Spatial-Temporal Clustering of a Self-Organizing Map

Carlos Enrique Gutierrez[1] , **Prof. Mohamad Reza Alsharif**[1], **Prof. Katsumi Yamashita**[2]
Rafael Villa[3], **He Cuiwei**[1], **Prof. Hayao Miyagi**[1]
[1]Department of Information Engineering. Univ. of the Ryukyus. Okinawa, Japan.
carlosengutierrez@yahoo.com.ar, asharif@ie.u-ryukyu.ac.jp, hecuiwei0924@gmail.com, miyagi@ie.u-ryukyu.ac.jp
[2]Graduate School of Engineering. Osaka Prefecture University. Osaka, Japan. yamashita@eis.osakafu-u.ac.jp
[3]Regional Public Goods. InterAmerican Development Bank. Washington DC. USA. rafaelv@iadb.org

Abstract - *In this paper we explore the spatial and temporal properties of a set of news published after a natural disaster by using SOM (Self-organizing Maps). SOM develops a low-dimensional representation of the input data space by mapping high dimensional vectors into a 2-dimensional grid. Training stage produces a visual representation and a set of quantization points that can be considered as groups of spatially related news. Temporal dependency is detected by analyzing SOM units' activation over the time, discovering temporal associations between data items. Our SOM stores information throughout its grid in a way such that space and time structures of the input data set are discovered and stored. First it has no knowledge, but after learning it develops spatial-temporal representations.*

Spatial-Temporal relations can be used for predictive modeling, search of sequential patterns, and mostly used for understanding. In our case, discovered dependencies describe the causes or context that precede a topic, showing how it evolved and moved over the time.

Keywords:, neural networks, self-organizing map, temporal clustering, principal component analysis.

1. Introduction

Data mining can be applied on various sources of data such as on-line newspapers, social networks, blogs, etc; to discover groups or clusters of related events, having as disadvantage that it implies to manage high-dimensional data. If we take in account that each single word represents a variable, it is very complex to process the full set of variables and visualize events' relationships, fortunately, groups of variables often move together in time and space; one reason for this is that more than one variable might be measuring the same driving principle governing the system's behavior.

But besides the events' relationships that can be found by a computation of a "distance" between vectors; one of the most interesting problem is to find an effective and simple method able to discover temporal relations as well.

The purpose of this paper is to create a practical method based on artificial neural networks, to find spatial-temporal representations within raw data. In particular, we use a type of self-organizing map (SOM) called Kohonen's network, which is applied to uncover and visualize the inherent structure and topology of a set of news. If the data form clusters in the input space, i.e. if there are regions with very frequent and at the same time very similar data, the self-organizing process will ensure that the data of a cluster are mapped to a common localized domain in the map. Moreover, the process will arrange the mutual placement of domains in such a way as to capture as much of the overall topology of the cluster arrangement as possible. In this way, even hierarchical clustering can be achieved. The temporal factor is added to the map by the analysis of temporal dependencies between units, with the introduction of a time-dependent matrix that stores unit-to-unit and neighborhood-to-unit temporal relations.

In our work, a set of news published between March 11th and March 18th 2011 (March 11th is sadly remembered as the day where multiple earthquakes triggered a huge tsunami in Japan) are encoded as numeric vectors by using PCA. Secondly, a SOM is trained to build an organized representation of the input space. Next, a second training stage is applied to find temporal clusters. Finally, temporal representations are interpreted and analyzed, showing topics evolving over the time.

2. Input Data Encoding

Our data set contains 421 text files (news available at CNN web site). At this step, our aim is to develop a suitable representation of the input data as a numerical matrix. An implementation in C++ was developed to extract from each file the words, create a dictionary and compute words frequency. Special characters, numbers, symbols, and meaningless words such as conjunctions, prepositions and adverbs were removed. In addition, our implementation includes a Porter stemming algorithm [10]. Stemming is the process for reducing inflected (or sometimes derived) words to their stem, base or root form. The general idea underlying stemming is to identify words that are the same in meaning but different in form by removing suffixes and endings; for instance, words such as "expanded", "expanding", "expand", and "expands" are reduced to the root word, "expand ". The output was a dictionary of words (vector I of 9961 elements) and a matrix X of 421x9961 (news x words), where each

element $x_{i,j}$ is a number equal to the frequency of $word_j$ at news i. As expected, the result is a high-dimensional matrix.

By using principal component analysis is it possible to transform a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. Each principal component is a linear combination of the original variables, and all of them are orthogonal to each other, so there is no redundant information. By this method it is possible to compress the data by reducing the number of dimensions, without much loss of information [8]. Let X and Y be $m \times n$ matrices related by a linear transformation P ($n \times n$); m indicates the observation number with n variables; X is the initial data set and Y is a re-representation of X. PCA re-express the initial data as a linear combination of its basis vectors:

$$Y = XP \tag{1}$$

p_i are column vectors of P.

x_i are row vectors of X.

Each row of Y has the form:

$$y_i = \left[x_i p_1 \cdots x_i p_n\right] \tag{2}$$

We recognize that each coefficient of y_i is a dot product of x_i with the corresponding column in P, in other words, the j^{th} coefficient of y_i is a projection on to the j^{th} column of P. By assuming linearity, the problem reduces to find the appropriate change of basis, the columns vectors p_i of P, also known as the principal components of X.

But first, let's define S_x as the covariance matrix of X. S_x is a simple way to quantify redundancy by calculating the spread between variables. X is in mean deviation form because the means have been subtracted off or are zero.

$$S_x = \frac{1}{n-1} X^T X \tag{3}$$

S_x is a square symmetric $n \times n$ matrix. Its diagonal terms are the variance of particular variables. The off-diagonal terms are the covariance between variables. From S_x the eigenvectors with their corresponding eigenvalues are calculated. Eigenvectors are a special set of vectors associated with a linear system of equations (i.e., a matrix equation), also known as characteristic vectors, proper vectors, or latent vectors [11]. Each eigenvector is paired with a corresponding factor so-called eigenvalue by which the eigenvector is scaled when multiplied by its matrix. A non-zero vector p_i is an eigenvector of the covariance matrix S_x if there is a factor λ_i such that:

$$S_x p_i = \lambda_i p_i \tag{4}$$

Generalizing:

$$S_x P = \Lambda P \tag{5}$$

The full set of eigenvectors is as large as the original set of variables. In PCA, the eigenvectors of S_x are the principal components of X. Matrix P ($n \times n$) contains n eigenvectors, arranged in a way such as p_1 is the principal component with the largest variances (the most important, the most "principal"); p_2 is the 2nd most important, and so on. In addition, the eigenvalues contained in diagonal matrix Λ are arranged in descending order $\lambda_1 > \lambda_2 > \cdots > \lambda_n$ and they represent the variance of X captured by the principal components. This last relation is used for dimensionality reduction.

$$S_x p_i = \sigma_i^2 p_i \quad \text{with} \quad \sigma_1^2 > \sigma_2^2 > \cdots > \sigma_n^2 \tag{6}$$

From matrix X, after subtracting off the mean for each variable, the covariance matrix and its principal component are computed. As result, the full set of 9961 principal components, matrix P, and the corresponding 9961 eigenvalues, diagonal matrix Λ, are obtained.

Principal components with larger associated variances have important dynamics; while those with lower variances represent noise. It is common to consider only the first few principal components whose variances exceed 80% of the total variance of the original data. In our case, the first 362 principal components (out of 9961) describe almost all the variability of the data set. Let's take, for instance, the $l = 362$ largest eigenvalues $\lambda_1 > \lambda_2 > \cdots > \lambda_l, (l < n)$; and truncate the matrix P at column l. That means, we are taking only the first l principal components of P. It implies a strong dimensionality reduction, in the order of 96%. This reduced ($n \times l$) matrix is called $\hat{P}$. The original data set X ($m \times n$) is re-expressed then as Y ($m \times l$) by using matrix $\hat{P}$ ($n \times l$):

$$Y = X \hat{P} \tag{7}$$

Equation (7), as a dot product, shows how matrix Y compresses X and contains the distribution of the news along the most important l components. Rows of matrix Y will be the input vectors of the Self-Organized Map used to discover spatial-temporal relations.

3. Spatial Relations Uncovering by Self-organizing maps (SOM).

At this stage, our work seeks to generate an understandable spatial map of the input space. To achieve it, we use matrix Y to feed a 2-dimensional 10x10 SOM network; each row y_i of Y represents a news. Considering the 421 training vectors, we assume that a grid of 100 units may produce a reasonable amount of quantization points and a suitable visualization. Generally, in SOM, variables are normalized by dividing each column of Y by its standard deviation, however, in our case; we consider that representation by PCA has homogenized the input data.

A SOM consists of components called units, cells or neurons. Associated with each unit there is a weight vector of the same dimension as the input data and a position vector in the map space. In learning stage an input vector is presented to the SOM at each step. These vectors constitute the "environment" of the network. SOM includes a competitive and unsupervised learning able to find clusters from the input data. Competitive learning means that a number of units are comparing the same input signals with their internal parameters, and the unit with the best match, the winner, is tuned itself to that input affecting also its neighbors. Therefore, different units learn different aspects from the input.

Some requirements are needed for self-organization: i) the units are exposed to a sufficient number of different inputs; ii) for each input, the synaptic input connections to the exited group of units are only affected; iii) similar updating is imposed on many adjacent neurons; iv) the resulting adjustment is such that it enhances the same responses to a subsequent, sufficiently similar input.

The most popular model of SOM is the model proposed by Teuvo Kohonen[5] called Kohonen network. Kohonen algorithm introduces a model that is composed of two interacting subsystems. One of these subsystems is a competitive neural network that implements the winner-take-all function. The other subsystem modifies the local synaptic plasticity of the neurons in learning [6]. Kohonen learning uses a neighborhood function ϕ, whose value $\phi = (i,k)$ represents the strength of the coupling between unit i and unit k during the training process. The learning algorithm is as follows [6]:

- Start: n-dimensional weight vectors $w_1, w_2, ..., w_m$ for the m computing units are selected at random. An initial radius of the neighborhood r, a learning constant η, and a neighborhood function ϕ are selected. The neighborhood function $\phi = (i,k)$ is defined as:

$$\phi(i,k) = \exp\left(-\left(\frac{|i-k|}{r}\right)^2\right) \qquad (8)$$

Where i is the position of the i^{th} unit and k is the position of the unit with the maximum excitation. The neighborhood function ϕ changes according to a schedule, producing larger corrections at the beginning of the training that at the end.

- Step 1: Select an input vector y using the desired probability distribution over the input space.
- Step 2: The unit k with the maximum excitation is selected (that is, for which the Euclidean distance between w_i and y is minimal, $i = 1,2,...,m$).
- Step 3: The weight vectors are updated using the neighborhood function ϕ and the following rule:

$$w_i \leftarrow w_i + \eta\phi(i,k)(y - w_i) \text{ for } i = 1,2,...,m \qquad (9)$$

- Step 4: Stop if the maximum number of iterations has been reached; otherwise modify η and ϕ as scheduled and continue with step 1.

By repeating this process several times, it is expected to arrive to a uniform distribution of weight vectors for the input space. We perform 3000 iterations, at each one the complete set of training vectors is entered into the network once. The result is a nonlinear projection of the input space onto a map (Figure 1). A main property of the map is that, the distance relationships between the input data are preserved by their images in the map as faithfully as possible. However, a mapping from a high-dimensional space to a lower-dimensional one will usually distort most distances and only preserve the most important neighborhood relationships between data items. Figure 1 shows the SOM map after training and the 95 quantization points generated as gray dots. The size of the dots represents the amount of input vectors captured by weight vectors. For instance, w_{36} at coordinates $(6,4)$ captures more inputs than w_{18} at $(8,2)$. Within a quantization point the news are spatially related; therefore, a quantization point is by itself a group and represents a sub-set of input vectors.

Convergence of the network is evaluated empirically; it gets stable state after 3000 iterations, at this stage the map doesn't change and weight vectors experiment very small updates.

The obtained SOM mainly reflects metric distance relations between input vectors. In order to give a semantic-meaningful component to the map, we present, after the spatial training the context where the input data may be located, in that way the map reflects logic or semantic similarities. Context is a background, environment, framework, setting, or situation surrounding an event or occurrence. In linguistic it is defined as words and sentences that occur before or after a word or sentence and imbue it with a particular meaning.

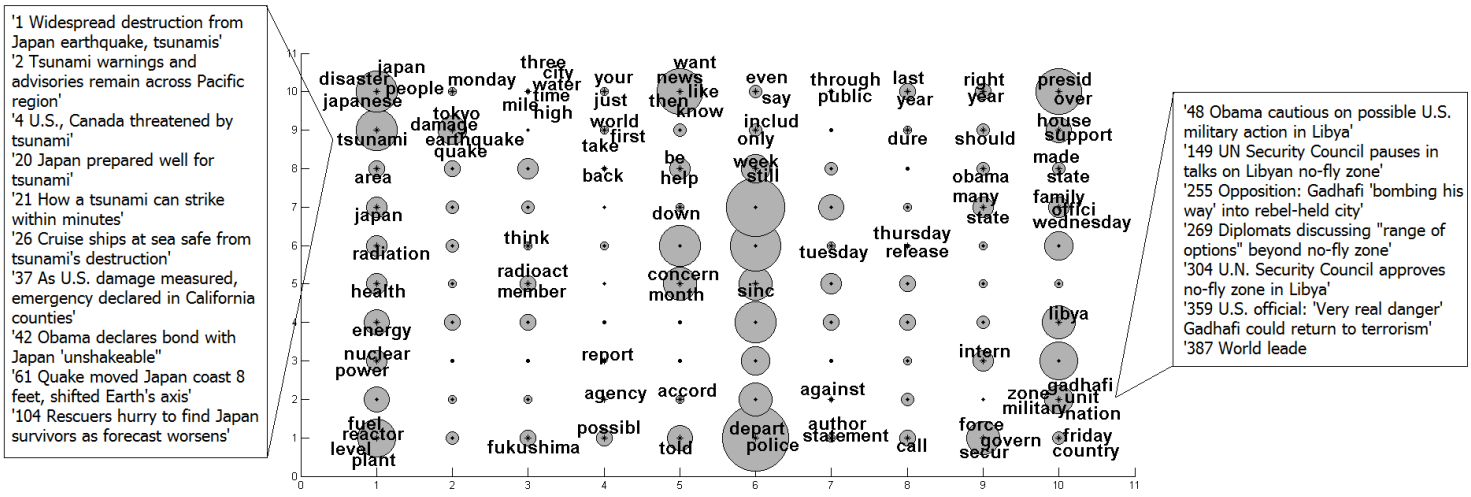

Figure 1. SOM map after training and the 95 quantization points generated as gray dots. The size of the dots represents the amount of input vectors captured by weight vectors. Most frequent words are mapped to give a semantic characteristic to the discovered structure.

We assume that the most frequent words have strong correlations with contexts that surround events described in our set of news. Hence we choose to represent each most frequent *k-th* word by a *n*-dimensional vector, whose *k-th* component has a fixed value equal to *k-th* word's total frequency and whose remaining components are zero. Each vector then is compressed by equation (7) that reduces their dimensionality by using l principal components. The words are presented to the network and the strongest responsive units are detected and labeled with the words. The responses on the map show how the network captured the spatial relations among the news. News related to earthquake-tsunami in Japan are distributed on the left side, while those related to Middle East and Libya on the right. Middle units captured a variety of topics, unit w_6 at $(6,1)$ for example, captured several news related to crime.

Earthquake-tsunami news are differentiated in sub-categories, corresponding to more specialized items such as radiation, health, energy, etc. The labels uncover the semantic relation between items; they show the contexts where the news items are located. Each news incorporates frequent words in its representation as vector; with a sufficient amount of training the inputs leave memory traces on the same units at which later the words individually converge.

Therefore, a meaningful topographic spatial map is obtained by adding 100 most frequent words, showing logical similarities among inputs. It is possible to add more words which will enrich the map and will add more details to the semantic relations, but for simplicity and good visualization, we chose only 100 words.

4. Temporal Learning:

Previous section described how SOM's units stored spatial patterns. At this section temporal dependency analysis is performed to find significant temporal associations between data items or events. The main idea is to identify temporal sequences of spatial patterns that are likely to occur one after another.

Our spatially-trained SOM is fed once more with the input data set and temporal sequences of activated units are monitored and stored in a time-dependent matrix. The temporal aspect comes from movements or changes of the input data. Every time a unit *k* fires, our model creates a vector *d* of dimension *m*, where *m* is the total amount of SOM units. The element *d(k)* has a fixed value equal to 1 and the remaining components are zero. Vectors *d* are the inputs of the time-dependent matrix denoted as *T*.

In order to analyze the temporal proximity of units, the matrix *T* is created with *m* rows and *m* columns and its elements are initialized to 0. Rows correspond to units activated at time *(t-1)* and columns to the units at time *(t)*. Our model memorizes the previously activated unit, in a way such that for an input *d* at time *t*, the matrix *T* is updated by increasing *T(i,j)* an amount equal to *a*, where *i* is the unit fired on time *(t-1)*, and *j* is the unit fired on time (*t)*. The value added to *T(i,j)* corresponds to a transition value from the past unit to the current unit. Neighbors of unit *i* are also considered, the reasoning is that if unit *j* is frequently followed by unit *i*, the model considers that there is a high probability that neighbors of *i* follow unit *j* as well. For this last case, matrix *T* is updated by a scale down increment (*a* β)* in elements *T(Ni,j)*, where *Ni* denotes neighbors of *i*.

In that way time-dependent matrix *T* receives inputs and learns temporal relations among units over the time (Figure 2).

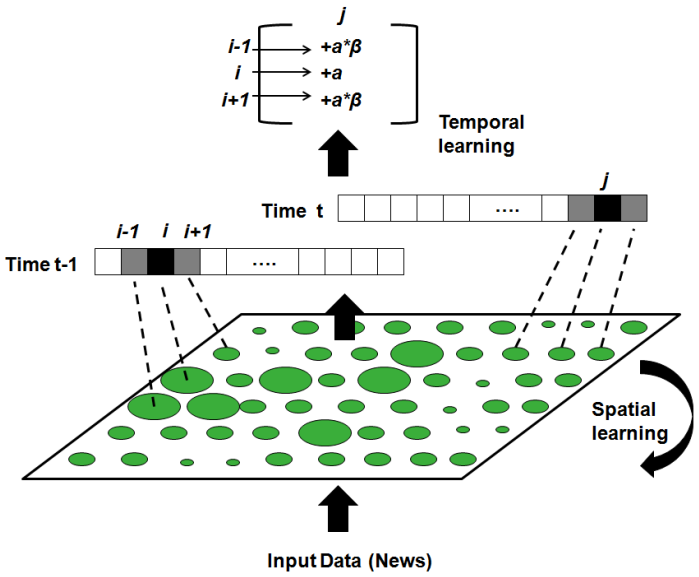

Figure 2. After spatial training, units activated are represented as vectors. They are the inputs of a time-dependent matrix T that learns over the time temporal relations among units

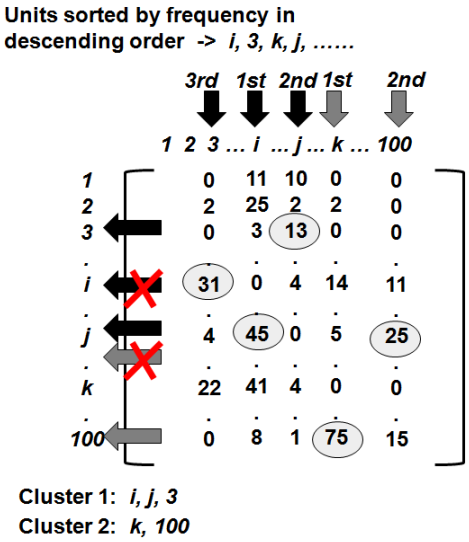

Figure 3. Temporal clustering example. 1st cluster start with column i (most frequent unit), taking most-connected unit j after 1st loop. During 2nd loop unit j takes unit 3. This last unit takes unit i at 3rd loop, but it is already in the cluster, therefore cluster 1 is closed.

5. Temporal Clustering:

After *T* is built, the process continues by clustering matrix *T*. The aim is to generate coherent clusters, which means, we seek to detect clusters where the units have high probability to follow each other through the time; these clusters are called temporal clusters and they contain groups of units that are likely to represent the evolution on time of a certain topic or event.

Vector *C* of dimension *m*, where *m* is the total amount of SOM units, stores the number of news pooled for each unit of the SOM. *C* and matrix *T* are used by the algorithm described below to detect temporal clusters. Figure 3, in addition, illustrates the process:

- Step1: Find from vector *C* the most frequent unit that is not yet part of a cluster. The most frequent unit is the one with the highest corresponding value in *C*.
- Step 2: Pick the unit that is most-connected to the most frequent unit. The model finds the most-connected unit by finding the highest value in the column of matrix *T* that corresponds to the current unit. Add the most connected unit to the cluster only if it is not part of a cluster.
- Step 3: Repeat step 2 for the most connected unit. Then recursively computes step 2 on its most connected unit, and so on, until no new unit is added.
- Step 4: All these units are added to a new temporal cluster.
- Step 5: Go to step 1 and find the most frequent unit that is not yet part of a cluster.

Once temporal clusters are formed, they are interpreted as frequent news topics and events evolving over the time. Each unit captures a subset of news; therefore, the five most frequent words are taken from each unit as a description of the unit's topic, results are shown in table below:

Temporal Cluster (Units coordinates frequently activated, in temporal order)	5 most frequent words for each unit in temporal order
(5,7), (6,2), (6,1)	'lodg', 'sweat', 'trial', 'particip', 'ray' 'court','charg', 'attorney', 'case', 'judg' 'polic', 'investig', 'depart', 'alleg', 'suspect'
(2,9), (1,9), (6,7)	'tokyo', 'earthquak', 'power', 'japan', 'quak' 'tsunami', 'japan', 'earthquak', 'warn', 'area' 'moon', 'year', 'last', 'look', 'zune'
(6,8), (5,10)	'seavey', 'bike', 'week', 'appl', 'kate' 'your', 'like', 'want', 'peopl', 'just'
(5,1), (10,10)	'investig', 'crash', 'driver', 'polic', 'william' 'aristid', 'haiti', 'spend', 'return', 'elect'
(9,1), (1,1), (6,3), (6,4)	'bahrain', 'forc', 'govern', 'secur', 'saudi' 'reactor', 'plant', 'radiat', 'fuel', 'nuclear' 'yale', 'school', 'clark', 'polic', 'sentenc' 'polic', 'accord', 'offic', 'baghdad', 'video'
(10,2), (2,4), (3,8), (10,4), (9,7), (10,3)	'zone', 'gadhafi', 'council', 'unit', 'resolut' 'nuclear','plant', 'power', 'japan', 'disast' 'earthquak', 'japan', 'school', 'might', 'peopl' 'gadhafi', 'govern', 'libyan', 'presid', 'libya' 'state', 'unit', 'hispan', 'medic', 'marijuana' 'gadhafi', 'zone', 'forc', 'libyan', 'libya'
(7,1), (4,1), (3,1), (10,6)	'palestinian', 'isra', 'author', 'hama', 'gaza' 'reactor', 'meltdown', 'nuclear', 'possibl', 'radiat' 'reactor', 'plant', 'nuclear', 'explos', 'tuesday' 'afghanistan', 'diplomat', 'petraeus', 'pakistan', 'court'
(1,6), (1,3)	'radiat', 'japan', 'airlin', 'nuclear', 'flight' 'nuclear', 'plant', 'power', 'energi', 'reactor'

(3,2), (8,1), (9,3)	'plant', 'reactor', 'nuclear', 'japan', 'agenc' 'protest', 'forc', 'govern', 'demonstr', 'secur' 'forc', 'bahrain', 'govern', 'gadhafi', 'intern'
(2,2), (1,7)	'power', 'nuclear', 'reactor', 'plant', 'daiichi' 'japan', 'food', 'govern', 'japanes', 'spaniard'
(2,1), (7,4)	'reactor', 'plant', 'japanes', 'report', 'nuclear' 'offici', 'defens', 'peopl', 'rain', 'civil'
(8,6), (7,6)	'releas', 'record', 'anonym', 'execut', 'donat' 'right', 'inmat', 'maryland', 'bill', 'california'
(8,9), (8,7)	'head', 'earli', 'educ', 'start', a'childhood' 'obama', 'conyer', 'presid', 'kenni', 'critic'
(7,3), (4,10)	'lucia', 'attack', 'accord', 'anti', 'baker' 'citi','just','parad', 'your', 'peopl'

Table 1. Main temporal clusters detected by proposed model.

Temporal clusters represent a high-level perception of meaning, knowledge, logic, etc, over the time. They can be interpreted as an image or memory of frequent sequences of topics. Main topics, those that remain in the time, come to light, while volatile topics are not displayed. For instance, temporal cluster of units (1,6), (1,3) shows that topic ('radiat', 'japan', 'airlin', 'nuclear', 'flight') follows frequently to topic ('nuclear', 'plant', 'power', 'energi', 'reactor'). We can infer that, during the disaster, there was a transition from issue "nuclear-radiation-flights" to issue "energy-power-reactor", and that transition was frequently mentioned.

Our SOM developed automatically the formation of a spatial-temporal "memory" in a way that its layout forms an image of the most important relations.

6. Conclusion and future work

Our application demonstrates that plain text sources can be represented as a numerical matrix, compressed and transformed to serve as input data for a SOM network. A SOM has been trained producing a spatial representation of the news set into a 2 dimensional map. This representation is a finite number of quantization points that group similar input vectors. Frequent words on map enabled to form a semantic structure. Time dimension was considered on SOM temporal learning, where groups of units were discovered having a high time-dependency. Temporal clusters detection was possible by the utilization of a time-dependent matrix that stores the transitions from a SOM unit to another; this matrix is the model's perception of frequent events over the time.

Although our data set was relatively small, the proposed model was able to discover temporal dependencies. We believe that results are improved and determined largely by what model is exposed to. Enough input must change and flow continuously through time for a suitable learning. The model can be scaled exponentially with diverse input data without complexity due its finite set of quantization points. SOM also can be modified as a self-growing map working "on demand".

Our time-dependent matrix also can be modified assigning a memory to it, in a way that it doesn't remember only the last fired unit at time *(t-1)*, but the last *k* units fired at times *(t-1), (t-2), (t-3), …, (t-k)* , expanding its ability to detect unknown temporal relations. Another improvement to consider is that neighbors of unit *i* fired at time *(t-1)* that follow unit *j* fired at time *(t)* are considered, but we do not evaluate the potential temporal relation among neighbors of *i* with neighbors of *j*.

In addition, when matrix *T* is updated by a scale down increment *(a*β)* in elements *T(Ni,j)*, where *Ni* denotes neighbors of *i*, we assign empirical amounts to transition value *a* and parameter *β*. If a memory is provided to matrix *T*, *a* and *β* should vary on time. Temporal clustering algorithm also can be improved considering, for example, not only the most-connected unit, but the 2nd most-connected, the 3rd most-connected.

Temporal clusters can be used to make predictions. The model computes for a new input *x* a spatial distribution on its *m* units, and a temporal distribution on its *c* temporal clusters.

Our application emphasizes the spatial-temporal arrangement of the units and the segregation of the information into separate areas. Temporal clusters give an idea of how frequent events evolve over the time, although in a high level it does completely on unsupervised way.

7. References

[1] C.E. Gutierrez, M.R. Alsharif, H. Cuiwei, M. Khosravy, R. Villa, K. Yamashita, H. Miyagi, Uncover news dynamic by principal component analysis. Shanghai, China, ICIC Express Letters, vol.7, no.4, pp.1245-1250, 2013.

[2] C.E. Gutierrez, M.R. Alsharif, H. Cuiwei, R. Villa, K. Yamashita, H. Miyagi, K. Kurata, Natural disaster online news clustering by self-organizing maps. Ishigaki, Japan, 27th SIP symposium, 2012.

[3] C.E. Gutierrez, M.R. Alsharif, R. Villa, K. Yamashita, H. Miyagi, Data Pattern Discovery on Natural Disaster News. Sapporo, Japan, ITC-CSCC, ISBN 978-4-88552-273-4/C3055, 2012.

[4] H. Ritter, T. Kohonen, Self-Organizing Semantic Maps. Biological Cybernetics. Springer-Verlag 61, pp. 241-254, 1989.

[5] T. Kohonen, Self-Organization and Associative Memory. Berlin, Springer, 1984.

[6] R. Rojas, Neural Networks. Berlin, Springer-Verlag, 1996.

[7] X. Wu, V. Kumar, J.R. Quinlan, Top 10 algorithms in data mining. London , Springer-Verlag, 2007.

[8] L. I. Smith, A tutorial on Principal Components Analysis. 2002.

[9] J. Shlens, A tutorial on Principal Component Analysis: Derivation, Discussion and Singular Value Decomposition. 2003.

[10] M.F. Porter, M.F. An algorithm for suffix stripping, Program, vol.14, no.3, pp.130–137, 1980.

[11] M. Marcus and H. Minc, Introduction to linear algebra. New York: Dover, pp.145-146, 1988.

[12] R. Yan and L. Kong, Timeline generation through evolutionary trans-temporal summarization. Conference on Empirical Methods in Natural Language Processing, Edinburg, Scotland, pp.433–443, 2011.

[13] Y. Zhang and L. E. Ghaoui, Large-Scale Sparse Principal Component Analysis with Application to Text Data. Advances in Neural Information Processing Systems (NIPS). 2011.

[14] O. Vikas, A. K. Meshram, G. Meena and A. Gupta, Multiple document summarizations using principal component analysis incorporating semantic vector space model. Computational Linguistics and Chinese Language Processing. vol.13, no.2, pp.141-156, 2008.