Journal of Computer Science and Cybernetics, V.31, N.3 (2015), 215–229
DOI: 10.15625/1813-9663/31/3/6452

IMPROVING THE QUALITY OF SELF-ORGANIZING MAP BY "DIFFERENT ELEMENTS" COMPETITIVE STRATEGY

LE ANH TU

Thai Nguyen University of Information and Communication Technology;
anhtucntt@gmail.com

Abstract. A Self-Organizing Map (SOM) has good quality when both of its measures, quantization error (QE) and topographic error (TE), are small. Many researchers have tried to reduce these measures by improving SOM's learning algorithm, however, most results only decrease either QE or TE. In this paper, a method to improve the quality of the map obtained when the SOM's learning algorithm ended is proposed. The proposed method re-adjusts weight vector of each neuron according to cluster's center that neuron represents and optimizes clusters by "*different elements*" competitive strategy. In this method, QE always decreases each time the competition "*different elements*" occurs between all neurons, TE may reduce when the competition "*different elements*" occurs between adjacent neighbors. The experiments are performed on assumed datasets and real datasets. As the results, the average reduction ratio of QE is from 50% to 60%, TE gets the average reduction ratio from 10% to 20%. This reduction ratio is larger than some other solutions but does not need to adjust the parameters for each specific dataset.
Keywords. self-organizing map, competitive learning, different elements, quantization error, topographic error.

1. INTRODUCTION

The SOM neural network was proposed by Teuvo Kohonen in 1980s [16]. This is a feedforward neural network model, using an unsupervised competitive learning algorithm. The SOM allows mapping data from multi-dimensional space to less dimensional one (normally 2 dimensions), which makes up the feature map of the data. So far, there have been many different variations of SOM proposed [5] and there are many studies showing that feature map's quality of SOM depends greatly on the initialization parameters such as: Kohonen layer size, numbers of training and neighboring radius [7, 10, 16, 19, 29].

The quality of SOM's feature map is evaluated based on two criteria, learning quality and projection quality primarily [3, 13, 23, 27]. In particular, the learning quality is determined by measuring the QE (demonstrates the data representation accuracy) [4, 16] and the projection quality determined by measuring the TE (demonstrates the topology preservation) [2, 15, 20].

In fact, the method of "*trying error*" is used to choose suitable parameters [16]. According to Chattopadhyay [6], with a particular dataset, the network size is chosen by "*trying error*" until small QE and TE achieved. Polzlbauer indicates technical correlation between QE and TE [24], TE usually increases when the QE decreases; besides, in case Kohonen layer's size larges, QE reduces but TE rises (i.e. increasing Kohonen layer's size can lead to the map's topographic deformation) and whereas when Kohonen layer's size is too small, TE may be not trustful. The use of small neighboring

radius also leads to reduce the QE, if neighboring size is smallest, QE will achieve the minimum value [26].

Beside the "*trying error*" method to determine an appropriate network configuration, the researches for improving the learning algorithm are also developed by other researchers. Germen [8, 9] optimized QE by integrating parameter "*hit*" when updating the weights of the neurons. The term "*hit*" means the numbers of excitation of a neuron. As a consequence, the neurons representing major samples will be adjusted less than the neurons representing the minor samples (to ensure no loss of information). Neme [21, 22] proposed model SOM with selective refractoriness allows optimizing TE. In this model, the neighboring radius of BMU does not reduce gradually during the learning process, each training time, each neuron in the neighboring radius of BMU will decide whether the next training is influenced by the BMU again or not. Kamimura [14] integrated "*firing*" rate into distance function in order to maximize input information. The "*firing*" rate represents the importance of each feature compared to the remaining features. His method reduces both QE and TE, however, its limitation is each dataset needs to do "*trying error*" to achieve the appropriate "*firing*". Another research, Lopez-Rubio [18] gave out the cause of the TE due to the self-intersections (Fig.3) as in following definition: *A map is self-intersected if and only if there two triples of adjacent units* $\{i, j, k\}$ *and* $\{r, s, t\}$ *that satisfy two conditions:* $\{i, j, k\} \cap \{r, s, t\} = \emptyset$ *and* $(\Delta w_i w_j w_k) \cap (\Delta w_r w_s w_t) \neq \emptyset$ *where,* $\triangle \mathbf{abc}$ *triangle defined by vertices* $\mathbf{a}, \mathbf{b}, \mathbf{c} \in R^D$,

$$\Delta abc = \{(1 - u - v)\,\mathbf{a} + u\mathbf{b} + v\mathbf{c} | 0 \leq u + v \leq 1\}$$

Thereby, to reduce the TE, self-intersections have to be removed. He proposed the solution to detect self-intersections and redid the learning steps which caused it. His solution has disadvantages that when TE decreases, QE increases.

Obviously, trying to adjust learning algorithm to reduce both QE and TE is a difficult task. Thus, our solution is to re-adjust obtained map after the learning algorithm ends. In the competitive learning method [11, 25], the samples represented by each neuron are considered as a cluster, hence, the weight vector of the neuron will best represent for samples if it is the codebook vector of the cluster. In essence, a large QE is caused by the big difference of each data sample from its winner neuron (eq(4)), so to reduce the QE, the weight vectors must be adjusted according to the codebook vectors of the clusters and the clusters must be optimized according to the new weight vectors. This optimizing cluster method is called the competition "*different elements*". The "*different elements*" competitive process will promote weight vector of each neuron to move closer towards the weights of adjacent neighbors. This limits self-intersections status [18], so that reduces the TE.

The remaining of the paper includes: part 2 presents an overview of SOM and the quality measures of feature map; part 3 presents our solution; part 4 offers experimental results and final part is conclusions.

2. SOM NEURAL NETWORK AND FEATURE MAP QUALITY

2.1. An overview of SOM

The SOM neural network includes an input signal layer which is fully connected to an output layer called Kohonen layer (Figure.1). Kohonen layer is often organized as a two dimensional matrix of neurons. At t training times, a sample v is used to train the network. The training algorithm performs three steps:

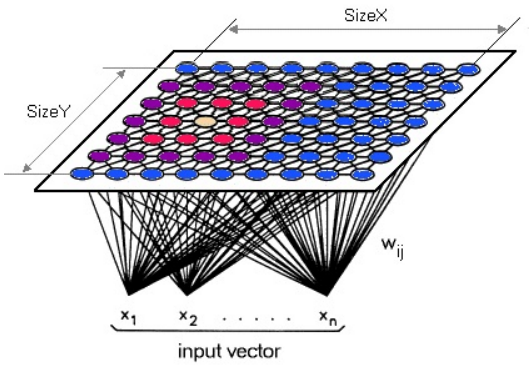

Figure 1: Illustrations of SOM.

- *Step 1:* Finding the best matching unit (BMU) with v as the eq(1) .

$$dist = ||v - \mathrm{w}_c|| = \min_i \{||v - \mathrm{w}_i||\} \tag{1}$$

- *Step 2:* Calculating the neighboring radius of BMU as the eq(2).

$$N_c(t) = N_0 \exp\left[-\frac{t}{\lambda}\right] \tag{2}$$

where, N_0 is the initial neighboring radius;

$$\lambda = \frac{K}{\log(N_0)}$$

is the time parameter, with K is the numbers of iterations.

- *Step 3*: Updating the weight vector of BMU, and neurons in the neighboring radius of BMU as the eq(3).

$$\mathrm{w}_i(t+1) = \mathrm{w}_i(t) + N_c(t)\, h_{ci}(t)\, [v - \mathrm{w}_i(t)] \tag{3}$$

where,

$$h_{ci}(t) = \exp\left[-\frac{||r_c - r_i||^2}{2{N_c}^2(t)}\right]$$

is the interpolation function over learning times, with $\|r_c - r_i\|^2$ is the distance from BMU (neuron c) to neuron i in the Kohonen layer.

2.2. Quality measures

Quantization Error [16]: the average difference of inputs compared to their corresponding $BMUs$.

$$QE = \frac{1}{T}\sum_{t=1}^{T} \|x(t) - \mathrm{w}_c(t)\| \tag{4}$$

where, $w_c(t)$ is the weight vector of BMU corresponding to $x(t)$, T is the total number of data samples.

Topographic Error: the numbers of the samples whose the first best matching unit (BMU_1) and the second best matching unit (BMU_2) are not adjacent [15, 20].

$$TE = \frac{1}{T}\sum_{t=1}^{T} d\left(x\left(t\right)\right) \tag{5}$$

where, $d(x(t)) = 1$ if BMU_1 and BMU_2 of $x(t)$ are adjacent, and $d(x(t)) = 0$, vice verse.

Topographic Product (TP): assess the neighborhood relation preservation in the map [3]. However, TP is only reliable for linear datasets [28].

$$TP = \sum_{i=1}^{n\times m} H_i \tag{6}$$

where, $H_i = 1$ if k nearest neighbors of neuron i, which have the identical weight vector, $n \times m$ is the size of Kohonen layer.

Distortion Measure (DM): the overall quality of the SOM neural network is evaluated by energy function E_d [17]. E_d is used to pick out the best map from different trainings with the same dataset. However, Heskes [12] shows that E_d can only be optimized as training set that is finite and neighboring radius fixed.

$$E_d = \sum_{t=1}^{T}\sum_{i=1}^{q} h_{ci}\left(t\right)\left\|x\left(t\right) - \mathrm{w}_i\left(t\right)\right\| \tag{7}$$

with q is the number of neurons in the neighboring radius of the BMU at iteration t.

Indeed, QE and TE are two main measures used to assess the quality of feature map [6]. The next section presents the solution to reduce the QE and TE.

3. "DIFFERENT ELEMENTS" COMPETITIVE STRATEGY

Obviously, after the training process, each neuron in the Kohonen layer will represent a data cluster including closest samples to weight vector of the neuron. So, the training dataset is divided into s subsets corresponding to s neurons (with $s = m \times n$, where $m \times n$ is the size of Kohonen layer). Suppose I is training dataset, it yields

$$I = \{I_1, I_2, ..., I_s\}$$

where, I_i is a subset including samples represented by neuron i (with $i = 1..s$).

Call Q_i the difference of neuron i (total of the distance of the samples of I_i to weight vector w_i):

$$Q_i = \sum_{v=1}^{p} d\left(x_v, w_i\right) \tag{8}$$

where,

$$d\left(x_v, w_i\right) = \left\|x_v - w_i\right\|$$

with $x_v \in I_i, p = |I_i|$ is the number of samples represented by neuron i.

The eq(4) is equivalent to the eq(9) below:

$$QE = \frac{1}{T}\sum_{i=1}^{s} Q_i \tag{9}$$

The eq(9) shows that: QE is minimized if Q_i is minimized, with $\forall i = 1..s$.
Call c_i the codebook vector of I_i (c_i is closest to all samples of I_i):

$$c_i = \frac{1}{p}\sum_{v=1}^{p} x_v \tag{10}$$

Let Q_i^c the total of the distance of the samples of I_i to the c_i.

$$Q_i^c = \sum_{v=1}^{p} d\left(x_v, c_i\right) \tag{11}$$

Hence, Q_i is minimized if it satisfies the eq(12)

$$Q_i = Q_i^c \Leftrightarrow \sum_{v=1}^{p} d\left(x_v, w_i\right) = \sum_{v=1}^{p} d\left(x_v, c_i\right) \tag{12}$$

In other words, Q_i is minimized if and only if $\mathrm{w}_i = c_i$, with $\forall i = 1..s$
From all above, a definition about the smallest quantization error is proposed:

Definition 1. *Quantization error of self-organizing map is the smallest if and only if* $\mathrm{w}_i = c_i$*, with* $\forall i = 1..s$*, where* w_i *is the weight vector of neuron* i*;* c_i *is the codebook vector of* I_i*, including samples represented by neuron* i*.*

Therefore, to reduce the QE we assign $w_i = c_i$, with $i = 1..s$. However, this leads to the consequence that some samples have to change its representative neuron, because it fits better with another neuron (compared with the neuron to which it belongs), i.e. the elements of each subset I_i need to be redefined. The samples which need to change representative neuron are called "*different elements*", as the following definition:

Definition 2. x *is called "different elements" of neuron* i *to neuron* j *(with* $\forall j \neq i$*) if and only if* $x \in I_i$ *and* $d\left(x, \mathrm{w}_i\right) > d\left(x, \mathrm{w}_j\right)$*.*

In the Figure.2, x_1 is the "*different elements*" of neuron i to neuron j, with $x_1 \in I_i$: $d\left(x_1, \mathrm{w}_i\right) > d\left(x_1, \mathrm{w}_j\right)$; x_2 is the "*different elements*" of neuron i to neuron k, with $x_2 \in I_i$: $d\left(x_2, \mathrm{w}_i\right) > d\left(x_2, \mathrm{w}_k\right)$; $x_3 \in I_i$ is not the "*different elements*" of neuron i to neuron g because the condition $d\left(x_3, \mathrm{w}_i\right) > d\left(x_3, \mathrm{w}_g\right)$ is not satisfied.

From above definition results in the following theorem:

Theorem. *Give* x *is a "different elements" of neuron* i *to neuron* j *(with* $x \in I_i, i \neq j$*), we have* $QE^* < QE$ *if and only if* $I_i = I_i \backslash x$ *and* $I_j = I_j \cup x$*. In which* QE *is the quantization error before removing sample* x *from set* I_i *and updating* x *to* I_j*,* QE^* *is the quantization error achieved after removing sample* x *from set* I_i *and updating* x *to* I_j*.*

Proof.

Eq(9) $\Leftrightarrow QE = \frac{1}{T}\left(Q_1 + Q_2 + .. + Q_i + .. + Q_s\right)$ Let

$$Q = Q_1 + Q_2 + .. + Q_i + .. + Q_s = \overline{Q} + Q_i + Q_j \tag{13}$$

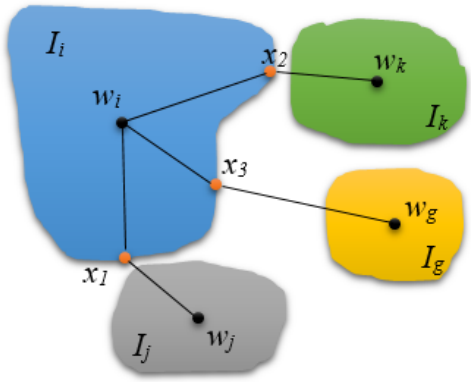

Figure 2: Illustrations of the "*different elements*" of neuron i.

with

$$\overline{Q} = \sum_{k=1,k\neq i,k\neq j}^{s} Q_k$$

Let $\overline{Q}_i = Q_i - d\left(x, \mathrm{w}_i\right)$ is the difference of the neuron i on the set $\overline{I}_i = I_i \backslash x$

$$\Leftrightarrow Q_i = \overline{Q}_i + d\left(x, \mathrm{w}_i\right) \tag{14}$$

Let $\overline{Q}_j$ is the difference of the neuron j on the set $\overline{I}_j = I_j \cup x$

$$\overline{Q}_j = Q_j + d\left(x, \mathrm{w}_j\right) \tag{15}$$

Calling Q^* the total of the difference of all neurons after removing sample x from set I_i and updating x to set I_j, yields:

$$Q^* = \overline{Q} + \overline{Q}_i + \overline{Q}_j \tag{16}$$

From the hypothesis

$$x \in I_i : d\left(x, \mathrm{w}_i\right) > d\left(x, \mathrm{w}_j\right)$$

with $i \neq j$, it yields:

$$\overline{Q}_i + d\left(x, \mathrm{w}_i\right) > \overline{Q}_i + d\left(x, \mathrm{w}_j\right) \tag{17}$$

Replacing (14) to (17), results in:

$$Q_i > \overline{Q}_i + d\left(x, \mathrm{w}_j\right)$$

$$\Leftrightarrow Q_i + Q_j > \overline{Q}_i + d\left(x, \mathrm{w}_j\right) + Q_j \tag{18}$$

Replacing (15) to (18), yields:

$$\overline{Q}_i + \overline{Q}_j < Q_i + Q_j$$

$$\Leftrightarrow \overline{Q} + \overline{Q}_i + \overline{Q}_j < \overline{Q} + Q_i + Q_j \tag{19}$$

Replacing (13) and (16) to (19), results in the following:

$$Q^* < Q \Leftrightarrow QE^* < QE \tag{20}$$

Lemma. *Give x is "different elements" of neuron i ($x \in I_i$) to two neurons j and k (wih $i \neq j, i \neq k, j \neq k$), we have*

$$QE^*_{(j)} < QE^*_{(k)}$$

if and only if $d(x, \mathrm{w}_j) < d(x, \mathrm{w}_k)$, with $QE^*_{(j)}$ is the quantization error achieved if x is updated to set I_j; $QE^*_{(k)}$ is the quantization error achieved if x is updated to set I_k.

Proof.

Calling $Q^*_{(j)}$ the total of the difference of all neurons if x is updated to I_j, yields:

$$Q^*_{(j)} = Q - d(x, \mathrm{w}_i) + d(x, \mathrm{w}_j) \tag{21}$$

Calling $Q^*_{(k)}$ the total of the difference of all neurons if x is updated to I_k, results in:

$$Q^*_{(k)} = Q - d(x, \mathrm{w}_i) + d(x, \mathrm{w}_k) \tag{22}$$

From hypothesis

$$d(x, \mathrm{w}_j) < d(x, \mathrm{w}_k)$$

yields:

$$Q^*_{(j)} < Q^*_{(k)} \Leftrightarrow QE^*_{(j)} < QE^*_{(k)}$$

Thus, optimizing process of QE repeats two following steps: firstly, assign the weight vector of each neuron to the cluster's center that it represents; secondly, compete "*different elements*" between neurons to redetermine the data subset that each neuron represents.

The "*different elements*" competitive strategy is performed for all neurons aiming to optimize clusters's elements, which reduce QE and may also restrict the self-intersections [18] resulting in reducing TE. Figure.3 illustrates the self-intersections given by Lopez-Rubio. In particular, the weight vector of neuron i matches with other units rather than adjacent neighbors j, k, p, q, that is the cause of self-intersections.

The "*different elements*" competitive process will adjust the weights of i and its adjacent neighbors closer if between i and its adjacent neighbors exist "*different elements*". Hence, if there exist "*different elements*" of j, k, p, q to i, the cluster I_i will extend towards I_j, I_k, I_p, I_q and if there exist the "*different elements*" of i to j, k, p, q, the clusters I_j, I_k, I_p, I_q will extend towards I_i, vice verse. As a result, the corresponding cluster centers c_i, c_j, c_k, c_p, c_q approach closer to each other. In principle, the clusters which have large differences tending to shrink (to reduce the difference), and the clusters have small differences tending to expand and move towards the clusters which have big differences.

Thus, the competition "*different elements*" directly reduces QE but only contributes to reduce TE. In which, QE always decreases when competition occurs, while reduction of TE depends on the number of "*different elements*" between the neuron which causes self-intersection and its adjacent neighbors. If the neuron which causes self-intersection is too different from its adjacent neighbors, there is non-existent of "*different elements*" between them, hence, the competition "*different elements*" can not remove the self-intersection status in this case.

We propose the IMQS algorithm which used to improve the feature map's quality, including the following steps:

- *Step 1:* Determine the subset I_i of $I = \{I_1, I_2, .., I_s\}$, with $i = 1..s$

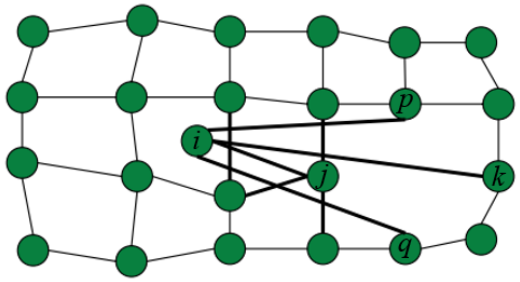

Figure 3: Illustrations of seft-intersections.

- *Step 2:* Repeat competition "*different elements*" between all neurons until satisfying one of the following stopping conditions: there is no existence of any "*different elements*" between all neurons or $||QE - QE^*|| < \theta$, with θ is option threshold value.

The competition "*different elements*" procedure between neuron i and others is as follows:
With $\forall x \in I_i, i \neq j, i \neq k, j \neq k$,
if $d(x, \mathrm{w}_i) > d(x, \mathrm{w}_j)$ and $d(x, \mathrm{w}_i) > d(x, \mathrm{w}_k)$ and $d(x, \mathrm{w}_j) < d(x, \mathrm{w}_k)$ then

$$I_i = I_i \backslash x; I_j = I_j \cup x;$$

$$c_i = \frac{1}{|I_i|} \sum_{v=1}^{|I_i|} x_v; c_j = \frac{1}{|I_j|} \sum_{v=1}^{|I_j|} x_v;$$

$$\mathrm{w}_i = c_i; \mathrm{w}_j = c_j$$

For ease of installation, the Batch-IMQS version is recommended, allowing batch processing. Repeat two following steps until the stop condition is satisfied:

- *Step 1:* Determine the subset I_i of $I = \{I_1, I_2, .., I_s\}$, with $\forall i = 1..s$.
- *Step 2:* Calculate the codebook vectors c_i, and assign $w_i = c_i$, with $\forall i = 1..s$.

The Batch-IMQS is suitable for the case that the dataset's size and Kohonen layer's size are small. The next section presents experiments of the method and evaluates the effectiveness of the algorithm through measures, QE and TE.

4. EXPERIMENTS

The experiments are performed with two assuming datasets (XOR, Odd-Even) and eleven real datasets [1]. In each dataset, the experiment includes two phases: the first phase is training SOM neural network; the second phase is improving feature map's quality obtained of SOM by IMQS. In both phases, the QE, TE values are tracked for comparison. Kohonen layer's size is 15×15 which is used for all datasets. The IMQS algorithm has stop condition parameter $\theta = 10^{-4}$, it is used to improve the map's quality obtained of SOM in two cases:

In the first case, the SOM algorithm's stop condition is when the number of training times is three times the size of the dataset (because the number of samples of some datasets is small). Table 1 is the result of QE, TE before and after applying IMQS.

No.	Dataset	Vectors(N)	Features	SOM		IMQS	
				QE	TE	QE	TE
1	XOR	4125	2	0.0483	0.9707	0.0311	0.9350
2	Odd-Even	400	8	130.3291	0.85	67.6296	0.39
3	Iris	150	4	0.5593	0.8	0.2098	0.4467
4	Glass	214	9	4.0375	0.3879	1.6695	0.3411
5	Flame	240	2	0.9542	0.5708	0.3723	0.4125
6	Pathbased	300	2	3.2396	0.5833	0.9751	0.5067
7	Spiral	312	2	3.0021	0.5769	0.8925	0.4071
8	Jain	373	2	1.6923	0.4987	0.7313	0.4745
9	Compound	399	2	1.6329	0.6090	0.7192	0.6040
10	Aggregation	788	2	2.3739	0.5584	1.0257	0.4251
11	R15	600	2	0.6842	0.8917	0.2532	0.7283
12	D31	3100	2	4.8514	0.6468	1.0631	0.4958
13	Wine	178	13	63.4801	0.3146	13.0856	0.2753

Table 1: Results of QE, TE with training times of SOM by $3 \times N$.

No.	Dataset	Vectors(N)	Param k	SOM		IMQS	
				QE	TE	QE	TE
1	XOR	4125	10	0.0385	0.9685	0.0294	0.9372
2	Odd-Even	400	60	98.5358	0.8975	56.5497	0.4925
3	Iris	150	30	0.3553	0.8733	0.1177	0.6733
4	Glass	214	60	2.9664	**0.4439**	1.4712	**0.4626**
5	Flame	240	30	0.4684	0.9292	0.2184	0.8625
6	Pathbased	300	30	0.9543	0.8367	0.4825	0.74
7	Spiral	312	30	1.1807	0.8462	0.5970	0.7981
8	Jain	373	30	0.7374	0.8794	0.4187	0.8231
9	Compound	399	30	0.8649	0.9599	0.5373	0.8747
10	Aggregation	788	30	1.1502	0.8363	0.7541	0.75
11	R15	600	30	0.2724	0.8817	0.1623	0.76
12	D31	3100	10	1.7487	0.7916	0.7336	0.7103
13	Wine	178	60	19.5587	0.4270	6.2342	0.3764

Table 2: Results of QE, TE with training times of SOM by $k \times N$.

Table 1 shows that all the QE and TE values of algorithm IMQS decrease compared with the corresponding values of SOM. The average reduction ratio of QE is 61.2% and TE is 20.6%.

In the second case, the number of training times of the SOM algorithm is increased to test the ability of its convergence compared with IMQS. Although the number of training times of datasets increases largely (k times the size of the dataset), the QE and TE values of SOM in Table 2 are still greater than the QE, TE values of IMQS in Table 1. This means that, instead of increasing the number of SOM's training times too many, the IMQS algorithm should be used to optimize the feature map to achieve more optimal QE and TE values, in less time. In fact, the larger number of SOM training times is, the less fluctuation of QE and TE will be, however, only QE tends to decrease

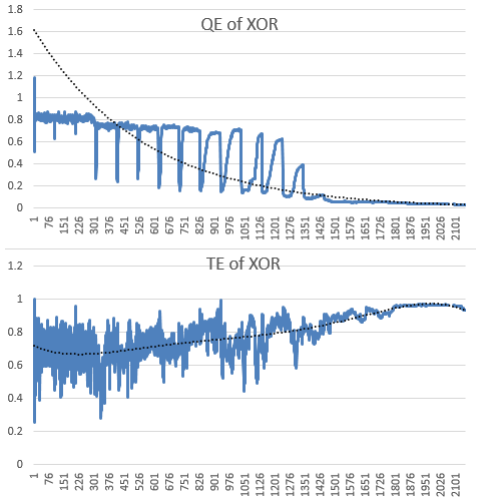

Figure 4: Chart of QE and TE of XOR.

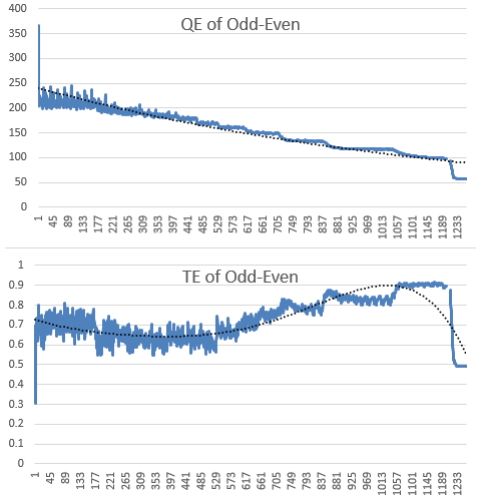

Figure 5: Chart of QE and TE of Odd-Even.

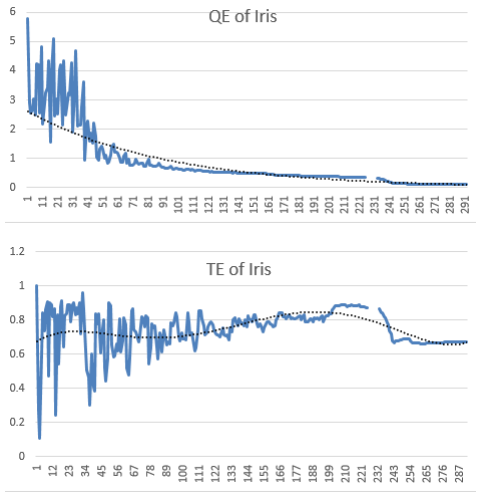

Figure 6: Chart of QE and TE of Iris.

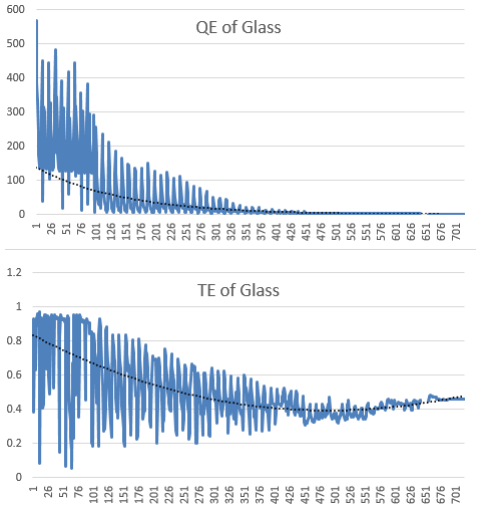

Figure 7: Chart of QE and TE of Glass.

gradually, TE tends to increase gradually. Thus, further increasing the number of training SOM is not as effective as using IMQS.

In the Table 2, the reduction ratio of QE, TE (of IMQS compared with SOM) is less than that in Table 1. In which, the average reduction ratio of QE is 47.5%, TE is 11.8%. It is caused by when the training times of SOM increases, the data representation accuracy increases (in other words, the difference reduces or the number of "*different elements*" reduces), but the nature of IMQS algorithm is trying to decrease the difference to reduce QE and TE, so IMQS will be less effective if the number of "*different elements*" is small. In the result of Glass dataset, the IMQS algorithm couldn't reduce TE while QE still decreased. This shows that the self-intersections are not removed (the cause is explained above). The reason that TE of IMQS rises slightly compared to TE of SOM is the correlation between QE and TE [24].

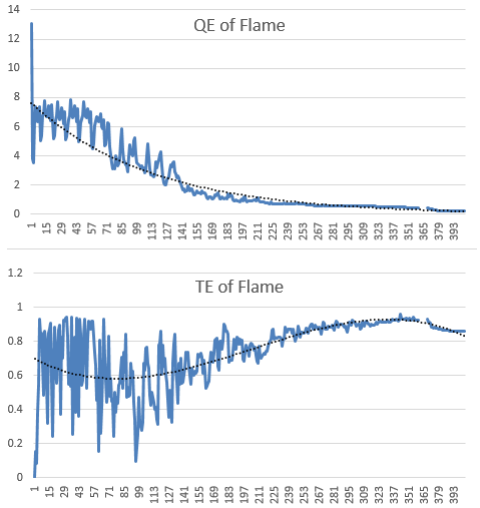

Figure 8: Chart of QE and TE of Flame.

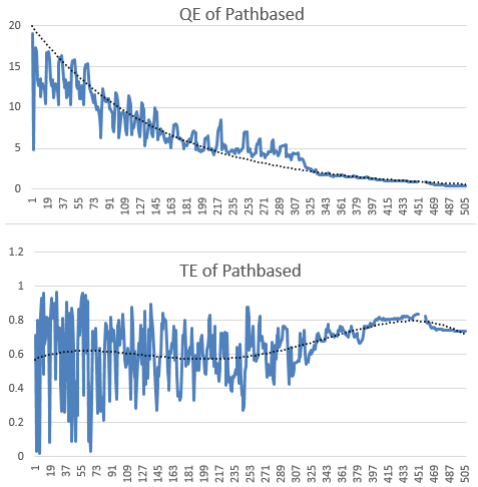

Figure 9: Chart of QE and TE of Pathbased.

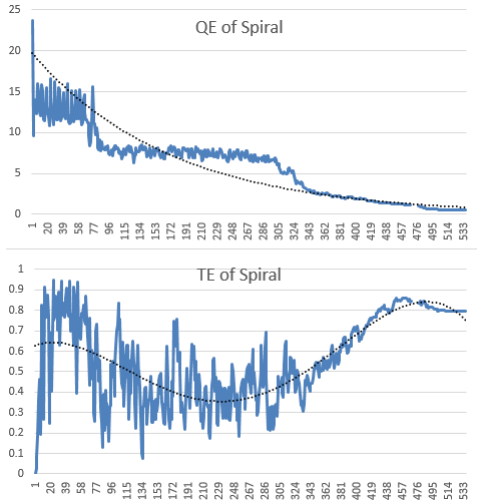

Figure 10: Chart of QE and TE of Spiral.

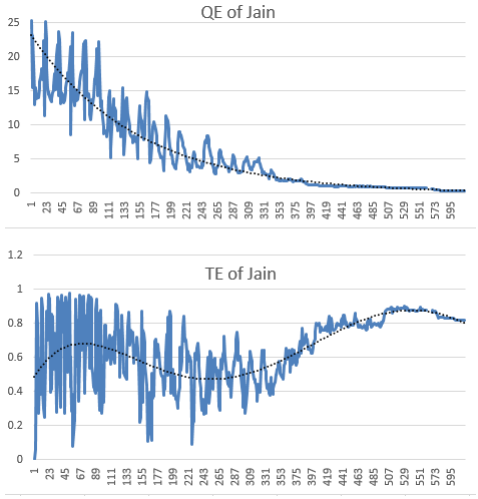

Figure 11: Chart of QE and TE of Jain.

The figures from Figure.4 to Figure.16 are the charts which present the fluctuation process of QE and TE of datasets in the second test case (details of all experimental results are presented in appendix). In particular, the vertical axis represents the value of the measures; a horizontal axis represents the times of training; the gap in the line represents the ending of SOM and the beginning of IMQS algorithm; dashed line shows trends.

Observing the charts of SOM (from the starting point to the gap) it is found that: Initially, the values of QE and TE fluctuates dramatically, but when the number of training times increases, the fluctuation decreases. It is because the neighboring radius gradually decreases in the training process. There is a tradeoff between QE and TE, where QE decreases gradually to 0, whereas, TE increases gradually to 1.

The charts of IMQS (from the gap to the end) is much shorter than the chart of SOM, because the

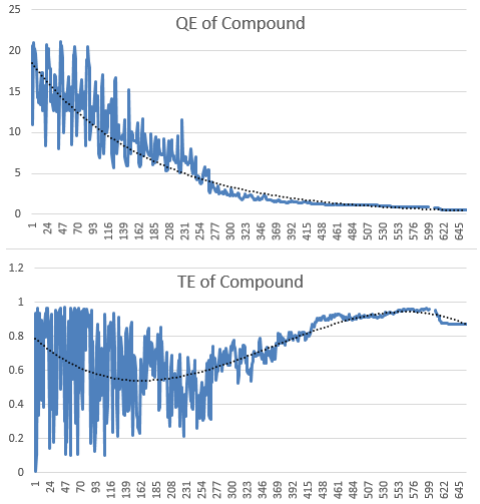

Figure 12: Chart of QE and TE of Compound.

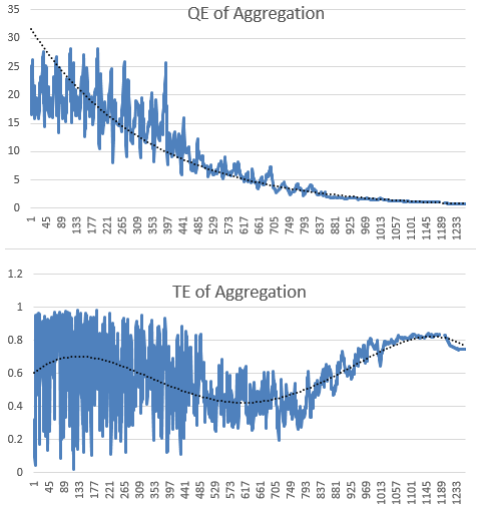

Figure 13: Chart of QE and TE of Aggregation.

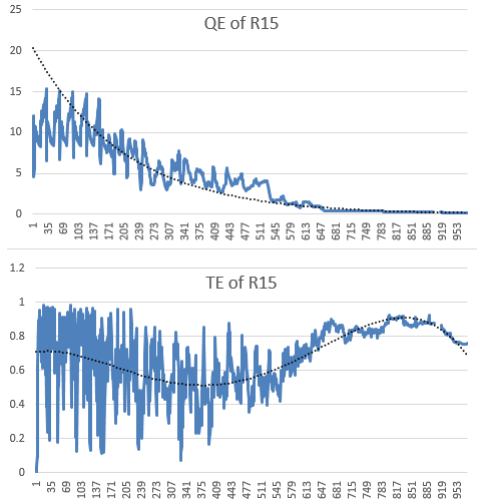

Figure 14: Chart of QE and TE of R15.

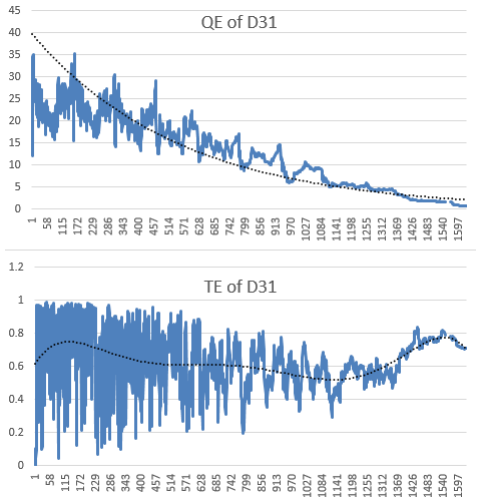

Figure 15: Chart of QE and TE of D31.

IMQS algorithm converges quite quickly. QE steadily decreases with all datasets, but TE has different reduction rate with each dataset because it depends on the ability to remove the self-intersections status. In most of the datasets, IMQS's TE has a downward trend. It is rare that TE does not reduce.

In both cases, the reduction ratio of TE is always less than the reduction ratio of QE because QE always decreases when competing "*different elements*" between all neurons occurring, but TE may only reduces when competing "*different elements*" between the neurons which cause self-intersection and its adjacent neighbors occurs.

Beside experiments on 13 datasets above, some experiments on other datasets such as image data, sound and the results are similar are also conducted.

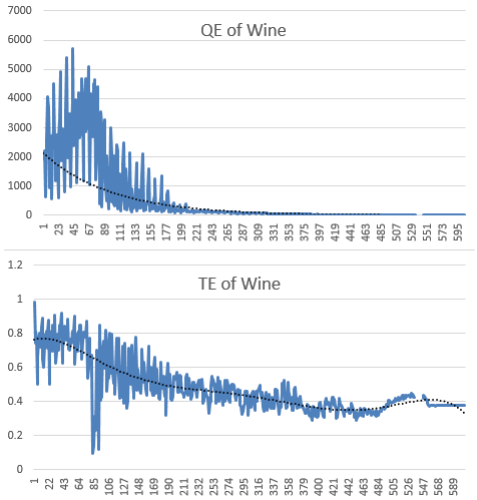

Figure 16: Chart of QE and TE of Wine.

5. CONCLUSIONS

Theoretically, the original SOM and some improved SOMs can achieve good map quality if the suitable configuration parameters of the network are determined. However, in reality, it is time consuming and very difficult to perform because we must be "trying error" several times to determine the parameter's value of each dataset. Thus, the map obtained by the unsuitable parameters is a common outcome in practice. Therefore, our solution does not aim to adjust the configuration parameters as well as learning algorithm, but to optimize obtained map after learning algorithm ends. The final obtained map has better quality because QE plummets and TE decreases in most cases, except that the number of "*different elements*" between the neuron which cause self-intersection and its adjacent neighbors is small or non-existed. Although our method can not reduce TE in all cases, but in reality, this is an effective method to improve the feature map quality. Compared with the method by Kamimura [14], our method has a larger reduction rate of QE and TE and can be applied to all datasets without valuating private parameters for each dataset. In addition, the proposed method completely appropriates the original SOM or any other variants of SOM, because we do not impact on learning algorithm but only optimize the obtained maps of those models.

The reduction ratio of TE may greater if IMQS algorithm prioritizes to competition between adjacent neighbors first, then to the remaining neurons (Batch-IMQS is unable to prioritize competing since the data is processed in batch), because self-intersection status will be eliminated sooner when the weights of the adjacent units move closer together faster. In fact, the reduction of TE is only effective if we exclude self-intersection status as soon as it appears in the training process.

APPENDIX

The following link provides detail of the experiments, including the value of QE, TE tracked in real time and the charts which show the process of the fluctuation of QE and TE in both experimental cases.

`http://www.mediafire.com/view/hrdpyoc3znot5y7/experimental_results.xlsx`

ACKNOWLEDGMENT

The author would like to thank the scientists of the Information Technology Institute - Vietnam Academy of Science and Technology for helping him during the time he had been studying there.

REFERENCES

[1] [Online]. Available: http://cs.joensuu.fi/sipu/datasets/

[2] H. Bauer, M. Herrmann, and T. Villmann, "Neural maps and topographic vector quantization," *Neural Netw*, vol. 12, no. 4-5, pp. 659–676, 1999.

[3] H. Bauer and K. Pawelzik, "Quantifying the neighborhood preservation of self organizing feature maps," *IEEE Transactions on Neural Networks*, vol. 3, no. 4, pp. 570–579, Jul 1992.

[4] E. Berglund and J. Sitte, "The parameterless self-organizing map algorithm," *Neural Netw IEEE Trans*, vol. 17, no. 2, pp. 305–316, 2006.

[5] A. César Astudillo and B. John Oommen, "Topology-oriented self-organizing maps: A survey," *Pattern Analysis and Applications*, vol. 17, no. 2, pp. 223–248, May 2014.

[6] M. Chattopadhyay, P. K. Dan, and S. Mazumdar, "Application of visual clustering properties of self organizing map in machine-part cell formation," *Applied Soft Computing*, vol. 12, no. 2, pp. 600–610, 2012.

[7] J. A. Flanagan, "Self-organization in kohonen's som," *Neural Networks*, vol. 9, no. 7, pp. 1185–1197, 1996.

[8] E. Germen, "Increasing the topological quality of kohonen's self organizing map by using a hit term," in *Neural Information Processing, Proceedings of the 9th International Conference on (ICONIP '02)*, vol. 2. Singapore: IEEE, Nov 18-22 2002, pp. 930–934.

[9] ——, "Improving the resultant quality of kohonens self organizing map using stiffness factor," in *Advances in Natural Computation, Lecture Notes in Computer Science (First International Conference, ICNC 2005)*, vol. 3610. Changsha, China: Springer Berlin Heidelberg, August 27-29 2005, pp. 353–357.

[10] E. Germen and S. Bilgen, "A statistical approach to determine the neighborhood function and learning rate in self-organizing maps," in *Proc. ICONIP97*. Springer, 1997, pp. 334–337.

[11] S. Grossberg, "Competitive learning: From interactive activation to adaptive resonance," *Elsevier*, vol. 11, no. 1, pp. 23–63, JanuaryMarch 1987.

[12] T. Heskes, "Energy functions for self-organizing maps," in *Kohonen Maps*, S. K. E. Oja, Ed. Amsterdam: Elsevier Science, 1999, pp. 303–316.

[13] C. Kahraman, *Computational Intelligence Systems in Industrial Engineering*, 1st ed., C. E. Kahraman, Ed. Atlantis Press, 2012, vol. 6.

[14] R. Kamimura, "Input information maximization for improving self-organizing maps," *Applied Intelligence*, vol. 41, no. 2, pp. 421–438, Mar 20 2014.

[15] K. Kiviluoto, "Topology preservation in self-organizing maps," in *Neural Networks, IEEE International Conference on (ICNN96)*, vol. 1. Washington, DC: IEEE, Jun 3-6 1996, pp. 294–299.

[16] T. Kohonen, *Self-Organizing Maps*, 3rd ed. Springer-Verlag, 2001.

[17] J. Lampinen and E. Oja, "Clustering properties of hierarchical self-organizing maps," *Journal of Mathematical Imaging and Vision*, vol. 2, no. 2-3, pp. 261–272, November 1992.

[18] E. Lopez-Rubio, "Improving the quality of self-organizing maps by self-intersection avoidance," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 24, no. 8, pp. 1253–1265, Aug 2013.

[19] F. Mulier and V. Cherkassky, "Statistical analyses of self-organization," *Neural Networks*, vol. 8, no. 5, pp. 717–727, 1995.

[20] J. I. Mwasiagi, H. XiuBao, W. XinHou, and C. Qing-dong, "The use of k-means and kohonen self organizing maps to classify cotton bales," in *Beltwide Cotton Conferences (BWCC'07)*, New Orleans, Louisiana, January 9-12 2007.

[21] A. Neme, E. Chavez, A. Cervera, and V. Mireles, "Decreasing neighborhood revisited in self-organizing map," in *Artificial Neural Networks - ICANN 2008*, vol. 5163. Prague, Czech Republic: Springer Berlin Heidelberg, September 3-6 2008, pp. 671–679.

[22] A. Neme and P. Miramontes, "Self-organizing map formation with a selectively refractory neighborhood," *Neural Processing Letters*, vol. 39, no. 1, pp. 1–24, February 2014.

[23] D. Polani, "Measures for the organization of self-organizing maps," *Studies in Fuzziness and Soft Computing*, vol. 78, pp. 13–44, 2002.

[24] G. Pölzlbauer, "Survey and comparison of quality measures for self-organizing maps," in *Proceedings of the Fifth Workshop on Data Analysis (WDA - 04)*, J. Paralič, G. Pölzlbauer, and A. Rauber, Eds. Sliezsky dom, Vysoké Tatry, Slovakia: Elfa Academic Press, June 24-27 2004, pp. 67–82.

[25] D. E. Rumelhart and D. Zipser, *Feature Discovery By Competitive Learning.* MA, USA: MIT Press Cambridge, 1986, vol. 1, ch. 5, pp. 151–193.

[26] Y. Sun, "On quantization error of self-organizing map network," *Neurocomputing*, vol. 34, no. 1-4, pp. 169–193, September 2000.

[27] A. E. Uriarte and D. F. Martín, "Topology preservation in som," *Int J Appl Math Comput Sci*, vol. 1, no. 1, pp. 19–22, 2005.

[28] T. Villmann, R. Der, and T. Martinez, "A new quantitative measure of topology preservation in kohonen's feature maps," in *Proceedings of the IEEE International Conference on Neural Networks 94*, vol. 2. Orlando, Florida, USA: IEE, 1994, pp. 645–648.

[29] S. Wang and H. Wang, "Knowledge discovery through self-organizing maps: Data visualization and query processing," *Knowledge and Information Systems*, vol. 4, no. 1, pp. 31–45, 2002.

Received on June 17 - 2015
Revised on September 14 - 2015