training of the SOM is as in Figure 6, where each hexagon reports the classification label of documents sharing a given BMU (within brackets, the number of associated documents). From Figure 6, it can be observed that the unsupervised clustering produced by the SOM algorithm is quite coherent with the actual classification

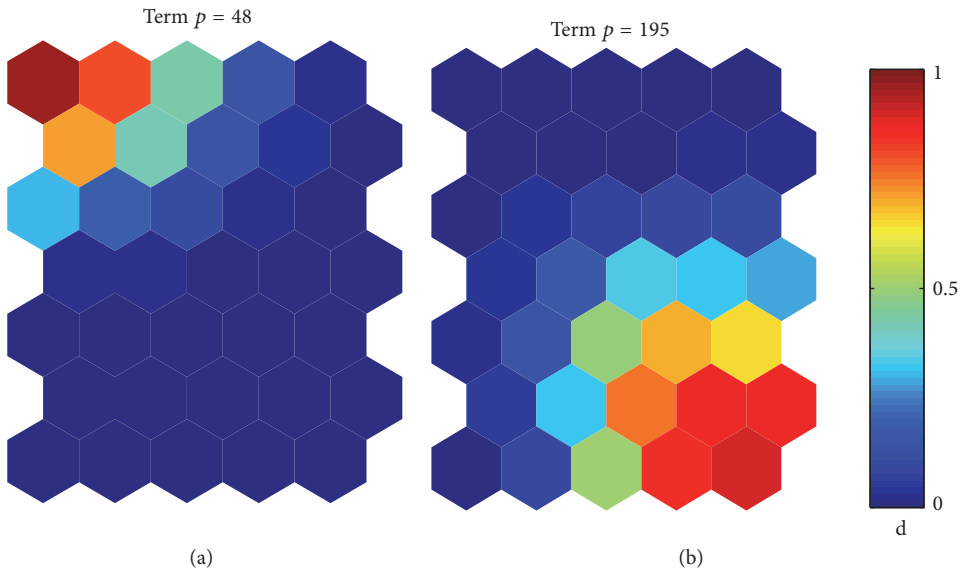

FIGURE 4: Component Planes of two variables chosen as example. (a) Component Plane related to the variable of index $p = 48$ corresponding to the term "Antenna." (b) Component Plane of the variable of index $p = 195$ corresponding to the term "Metal-Sheet." (a) and (b) are linked by position: in each figure, the hexagon in a certain position corresponds to the same map units. Each Component Plane shows the values of the variable in each map unit using color-coding. Letter "d" means that denormalized values are used; that is, the values in the original data context are used [8].

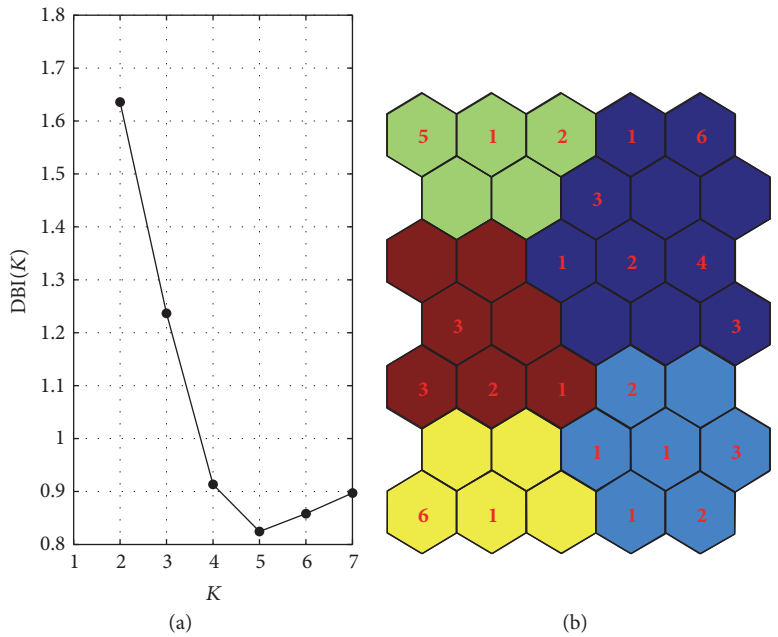

FIGURE 5: (a) Davies Bouldin Index (DBI) values for number of clusters K in $[2, 7]$. (b) Clustering result of SOM using K-means with $K^* = 5$ clusters. In each hexagon the number of documents sharing the same BMU is provided.

given by process operators; in fact ECR documents sharing the same classification label are included in BMUs belonging to the same cluster. It is worth noting that the actual label is assigned to each document after the SOM unsupervised training has been carried out. From Figure 6, it can be also noted that ECRs, classified either as "PS" and "HP," are all included in a unique cluster. Furthermore, two documents out of twelve with label "MS" are assigned to clusters that include different labels, namely, "CR," "PS," and "HP." We investigated this misclassification and we found that causes

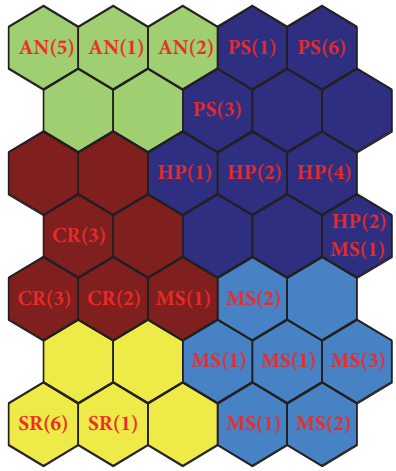

Figure 6: Labeled map grid of SOM. In each hexagon, labels and total number of documents sharing the same BMU. Coloring refers to K-means unsupervised clustering of the SOM.

Table 2: Contingency matrix $\mathbf{N}$.

	MS $\mathcal{T}_1$	CR $\mathcal{T}_2$	AN $\mathcal{T}_3$	SR $\mathcal{T}_4$	HP $\mathcal{T}_5$	PS $\mathcal{T}_6$	n_r
C_1	0	0	8	0	0	0	8
C_2	1	0	0	0	9	10	20
C_3	1	8	0	0	0	0	9
C_4	0	0	0	7	0	0	7
C_5	10	0	0	0	0	0	10
m_k	12	8	8	7	9	10	$N = 54$

of changes described in these two "MS" documents are quite similar to those contained in documents labeled as "CR," "PS," and "HP."

Given the actual classification, the quality of the obtained clustering can be evaluated by computing four indices: purity, precision, recall, and F-measure [36].

Let $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_r, \ldots, \mathcal{T}_R\}$ be the true partitioning given by the process operators, where the partition $\mathcal{T}_r$ consists of all the documents with label r. Let $m_r = |\mathcal{T}_r|$ denote the number of documents in true partition $\mathcal{T}_r$. Also let $\mathcal{C} = \mathcal{C}_1, \ldots, \mathcal{C}_k, \ldots, \mathcal{C}_K$ denote the clustering obtained via the SOM text clustering algorithm, and $n_k = |\mathcal{C}_k|$ denote the number of documents in cluster $\mathcal{C}_k$. The $K \times R$ contingency matrix $\mathbf{N}$ induced by clustering $\mathcal{C}$ and the true partitioning $\mathcal{T}$ can be obtained by computing the elements $\mathbf{N}(k, r) = n_{kr} = |\mathcal{C}_k \cap \mathcal{T}_r|$, where n_{kr} denotes the number of documents that are common to cluster $\mathcal{C}_k$ and true partition $\mathcal{T}_r$. The contingency matrix for SOM text clustering of ECRs is reported in Table 2.

Starting from Table 2, the following indices are computed:

(i) *Purity Index*. The cluster-specific purity index of cluster $\mathcal{C}_k$ is defined as

$$\text{purity}_k = \frac{1}{n_k} \max_{r=1}^{R} \{n_{kr}\}. \tag{8}$$

The overall purity index of the entire clustering $\mathcal{C}$ is computed as

$$\text{purity} = \sum_{k=1}^{K} \frac{n_k}{N} \text{purity}_k = \frac{1}{N} \sum_{k=1}^{K} \max_{r=1}^{R} \{n_{kr}\}. \tag{9}$$

As shown in Table 3, clusters $\mathcal{C}_1$, $\mathcal{C}_4$, and $\mathcal{C}_5$ have purity index equal to 1; that is, they contain entities from only one partition. Clusters $\mathcal{C}_2$ and $\mathcal{C}_3$ gather entities from different partitions, that is, $\mathcal{T}_1$, $\mathcal{T}_5$, and $\mathcal{T}_6$ for cluster $\mathcal{C}_2$ and $\mathcal{T}_1$, $\mathcal{T}_2$ for cluster $\mathcal{C}_3$. The overall purity index is equal to 0.79.

(ii) *Precision Index*. Given a cluster $\mathcal{C}_k$, let $\mathcal{T}_{r_k}$ denote the majority partition that contains the maximum number of documents from $\mathcal{C}_k$; that is, $r_k = \text{argmax}_{r=1}^{R} n_{kr}$. The *precision* index of a cluster $\mathcal{C}_k$ is given by

$$\text{prec}_k = \frac{1}{n_k} \max_{r=1}^{R} \{n_{kr}\} = \frac{n_{kr_k}}{n_k}. \tag{10}$$

For clustering in Table 2 the majority partitions are $\mathcal{T}_{3_1}$, $\mathcal{T}_{6_2}$, $\mathcal{T}_{2_3}$, $\mathcal{T}_{4_4}$, and $\mathcal{T}_{1_5}$. Precision indices show that all documents gathered in clusters $\mathcal{C}_1$, $\mathcal{C}_4$, and $\mathcal{C}_5$ belong to the corresponding majority partitions $\mathcal{T}_{3_1}$, $\mathcal{T}_{4_4}$, and $\mathcal{T}_{1_5}$. For cluster $\mathcal{C}_2$ the 50% of documents belong to $\mathcal{T}_{6_2}$ and finally the 88% of documents in cluster $\mathcal{C}_3$ belong to $\mathcal{T}_{2_3}$.

(iii) *Recall Index*. Given a cluster $\mathcal{C}_k$, it is defined as

$$\text{recall}_k = \frac{n_{kr_k}}{|\mathcal{T}_{r_k}|} = \frac{n_{kr_k}}{m_{r_k}}, \tag{11}$$

where $m_{r_k} = |\mathcal{T}_{r_k}|$. It measures the fraction of documents in partition $\mathcal{T}_{r_k}$ shared in common with cluster $\mathcal{C}_k$. The recall indices reported in Table 3 show that clusters $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ shared in common the 100% of documents in majority partitions $\mathcal{T}_{3_1}$, $\mathcal{T}_{6_2}$, $\mathcal{T}_{2_3}$, and $\mathcal{T}_{4_4}$, respectively. Cluster $\mathcal{C}_5$ shared the 83% of documents in $\mathcal{T}_{1_5}$.

(iv) *F-Measure Index*. It is the harmonic mean of the precision and recall values for each cluster. The F-measure for cluster C_k is therefore given as

$$F_k = \frac{2 \cdot \text{prec}_k \cdot \text{recall}_k}{\text{prec}_k + \text{recall}_k} = \frac{2n_{kr_k}}{n_k + m_{r_k}} \tag{12}$$

The overall F-measure for the clustering $\mathcal{C}$ is the mean of the clusterwise F-measure values:

$$F = \frac{1}{K} \sum_{k=1}^{K} F_k. \tag{13}$$

Table 3 shows that F-measure of clusters $\mathcal{C}_1$ and $\mathcal{C}_4$ is equal to 1, while other values are less than 1. The low values of F-measures for clusters $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_5$ depend on a low precision index for clusters $\mathcal{C}_2$ and $\mathcal{C}_3$ and on a low recall index for cluster $\mathcal{C}_5$. Consequently, the overall F-measure is equal to 0.90.

Table 3: Clustering validation.

Cluster-specific and overall indices				
Purity				
$purity_1 = 1$	$purity_2 = 0.5$	$purity_3 = 0.89$	$purity_4 = 1$	$purity_5 = 1$
		$purity = 0.79$		
Precision				
$prec_1 = 1$	$prec_2 = 0.5$	$prec_3 = 0.89$	$prec_4 = 1$	$prec_5 = 1$
Recall				
$recall_1 = 1$	$recall_2 = 1$	$recall_3 = 1$	$recall_4 = 1$	$recall_5 = 0.83$
F-measure				
$F_1 = 1$	$F_2 = 0.66$	$F_3 = 0.94$	$F_4 = 1$	$F_5 = 0.91$
		$F = 0.90$		

Table 4: Results of SOM-based classification. In the first and in the second column, labels and number of ECR texts of the actual classification are reported. In the third column, the number of ECRs correctly classified through the label of first associated BMU. In the fourth column, the number of document associated with an empty first BMU but correctly classified by considering the label of documents sharing the second associated BMU.

Labels	Number of ECRs	Number of ECRs classified through the 1st BMU	Number of ECRs classified through the 2nd BMU
MS	12	10	2
CR	8	6	2
AN	8	7	1
SR	7	6	1
HP	9	6	3
PS	10	10	—

Given the actual classification, the SOM can be further validated through a leave-one-out cross validation technique in order to check its classification ability. In particular, $N - 1$ ECR documents are used for training and the remaining one for testing (iterating until each ECR text in the data has been used for testing).

At each iteration, once SOM has been trained on $N - 1$ ECRs, and when the testing sample is presented as input, a BMU is selected in the matching step of the SOM algorithm. The label of the training documents associated with that BMU is considered. In the case of an empty BMU, that is, which results are not associated with any training documents, the closest one associated with at least one training document is considered instead, while in case of a BMU associated with training documents with more than one label, the label with greater number of documents is considered.

Table 4 shows the results of the leave-one-out cross validation. For each row, that is, for a given ECR label, the second column reports the total number of documents in the dataset, while the last two columns report the number of testing ECRs correctly classified by the SOM. In particular, the third column reports the number of testing ECRs correctly classified as they were connected to a first BMU associated with training documents with the same label. The last column refers to the number of ECRs associated with an empty first BMU that, nevertheless, resulted closest to a second BMU related to documents belonging to the same class of the testing sample. Also cross validation study demonstrates that labels given by SOM are coherent with actual classification and confirms the ability of SOM as classification tool.

6. Conclusions

In this paper, a real case study regarding the engineering change process in complex products industry was conducted. The study concerned the postchange stage of the engineering change process, in which past engineering changes data are analyzed to discover information exploitable in new engineering changes. In particular, SOM was used for clustering of natural language written texts produced during the engineering change process. The analyzed texts included the descriptions of the causes of changes contained in the ECR forms. Firstly, SOM algorithm was used as clustering tool to find relationships between the ECR texts and to cluster them accordingly. Subsequently, SOM was tested as classification tool and the results were validated through a leave-one-out cross validation technique.

The results of the real case study showed that the use of the SOM text clustering can be an effective tool in improvement of the engineering change process analysis. In particular, some of the advantages highlighted in this study are as follows:

(1) Text mining methods allow analyzing unstructured data and deriving high-quality information. The main difficulty in ECR analysis consisted in analyzing natural language written texts.

(2) Clustering analysis of past ECRs stored in the company allows automatically gathering ECRs on the basis of similarity between documents. When a new change triggers, the company can quickly focus on

the cluster of interest. Clustering can support the company to know if a similar change was already managed in the past, to analyze the best solution adopted and to learn avoiding the same mistakes made in the past.

(3) Use of SOM for ECRs text clustering allows automatically organizing large documents collection. With respect to other clustering algorithms, the main advantage of SOM text clustering is that the similarity of the texts is preserved in the spatial organization of the neurons. The distance among prototypes in the SOM map can therefore be considered as an estimate of the similarity between documents belonging to clusters. In addition, SOM can first be computed using a representative subset of old input data. New input can be mapped straight into the most similar model without recomputing the whole mapping.

Nevertheless, the study showed some limitations of the application of SOM text clustering and classification. A first limitation is linked to natural language written texts. The terms contained in different texts may be similar even if an engineering change request concerns a different product. The similarity of terms may influence the performance of SOM-based clustering. A second limitation is linked to the use of SOM as classification method. Classification, indeed, requires the labeling of a training dataset. This activity requires a deep knowledge of the different kinds of ECRs managed during the engineering change process and may be difficult and time consuming.

As a summary, research on use of SOM text clustering in engineering change process analysis appears to be a promising direction for further research. A future direction of the work will consider the use of SOM text clustering on a larger dataset of ECRs comparing SOM with other clustering algorithms such as K-means or hierarchical clustering methods. Another direction of future research concerns the analysis of SOM robustness to parameters selection (i.e., the size and structure of the map, parameters and kinds of learning, and neighborhood functions).

Symbols

$\alpha(t)$: Scalar-valued learning-rate factor
$\mathbf{N}$: Contingency matrix
$\delta(\mathscr{C}_k, \mathscr{C}_l)$: Between-cluster distance
$\Delta(\mathscr{C}_k)$: Within-cluster distance
$\Delta\mathbf{m}_i(t)$: Adjustment of the prototype vector $\mathbf{m}_i(t)$
$\mathscr{C}$: Collection of documents
$\mathscr{C}_k$: Cluster of documents of index k
$\mathscr{T}$: True partition of documents given by process operators
$\mathscr{T}_r$: True partition of documents with label r
$\mathscr{T}_{r_k}$: Majority partition containing the maximum number of documents from $\mathscr{C}_k$
$\sigma(t)$: Width of the kernel which corresponds to the radius of $N_c(t)$
$c(\mathbf{x}_j)$: Index of the BMU associated with $\mathbf{x}_j$
d_j: Document of index j
DBI(K): Davies Bouldin Index for K clusters
df_p: Number of documents in the collection $\mathscr{C}$ which contains term p
$h_{ci}(t)$: Neighborhood function
K: Total number of clusters
K^*: Optimum number of clusters
M: Total number of terms in the documents collection
m_r: Number of documents in partition $\mathscr{T}_r$
N: Total number of processed documents
$N_c(t)$: Neighborhood around node c
n_k: Number of documents in cluster $\mathscr{C}_k$
n_{kr}: Number of documents common to cluster $\mathscr{C}_k$ and partition $\mathscr{T}_r$
P: Total number of units of SOM
t: Index of time
$tf_{p,j}$: Term frequency of term p in the jth document
$w_{p,j}$: Weight associated with term p in the jth document
$\mathbf{m}_c$: Prototype vector associated with the BMU
$\mathbf{m}_i$: Prototype vector of index i
$\mathbf{r}_c$: Position vector of index c
$\mathbf{r}_i$: Position vector of index i
$\mathbf{x}_j$: Numerical feature vector associated with jth document.

Competing Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research has been funded by MIUR, ITIA CNR, and Cluster Fabbrica Intelligente (CFI) Sustainable Manufacturing project (CTN01_00163_148175) and Vis4Factory project (Sistemi Informativi Visuali per i processi di fabbrica nel settore dei trasporti PON02_00634_3551288). The authors are thankful to the technical and managerial staff of MerMec S.p.A. (Italy) for providing the industrial case study of this research.

References

[1] T. Jarratt, J. Clarkson, and C. E. Eckert, *Design Process Improvement*, Springer, New York, NY, USA, 2005.

[2] C. Wänström and P. Jonsson, "The impact of engineering changes on materials planning," *Journal of Manufacturing Technology Management*, vol. 17, no. 5, pp. 561–584, 2006.

[3] S. Leng, L. Wang, G. Chen, and D. Tang, "Engineering change information propagation in aviation industrial manufacturing execution processes," *The International Journal of Advanced Manufacturing Technology*, vol. 83, no. 1, pp. 575–585, 2016.

[4] W.-H. Wu, L.-C. Fang, W.-Y. Wang, M.-C. Yu, and H.-Y. Kao, "An advanced CMII-based engineering change management framework: the integration of PLM and ERP perspectives," *International Journal of Production Research*, vol. 52, no. 20, pp. 6092–6109, 2014.

[5] T. A. W. Jarratt, C. M. Eckert, N. H. M. Caldwell, and P. J. Clarkson, "Engineering change: an overview and perspective on the literature," *Research in Engineering Design*, vol. 22, no. 2, pp. 103–124, 2011.

[6] B. Hamraz, N. H. M. Caldwell, and P. J. Clarkson, "A holistic categorization framework for literature on engineering change management," *Systems Engineering*, vol. 16, no. 4, pp. 473–505, 2013.

[7] H. Ritter, T. Martinetz, and K. Schulten, *Neural Computation and Self-Organizing Maps. An Introduction*, Addison-Wesley, New York, NY, USA, 1992.

[8] J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas, SOM toolbox for Matlab 5, 2005, http://www.cis.hut.fi/somtoolbox/.

[9] E. Fricke, B. Gebhard, H. Negele, and E. Igenbergs, "Coping with changes: causes, findings, and strategies," *Systems Engineering*, vol. 3, no. 4, pp. 169–179, 2000.

[10] J. Vesanto and E. Alhoniemi, "Clustering of the self-organizing map," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 586–600, 2000.

[11] A. Sharafi, P. Wolf, and H. Krcmar, "Knowledge discovery in databases on the example of engineering change management," in *Proceedings of the Industrial Conference on Data Mining-Poster and Industry Proceedings*, pp. 9–16, IBaI, July 2010.

[12] F. Elezi, A. Sharafi, A. Mirson, P. Wolf, H. Krcmar, and U. Lindemann, "A Knowledge Discovery in Databases (KDD) approach for extracting causes of iterations in Engineering Change Orders," in *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 1401–1410, 2011.

[13] A. Sharafi, *Knowledge Discovery in Databases: an Analysis of Change Management in Product Development*, Springer, 2013.

[14] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[15] T. Kohonen, "Essentials of the self-organizing map," *Neural Networks*, vol. 37, pp. 52–65, 2013.

[16] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[17] Y.-C. Liu, M. Liu, and X.-L. Wang, "Application of self-organizing maps in text clustering: a review," in *Applications of Self-Organizing Maps*, M. Johnsson, Ed., chapter 10, InTech, Rijeka, Croatia, 2012.

[18] A. Ultsch and H. P. Siemon, "Kohonen's self organizing feature maps for exploratory data analysis," in *Proceedings of the Proceedings of International Neural Networks Conference (INNC '90)*, pp. 305–308, Kluwer, Paris, France, 1990.

[19] J. Vesanto, "SOM-based data visualization methods," *Intelligent Data Analysis*, vol. 3, no. 2, pp. 111–126, 1999.

[20] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, 1967.

[21] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 224–227, 1979.

[22] N. Yorek, I. Ugulu, and H. Aydin, "Using self-organizing neural network map combined with Ward's clustering algorithm for visualization of students' cognitive structural models about aliveness concept," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 2476256, 14 pages, 2016.

[23] T. Honkela, S. Kaski, K. Lagus, and T. Kohonen, "Websom—self-organizing maps of document collections," in *Proceedings of the Work shop on Self-Organizing Maps (WSOM '97)*, pp. 310–315, Espoo, Finland, June 1997.

[24] S. Kaski, T. Honkela, K. Lagus, and T. Kohonen, "Websom—self-organizing maps of document collections," *Neurocomputing*, vol. 21, no. 1–3, pp. 101–117, 1998.

[25] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.

[26] G. Salton and R. K. Waldstein, "Term relevance weights in online information retrieval," *Information Processing & Management*, vol. 14, no. 1, pp. 29–35, 1978.

[27] Y. C. Liu, X. Wang, and C. Wu, "ConSOM: a conceptional self-organizing map model for text clustering," *Neurocomputing*, vol. 71, no. 4–6, pp. 857–862, 2008.

[28] Y.-C. Liu, C. Wu, and M. Liu, "Research of fast SOM clustering for text information," *Expert Systems with Applications*, vol. 38, no. 8, pp. 9325–9333, 2011.

[29] T. Kohonen, S. Kaski, K. Lagus et al., "Self organization of a massive document collection," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 574–585, 2000.

[30] J. Zhu and S. Liu, "SOM network based clustering analysis of real estate enterprises," *American Journal of Industrial and Business Management*, vol. 4, no. 3, pp. 167–173, 2014.

[31] B. C. Till, J. Longo, A. R. Dobell, and P. F. Driessen, "Self-organizing maps for latent semantic analysis of free-form text in support of public policy analysis," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 4, no. 1, pp. 71–86, 2014.

[32] G. Q. Huang, W. Y. Yee, and K. L. Mak, "Current practice of engineering change management in Hong Kong manufacturing industries," *Journal of Materials Processing Technology*, vol. 139, no. 1–3, pp. 481–487, 2003.

[33] S. Angers, "Changing the rules of the 'Change' game—employees and suppliers work together to transform the 737/757 production system," 2002, http://www.boeing.com/news/frontiers/archive/2002/may/i_ca2.html.

[34] E. Subrahmanian, C. Lee, and H. Granger, "Managing and supporting product life cycle through engineering change management for a complex product," *Research in Engineering Design*, vol. 26, no. 3, pp. 189–217, 2015.

[35] D. Zeimpekis and E. Gallopoulos, TMG: A Matlab toolbox for generating term-document matrices from text collections, 2006, http://scgroup20.ceid.upatras.gr:8000/tmg/.

[36] M. J. Zaki and W. Meira Jr., *Data Mining and Analysis: Fundamental Concepts and Algorithms*, Cambridge University Press, 2014.