Neurocomputing 169 (2015) 89–99

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A new incremental neural network for simultaneous clustering and classification

Amel Hebboul [a,*], Fella Hachouf [b], Amel Boulemnadjel [b]

[a] Département de Sciences Exactes et Informatique, Ecole Normale Supérieure de Constantine, Ali Mendjli, Constantine, Algérie

[b] Laboratoire d'Automatique et de Robotique, Département d'Electronique Faculté des sciences de la technologie, Université Constantine 1 Route d'Ain el bey, 25000 Constantine, Algérie

ARTICLE INFO

Article history:
Received 1 May 2014
Received in revised form
8 February 2015
Accepted 10 February 2015
Available online 29 May 2015

Keywords:
Neural network
Incremental
Unsupervised learning
Semi-supervised learning
Clustering
Support vector machine SVM

ABSTRACT

In this paper, an Incremental Neural Network for Classification and Clustering (INNCC) is proposed. The main advantages of this neural network are the linkage between data topology preservation and classes representation by using the cluster posterior probabilities of classes. It is a constructive model without prior conditions such as a suitable number of nodes. A new neuron is inserted when new data are not represented by existing neurons. In training step, both supervised and unsupervised learning are used. The training dataset contains few samples with class labels and several unlabeled ones. The Support Vector Machines (SVM) operates in the training step to assess the INNCC classification result. The proposed approach has been tested on synthetic and real datasets. Obtained results are very promising.

1. Introduction

Traditionally, there have been two fundamentally different types of tasks in machine learning. The first one is unsupervised learning [1]. The samples without class labels are grouped into meaningful clusters. These clusters can be utilized to describe the underlying structure in data space, which is helpful for better understanding of data. The second task is supervised learning [2] (classification). The samples with class labels are used to build the classification mechanism, through which class labels can be provided for new samples. However, labeled data are usually insufficient and hard to obtain since data labeling requires extensive expert effort and is often time-consuming. Meanwhile, unlabeled data are often abundant in real world. Consequently, Semi-Supervised Learning (SSL) is halfway between supervised and unsupervised learning. In addition to unlabeled data, the algorithm is provided with some supervision information but not necessarily for all samples [3]. The key idea of Semi-Supervised Clustering [3–5] is to take advantage of different kinds of prior information to improve the performance of clustering. Another neighbor concept in semi-supervised learning is Semi-Supervised Classification methodology [6], as the name implies, a classifier is first trained by labeled data and used to classify unlabeled data. Consequently, unlabeled data that are classified with the highest confidence (probability of belonging to a certain class) are added incrementally to the labeled dataset with their predicted labels. The procedure is repeated until convergence. These methods will fail to approach the real data space if the labeled data cannot represent the underlying structure of the particular space. Because the initial trained classifier will give bad results on the unlabeled data. Recently, many researchers have given attention to not only use labeled and unlabeled data in the data set training, but use semi-supervised clustering approach in classifier training [7,8]. Generally, such extra information can be given in many forms. Three most common types are labeled data [5,7,8], relative associations [9] and constrained relations [10–12].

Recently, another concept in machine learning have attracted more attention. It is the incremental learning. It is the ability of learning new information without relearning and deleting the old data, so it raises the so-called stability/plasticity dilemma [13]. Its main advantage is the system ability to make decisions on-line.

Artificial Neural Network (ANNs) are used frequently in machine learning in different areas. Their simple implementation and the existence of mostly local dependencies exhibited in the structure allow fast, and parallel hardware implementations. In supervised learning, a Multi-Layer Perception (MLP) [14] and a

* Corresponding author.
E-mail addresses: ahebboul@yahoo.fr (A. Hebboul), hachouf.fella@gmail.com (F. Hachouf), amel.boulemnadjel@yahoo.fr (A. Boulemnadjel).

http://dx.doi.org/10.1016/j.neucom.2015.02.084
0925-2312/

Probabilistic Neural Network (PNN) [15] successfully use the class information of samples to achieve high classification accuracies. In suite of, they emphasize more the classification of the data than the revelation of the data distribution, they fail to interpret the obtained classification results well. The main limitation of the original PNN architecture proposed by Specht [15] is that it requires a separate neuron for each training pattern, which makes the computation very slow for large databases requiring a large amount of space in memory. The most popular unsupervised network is Self-Organizing Map (SOM) models [16]. A combination of Competitive Hebbian Learning (CHL) [17] and Neural Gas (NG) [18] is effective in constructing topological structure: Growing Neural Gas (GNG) [19] and Dynamic Cell Structures [20]. However these models suffer from two main problems; the limitation of the number of nodes to avoid exceeding the number of nodes and the inability of the GNG to eliminate noise. Adaptive Resonance Theory (ART) networks have been proposed as a solution to the stability/plasticity dilemma [13]. These networks learn top down expectations which are matched with bottom up input. The expectations which are called categories summarize sets of the input data into clusters. Several neurons networks based on the GNG [19] are proposed for the incremental learning, they use the similarity threshold criterion of insertion of new node: Prudent and Abdel Ennaji in [21] proposed an Incremental Growing Neural Gas (IGNG). But the similarity threshold should be a priori determined and it is sensitive of noise. The Self-Organizing Incremental Neural Network (SOINN) was introduced by Furao and Hasegawa [22]. SOINN has a two-layered structure representing the input distribution at different levels of detail; this structure reduces the sensitivity to noise. It uses an adaptive threshold. But the weights of the neurons do not stabilize completely in the incremental learning. The Enhanced Self-Organizing Incremental Neural Network (ESOINN) [23] and Incremental Growing with Neural Gas Utility parameter (IGNGU) [24] based on SOINN have been proposed to solve the above-mentioned problem: in ESOINN remove the second layer and one condition for the insertion of new neurons. Furthermore, the whole network can be trained online. But similar to SOINN, the weights do not stabilize completely. In IGNGU, Neurons type and age are added so that the networks become more stable in the incremental training. To deactivate neurons representing the old data, the neurons age is used to not remove the young neurons. M. Tscherepanow et al. proposed in [25] a novel unsupervised neural network combining elements from Adaptive Resonance Theory and topology-learning neural networks (TopoART). It enables a stable on-line clustering of stationary and non-stationary input data by learning their inherent topology. Here, two network components representing two different levels of detail are simultaneously trained.

In this paper, the focus is on the combination between the clustering and the classification learning in Incremental Semi-Supervised Neural Network for the following reasons: (1) In reality unlabeled data are more avoidable and low cost than labeled data. (2) The possibility of updating the system with new data without re-learning the old ones, thus reducing the computation time and the memory space for the old data. (3) ANNs can be naturally used in on-line learning and for large data set applications. Their simple implementation and the existence of mostly local dependencies exhibited in the structure allows for fast, parallel implementations in hardware. IGNG [21] realizes semi-supervised clustering, alternating between clustering part of huge datasets and having users to correct the network. Thus human operation remains complex, including finding clustering mistakes, correcting the network by edge insertion and deletion and assigning labels to disjoint subgraphs. Furao Shen et al. in [26] enhanced the SOINN to the semi-supervised learning. The new neural network has three layers: input, competitive and output. In the input layer, labeled and unlabeled data are mixed to form training data input to the competitive layer, such nodes and their connection represent the topology structure for inputting training data. Labeled data is used to label competitive layer nodes. A node labeled directly with an input vector label is called a "teacher node". And it displays results in the output layer, which has the same structured as the competitive layer but it has labeled nodes. The weight vectors of labeled nodes are used as prototypes to build classifiers. But, this model remains not stable in the incremental learning case.

In this work, we develop a new Incremental Neural Network for simultaneous Clustering and Classification (INNCC) improved by Support Vector Machines (SVM) [27] able to partition and classify the input data. This approach uses the unlabeled and labeled samples to extract the data topology and the labeled ones to create a classifier. We use the Bayesian theory and the cluster posterior probabilities of classes to join between the clustering and classification learning. In the semi-supervised learning, unlabeled data classified with the highest confidence (probability of belonging to a certain class) are incrementally added to the labeled dataset with their predicted labels. In our method before this data and their predicated labels are added in the labeled dataset; the Support Vector Machines (SVM) is used to assess their labels and only the data with proved labels are added into the labeled dataset. To make the approach more appropriate for the non-spherical distribution, a kernel-based metric is adopted. The goal is to develop a network that is able to (1) automatically learn the number of prototypes needed to represent every class. (2) Learn and classify new information without destroying old learned ones, i.e., realize incremental learning. (3) It is not affected by noise. Section 2 describes the proposed method. Experimental results on synthetic and real datasets are given in Section 3. Finally, conclusion and perspectives are given in Section 4.

2. Proposed approach

In this section, we will give a framework for the proposed incremental semi-supervised neural network for clustering and classification. First, an incremental semi-supervised neural network using both labeled and unlabeled data is employed to learn the underlying data space structure and a classifier is trained using labeled data. The incremental semi-supervised neural network produces the posterior probabilities of each unlabeled sample to different classes. The unlabeled sample that has higher certainty of belonging to one class (i.e., has one high value of posterior probabilities of classes) is then classified by the classifier. The most confidently classified unlabeled data with their predicted labels are added to the labeled set. The incremental semi-supervised neural network algorithm and the classifier are retrained. This procedure is repeated until all unlabeled data are labeled. In this way our framework combines clustering and classification in unison. The classifier is used to evaluate the computed labels of data by the network during the training to minimize the errors and to increase the performances of classification. We choose the SVM as classifier because the SVMs [27] are a powerful machine learning technique based on the principle of structural risk minimization. They can solve linearly non-separable problems using kernel tricks and have shown an excellent generalization performance. Detailed surveys of SVM can be found in [28].

Before presenting the INNCC training in details, we give some notations: the training data set BA is

$$BA = \{(x_i, y_i, PX_i)/y_i \in \{0, 1, \dots, K\} \bigwedge x_i \in R^d, \quad i = 1, \dots, N\}$$

$$BA = BA^+ \cup BA^-$$

where x_i is the feature vector of sample i and y_i is its label. $PX_i = p(k|x_i)$ is the posterior probabilities of the kth class and the sample x_i. PX is the posterior probabilities matrix. K is the number of class, d is the input space dimension and N is the number of samples. BA^+ is the set of labeled samples (i.e $y_i \neq 0$), and BA^- is the set of unlabeled ones (i.e $y_i = 0$).

$$PX = \begin{bmatrix} p(1|1) & \cdots & p(k|1) & p(K|1) \\ \vdots & \vdots & p(k|i) & \vdots \\ p(1|N) & \cdots & \cdots & p(K|N) \end{bmatrix}$$

The incremental neural network is represented by $INNCC = (W, D, AG, NA, T, S, U, m)$ where,

- m is the number of the neurons.
- $W = [w_1, .., w_m]]^T$ is the reference vector matrix, where $w_j = [w_{j1}, .., w_{jd}]^T$, $j = 1, \ldots, m$.
- $D = [D_1, .., D_m]^T$ is the vector of the neurons density, where $D_j \in R$, $j = 1..m$.
- $AG = [AG_1, .., AG_m]^T$ is the vector of the neurons age, where $AG_j \in R$, $j = 1..m$.
- $NA = [NA_1, .., NA_m]^T$ is the vector of the activation number of neurons, where $NA_j \in R$, $j = 1..m$.
- $T = [T_1, .., T_m]^T$ is the vector of the neurons similarity threshold, where $T_j \in R$, $j = 1..m$.
- $S = [S_1, .., S_m]^T$ is the vector of the neurons state, where $S_j \in \{0, 1\}$ $j = 1..m$, a activated neuron has $S_j = 1$.
- $U = [U_1, .., U_m]^T$ is the matrix of the posterior probabilities, where $U_j = [u_{j1}, .., u_{jK}]^T$, $j = 1..m$ and $u_{jk} = p(k|j)$, $p(k|j)$ is the posterior probabilities of the kth class and the neuron j.

And $\wedge$ is the logical and, $\vee$ is the logical or $|.|$ is the cardinal of set.

INNCC is an incremental self-organizing maps model. It does not impose any constraint on the structure of the neural network. It is updated in a continuous manner through a competitive learning. It yields the topological structure of the data with typical clusters by eliminating the remaining noise without any a priori information. The built rules of the neural network are: (1) A new neuron is inserted if it is necessary and (2) neurons in the low-density regions are deleted. (3) To ensure the INNCC stability in the incremental training; to learn the new data without forgetting the already learned data, it is necessary to control the elimination of the neurons i.e. to avoid deleting the neurons representing the old data. For these reasons, we associate to each neuron a state which can be activated or inactivated. After the INNCC training, all neurons will be inactivated. In the next new data training, the neurons are activated by these ones. In the useless neurons elimination only the activated neurons intervene.

Each neuron has a reference vector w_n, an age age_n, a state S_n, a similarity threshold T_n, an accumulator of density D_n, an accumulator of activations NA_n and a vector of posterior probability of class U_n. When an input sample (x_i, y_i, PX_i) is presented to INNCC, it finds the nearest neuron (winner) of the sample. Subsequently it judges if the input sample belongs to the same cluster of the winner using the similarity threshold criterion. The INNCC updates adaptively the similarity threshold of every neuron because the input data distribution is unknown. The similarity threshold T_n is calculated using the minimum distance between neuron n and the other neurons in the network.

$$T_n^{new} = \min_{j=1..m \wedge j \neq n} dist(w_j, w_n) \quad (1)$$

If the distance between the input sample and the winner neuron is higher than its similarity threshold, the sample is inserted as a new activated neuron with $w_{new} = x_i$, $NA_{new} = 1$, $D_{new} = 0$ and $AG_{new} = 0$ (see Fig. 1).

$$T_{new} = \min_{j=1..m} dist(w_j, x_i) \quad (2)$$

If the distance is lower than the similarity threshold, The winner neuron is updated (the input sample activates the winner neuron) (see Fig. 2) with

$$w_n^{new} = w_n^{old} + \eta * (x_i - w_n^{old}) \quad (3)$$

$$NA_n^{new} = NA_n^{old} + 1 \quad (4)$$

In order to not reside the neurons in the regions containing a large number of unlabeled data, we adapt the value of η as

$$\eta = \begin{cases} 0.02 & \text{if } y_i \neq 0 \\ 0.01 & \text{else} \end{cases}$$

$$D_n^{new} = D_n^{old} + (1 + dist(w_n, x_i))^{-1} \quad (5)$$

Our method is based on an incremental training. Therefore we suggest to compute the cluster posterior probabilities in a continuous manner. Formula (6) is used if the winner neuron is updated. Formula (7) is used in the insertion of a new neuron ($u_{jk} = p(k|j)$).

$$U_j^{new} = U_j^{old} + (PX_i - U_j) * a_j; \quad (6)$$

$$U_j = PX_i \quad (7)$$

where a_j is an adaptation variable. Its value is computed as

$$a_j = (1 + dist(x_i, j))^{-1} \quad (8)$$

After training of τ samples, to control neurons number and noise effect, the neurons having a high age and a low density are removed: if many input samples are near the neuron, the neuron density is high; if few input samples are near the neuron, the density of this neuron is low. The mean density of activated neurons is computed by formula (9) and the activated neurons satisfying condition (10) are removed.

$$md_j = \frac{D_j}{NA_j} \quad (9)$$

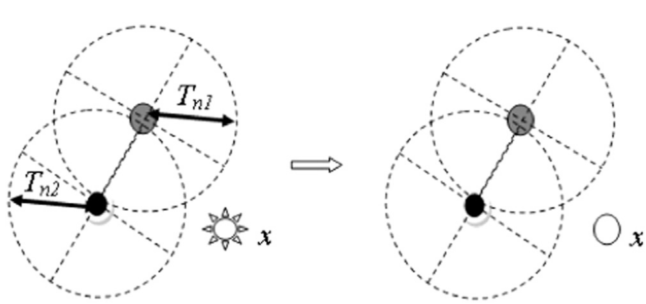


Fig. 1. New neuron insertion: if the distance between the input data x and the winner is larger than the similarity threshold.

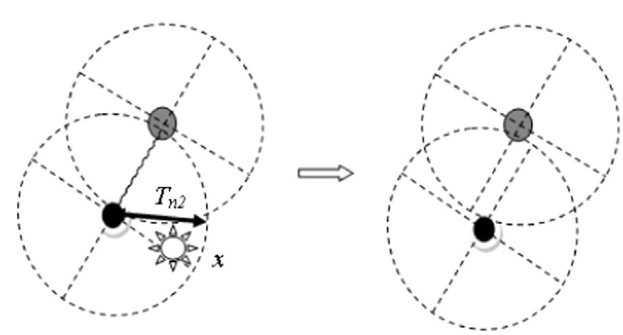


Fig. 2. Winner neuron is updated: if the distance between the input data x and the winner is lower than similarity threshold.

$$(md_j < c* \sum_{j=1..ma} md_j/ma) \wedge age_j > age \tag{10}$$

where ma is the number of the activated neurons and age is a predefine value. c is a scale factor which ranges between zero and one. It is a user defined parameter that controls the network growth (1 means minimum and 0 means maximum growth).

2.1. INNCC classification

Generally neural networks based on the SOM use the One-Nearest Neighbour ($1-NN$) rule or the k-Nearest Neighbour rule $k-NN$ [29] to classify unknown patterns. In this work, we use the posterior probability $p(k|x_i)$ and the output class label for x_i can be determined by (11).

$$y = argmax_{k=1..K} p(k|x_i) \tag{11}$$

In order to incorporate the cluster information into $p(k|x_i)$, it is reformulated through the total probability theorem:

$$p(k|x_i) = \sum_{j=1..m} p(k,j|x_i)$$

$$p(k|x_i) = \sum_{j=1..m} p(j|x_i)*p(k|j,x_i)$$

$$PX_{ki} = p(k|i) = \sum_{j=1..m} p(j|x_i)*p(k|j) \tag{12}$$

where k denotes the kth class, j represents the jth cluster, $p(j|x_i)$ is the posterior probabilities of the presence of corresponding samples in the input space and $p(k|j)$ denotes the cluster posterior probabilities of class membership. Notice that $p(k|j,x_i)$ has no relationship with x_i, and thus it can be simplified as $p(k|j)$. We use the similarity threshold criterion in $p(j|x_i)$ computed as

$$p(j|x_i) = \frac{I_{ji}*(1+dist(x_i,j))^{-1}}{\sum_{-j=1..m}(1+dist(x_i,j))^{-1}} \tag{13}$$

where I_{ji} is a boolean variable, its value given as

$$I_{ji} = \begin{cases} 1 & \text{if } dist(x_i,w_j) < T_j \\ 0 & \text{else} \end{cases}$$

In our approach $p(k|j) = u_{kj}$ is computed during the training of INNCC network. To associate each sample to a class, traditional methods assign each one to a class with great membership degree computed by the formula (11). We have used two selection criteria; confidence criterion $\varepsilon_1 \in [0,1]$ and uncertain criterion $\varepsilon_a \in [0, 0.3]$.

Algorithm 1. Computing labels

Input: PX, ε_1, ε_a.
Output: Y.
For each unlabled sample i:
1. Search y by formula (11)
2. **If** $PX_{yi} < \varepsilon_1$ **Then** $y_i = 0$
 Else
 - Search y' by formula (14) $y' = argmax_{k=1..K \wedge k \neq y} PX_{ki}$ (14)
 - **If** $(PX_{yi} - PX_{y'i}) > \varepsilon_a$
 Then $y_i = y$.
 Else $y_i = 0$.
End for

2.2. INNCC training

Our method is an incremental approach. Thus in the initialization step the INNCC may be already exists or empty (new network for new learning). During the training we denote the confidence thresholds of the INNCC and the SVM by ε_1 and ε_2, respectively. If the selected set T is empty, the value of ε_1 will drop 0.05 at each step. NU is the minimal number of unlabeled data and $TMAX$ is the maximum number of iterations. The proposed approach is resumed by the following algorithm:

Algorithm 2.

Input: BA, INNCC, $TMAX$, c, age, τ, ε_1, ε_a and ε_2.
Output: INNCC.
Initialization:
- If the neural network is not empty, inactivate the existence neurons.
- Initialize PX by: $PX_{ki} = \begin{cases} 1 & \text{if } y_i = k \\ 0 & \text{else} \end{cases} \quad k = 1..K, \quad i = 1..N$
- Add two neurons with reference vector chosen randomly from the input pattern.

Training:
Repeat unit $(t = T_{MAX}) \vee (|BA^-| <= NU)$
- Training INNCC one time with BA.
- Compute the output $p(k|x_i)$ of the INNCC for the unlabeled dataset BA⁻.

Table 1
Instances number of each class.

Class	1	2	3	4	5	6	7
Instances	45	170	102	273	34	130	34

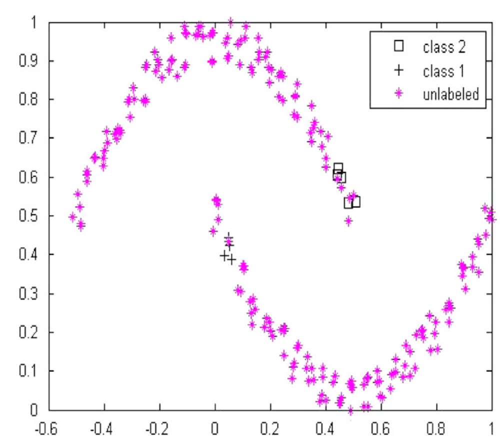


Fig. 3. Two moons training dataset.

Table 2
Algorithm parameters.

CASE	TMAX	ε_1	ε_2	σ	τ	age	c
Off-line	10	0.9	0.5	0.5	20	20	0.25
Noise	10	0.9	0.5	0.5	50	50	0.5

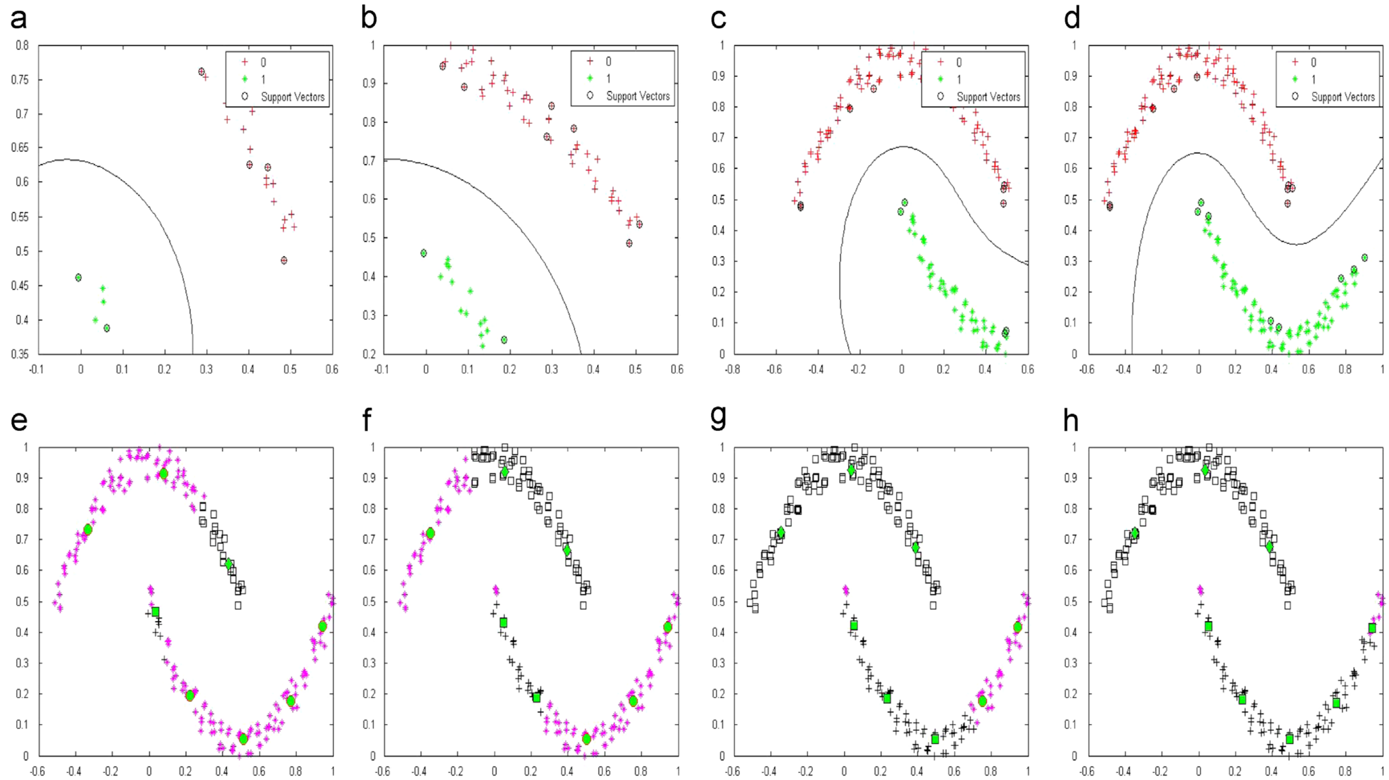


Fig. 4. The SVM and the INNCC result during the proposed method training with the two moons dataset. (a, b, c, d): The SVM result. (e, f, g, h): The INNCC result squat green points represent neurons of class 1, diamond green points represent neurons of class 2 and circle green points represent unlabeled neurons. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.) (a) 8 Iterations, (b) 15 Iterations, (c) 18 Iterations, (d) 25 Iterations, (e) 8 Iterations, (f) 15 Iterations, (g) 18 Iterations, and (h) 25 Iterations.

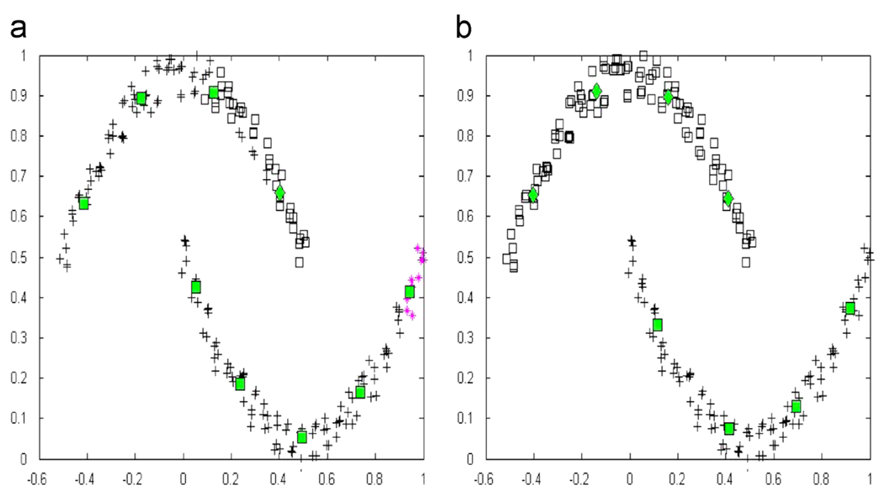


Fig. 5. Two INNCC results of two different attempts without SVM.

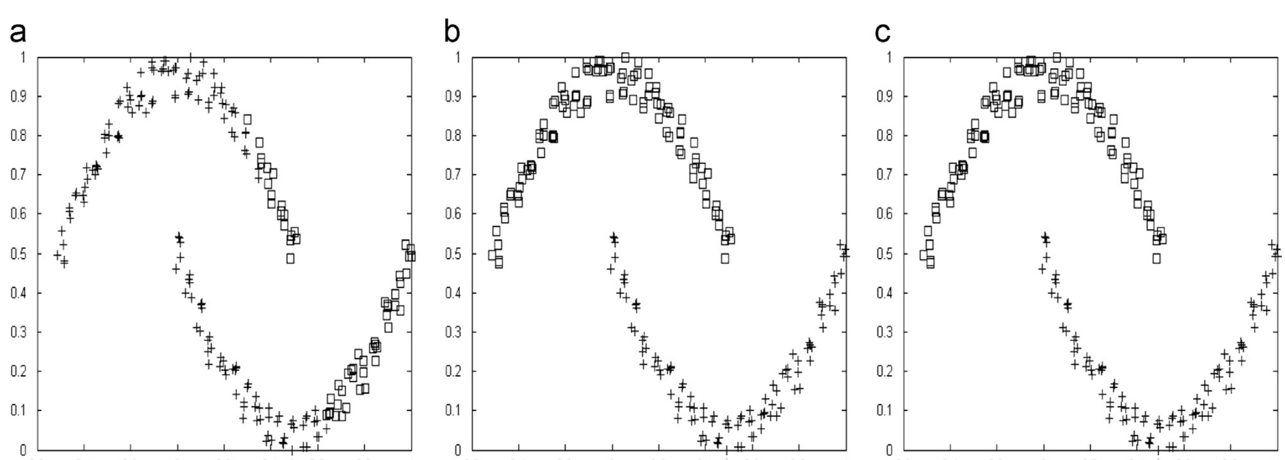


Fig. 6. (a) TSVM result, (b) SVM training with INNCC result.

- Select a dataset T and its labels LT_1 where each sample x_i has high certainty of belonging to one class in using the algorithm *computing labels*.
- Compute the output $f(x_i)$ of the SVM for the selected dataset T.
- Compute the labels LT_2 for the selected dataset T where the output of each sample x_i by the SVM has high values, i.e., $f(x_i) > \varepsilon_2$.
- Select the dataset T_2 where each sample x_i in the dataset T has the same labels, $LT_1(i) = LT_2(i)$.
- Update the current labeled set $BA^+ \longleftarrow BA^+ \cup T_2$.
- Update the current unlabeled set $BA^- \longleftarrow BA^- - T_2$.
- Reduce the value of ε_1 if $T = \oslash$.
- $t{+}{+}$.

2.2.1. Training INNCC one time

Because the training is on-line, we use a local variable *TR*, its value is initialised by *BA* (training dataset). In each iteration a sample is randomly selected from *TR*.

Algorithm 3.

Input: BA, INNCC, c, *age* and τ.

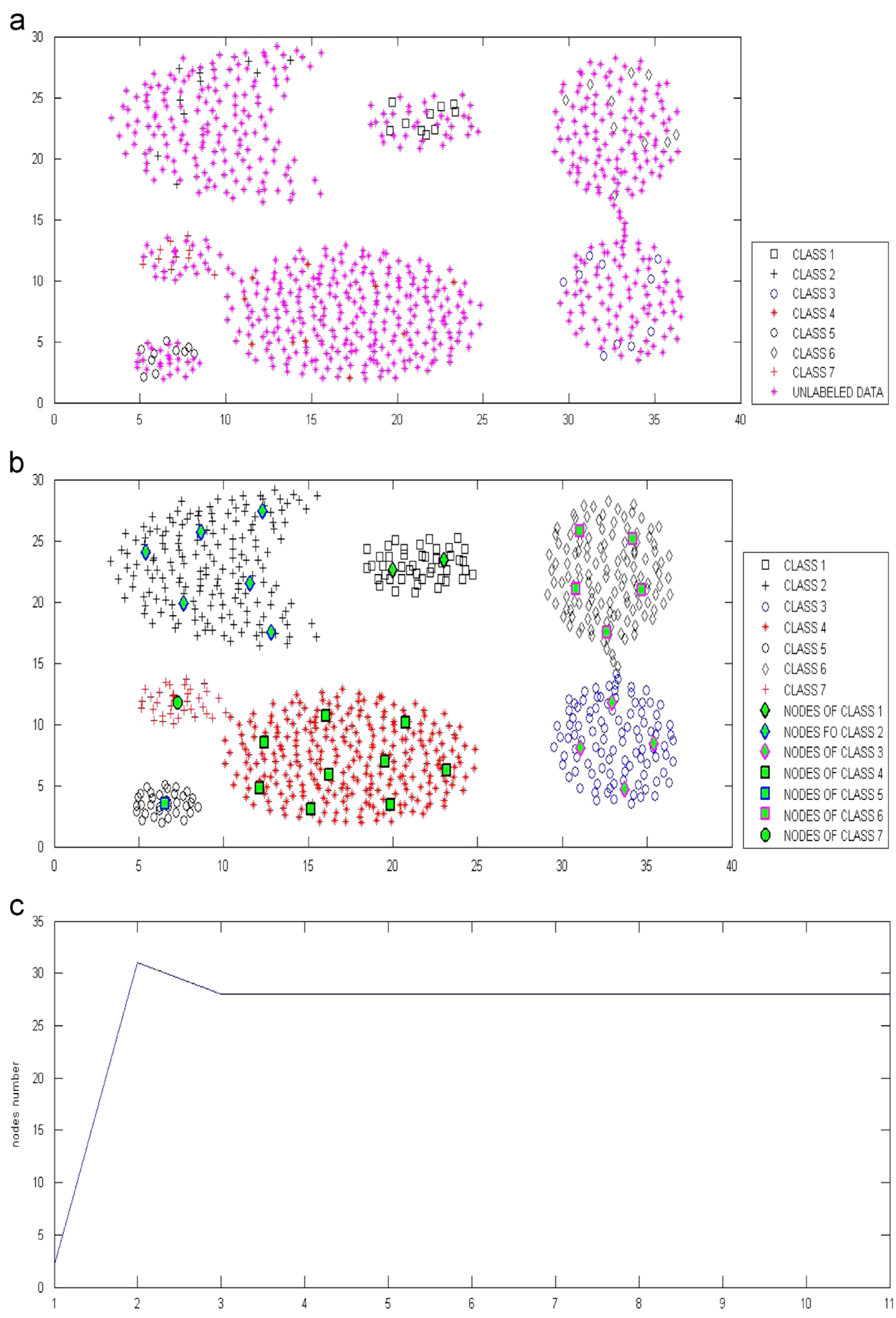


Fig. 7. (a) Artificial dataset Aggregation, (b) INNCC result, (c) The neurons number change during the training.

Output: INNCC.

●$TR \longleftarrow BA$

●Repeat unit $|TR| = 0$

1. Select randomly a sample i from the training dataset TR.

2. $TR \longleftarrow TR - \{(x_i, y_i, PX_i)\}$

3. Search the winner neuron by formula (15):

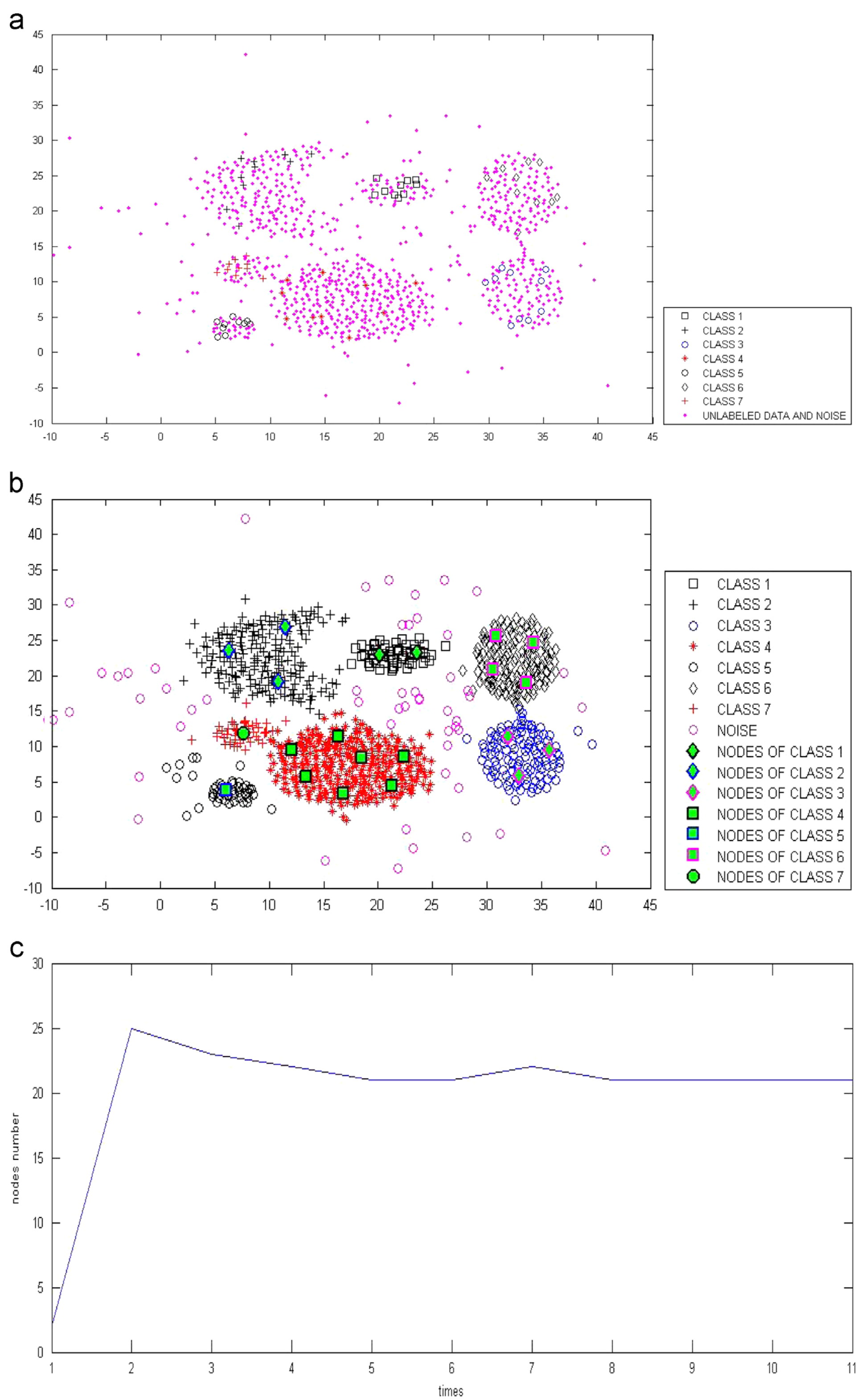


Fig. 8. (a) Noisy training dataset, (b) INNCC result, (c) The neurons number change during the training.

$$n = arg \min_{j=1..m} dist(w_j, x_i) \tag{15}$$

4. **If** $(dist(w_n, x_i) < T_n)$ **then**

Update the winner neuron n

Else

Add a new neuron.

5. After training of τ samples:

- Remove the useless activated neurons.

- Compute the similarity threshold of the activated neurons as:

$$T_i = min_{j=1..m \wedge i \neq j} dist(w_j, w_i) \tag{16}$$

6. Increment the age of activated neurons.

2.3. Distance

We use a Gaussian kernel distance $K(x_1, x_2)$:

$$K(x_1, x_2) = \exp\left(\frac{-\|x_1 - x_2\|^2}{\sigma}\right) \tag{17}$$

$$dist(x_1, x_2) = 2*(1 - K(x_1, x_2)) \tag{18}$$

where σ is a predefined value.

3. Experimental results

The proposed method has been tested on synthetic and real datasets. The synthetic datasets are used to show the efficiency of our approach to extract the topological structure of the input space in the off-line, incremental or noise cases.

3.1. Synthetic datasets

We have tested the proposed method in the Aggregation data set [30] and tow moons dataset. They are a 2 dimensions dataset. The Aggregation dataset has 788 instances subdivided in 7 classes where these classes have different shapes and instances number (Table 1). The two moons dataset as its name, it has two intertwined classes on shape of the crescent (Fig. 3).

We used the two moons dataset with 4 labeled data and 106 unlabeled data in each class to exhibit the efficacy of the proposed method in the semi-supervised learning when the labeled data cannot represent the underlying structure of the particular space (Fig. 3). The algorithm parameters are presented in Table 2. Fig. 4 presents the SVM and the INNCC result during the proposed method training. From these figures we have two observations. First, as can be seen, the INNCC can extract the data structure in the first iterations. Second the correctly labeled data are added to the training labeled dataset during the training and this was through cooperation between the SVM and the INNCC.

In order to show the SVM importance in the training of our approach we tested the training of INNCC without SVM. Fig. 5 shows tow different results of INNCC without SVM, we can note that the INNCC can extract the data structure topology, however it cannot constantly classify correctly the unlabeled data without SVM. Thus the SVM has maintained the stability of INNCC classification.

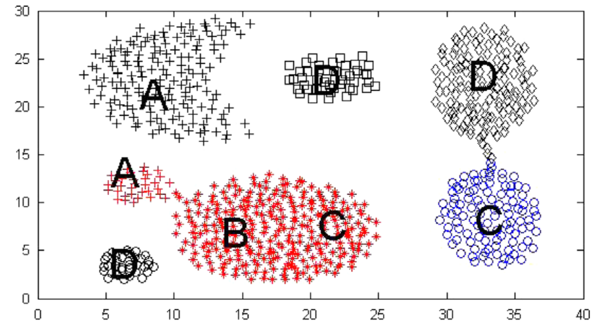


Fig. 9. Artificial dataset aggregation subdivided into 4 parts (A, B, C, and D).

To evaluate the performance of our approach, we run Transductive SVM (TSVM) [32] in using the implementation of SVM^{ligh} [32], see Fig. 6. Through the results shown in the figure we can observe two advantages of our method: First the efficacy of INNCC in the classification best than TSVM. Second in off-line learning we can benefit to two classifiers INNCC and SVM.

We used the Aggregation data set to show the efficacy of the proposed method in the incremental learning case and in the noise existence case. 10% of the dataset have been labeled and 90% of dataset is unlabeled. The algorithm parameters are presented in Table 2.

Fig. 7 shows the performance of INNCC in the topology structure representation. Fig. 7c gives how the number of neurons changes during training. The number of neurons increases. It is noticed that it is stable after the training of all the data. Random noise (25% of useful data) has been added to the data set to estimate the INNCC performance in noisy environment. As shown in Fig. 8a, overlaps exist among classes; noise is distributed over the entire data set. The proposed method represents efficiently the topology structure and gives typical prototype neurons of every class (Fig. 8b). Fig. 8c shows the neurons number stabilization even in the noise existence case.

In order to show the INNCC performance on on-line learning, we subdivide Aggregation dataset in 4 parts (A, B, C and D), as shown Fig. 9, and the INNCC learns the parts one after one.

Fig. 10 shows the capacity of the INNCC in the on-line learning. INNCC learns the new data without relearning or removing the already acquired data.

Fig. 11 shows the change of the neurons number during the on-line training. The neurons number increments when the new data arrive. For each part, the neurons number stabilizes after a few iterations.

3.2. Real datasets

The performances of the proposed method in off-line learning are studied for four UCI dataset [31] presented in Table 3. Each dataset is randomly subdivided into two subsets: 40% for training and 60% for testing. We change the labeled data percentage to show its impact in the INNCC classification performance. The accuracy of correct classification coefficient (ACC) computed by formula (19) is used to evaluate the method performances. The algorithm parameters are presented in Table 2. We repeat the training 10 times and we compute the mean and the standard deviation of ACC. The results are shown in Table 4, where α is the labeled data percentage in the training data set.

$$ACC = \frac{\sum_{i=1..N} \delta(y, f(x))}{N} \tag{19}$$

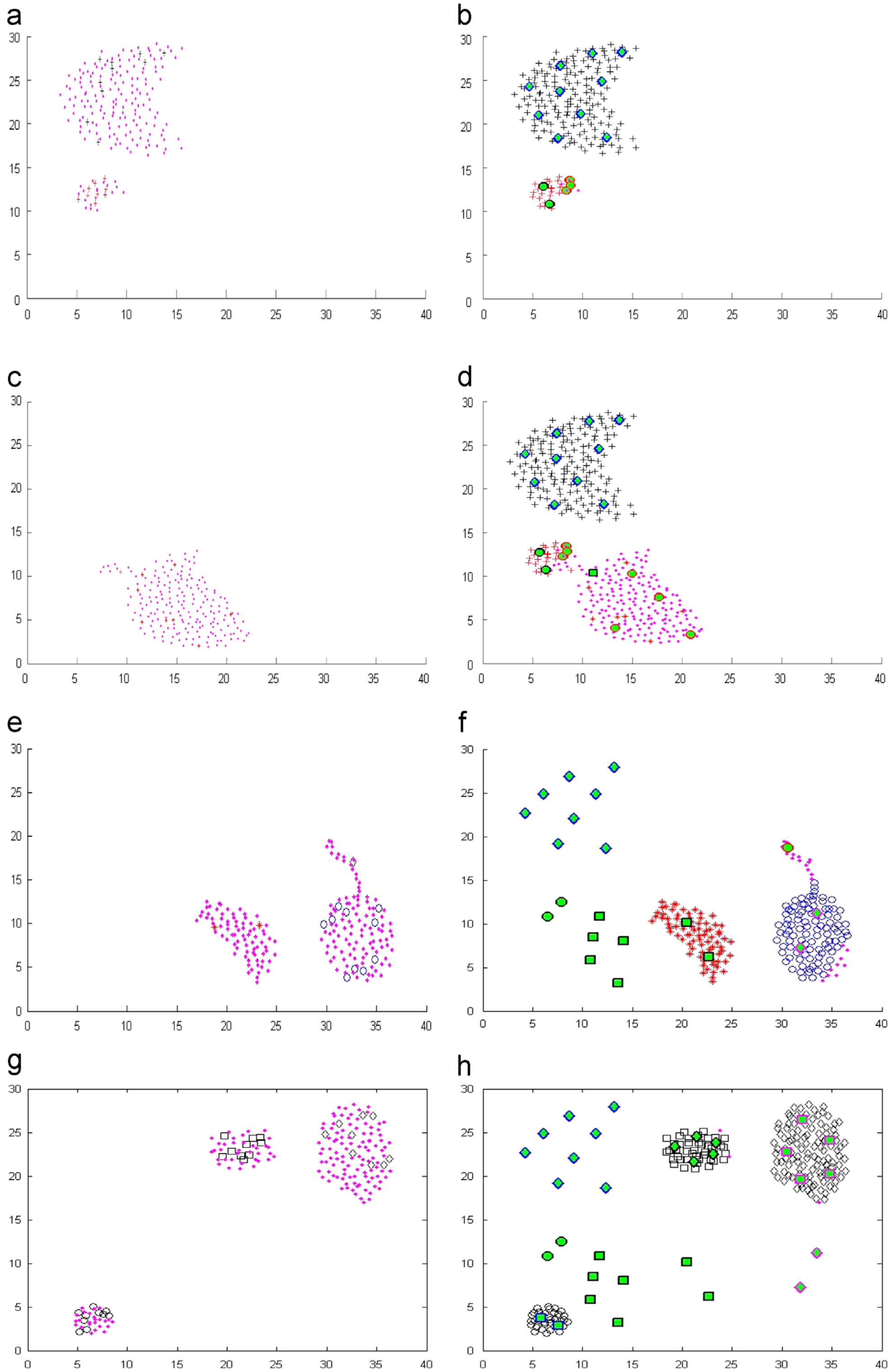


Fig. 10. (a, b, e ,f): the parts of training dataset, (c, d, g, h): INNCC results.

where

$$\delta(y,f(x)) = \begin{cases} 1 & \text{if } y = f(x) \\ 0 & \text{else} \end{cases}$$

where y is the real label of x and $f(x)$ is its calculated label. INNCC is an incremental neural network. It is able to train new data without forgetting old data. To assess INNCC performance in on-line learning, IRIS data set is subdivided in two subsets as show in Table 5. The subsets do not contain the data of the same classes. After each subset training, INNCC is tested on the two subsets. The training is repeated 10 times, ACC standard deviation and mean are computed. The results are shown in Table 6. After P1 training the ACC of P2 is approximately 0.333 because the P2 contains 1/3 of data of class 2; The INNCC has not been able to classify the class 3. It does not exist in P1. However after the INNCC training on P2 the ACC of P2 increased and the ACC of P1 remains stable. From these tables we have three observations. First, as can be seen, the proposed method gives good results in classification in off-line

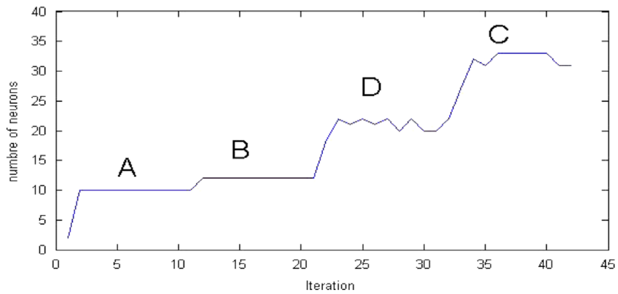


Fig. 11. The neurons number change during the on-line learning.

Table 3
Real datasets.

Data set	Instances	Attributes	Classes
IRIS	150	4	3
GLASS	214	9	6
WINE	178	13	3
Ionosphere	351	34	2

Table 4
Performance classification of INNCC on real data set.

α (%)	IRIS	GLASS	WINE	Ionosphere
10	0,863 ± 0,084	0,478 ± 0,042	0,768 ± 0,034	0,644 ± 0,037
20	0,863 ± 0,060	0,466 ± 0,053	0,732 ± 0,015	0,617 ± 0,038
30	0,889 ± 0,053	0,421 ± 0,129	0,680 ± 0,017	0,621 ± 0,039
40	0,901 ± 0,049	0,510 ± 0,047	0,680 ± 0,049	0,675 ± 0,029
50	0,912 ± 0,050	0,520 ± 0,042	0,658 ± 0,023	0,666 ± 0,035
60	0,920 ± 0,052	0,531 ± 0,032	0,632 ± 0,030	0,670 ± 0,031
70	0,921 ± 0,047	0,523 ± 0,028	0,628 ± 0,071	0,646 ± 0,033
80	0,930 ± 0,037	0,530 ± 0,029	0,624 ± 0,016	0,642 ± 0,029
90	0,931 ± 0,030	0,514 ± 0,037	0,570 ± 0,052	0,647 ± 0,028
100	0,931 ± 0,036	0,519 ± 0,034	0,618 ± 0,068	0,643 ± 0,024

Table 5
Subdivide of IRIS dataset.

Subset	P1	P2
Instances	75	75
Class	1, 2	2, 3
Instances of class 1	50	0
Instances of class 2	25	25
Instances of class 3	0	50

Table 6
Performance classification of INNCC in on-line learning

α (%)	After P1 training		After P2 training	
	P1	P2	P2	P1
10	0,957 ± 0,090	0,331 ± 0,008	0,809 ± 0,097	0,924 ± 0,093
20	0,976 ± 0,015	0,321 ± 0,013	0,855 ± 0,061	0,939 ± 0,056
30	0,991 ± 0,011	0,329 ± 0,009	0,911 ± 0,044	0,967 ± 0,030
40	0,989 ± 0,018	0,331 ± 0,008	0,889 ± 0,073	0,967 ± 0,022
50	0,989 ± 0,014	0,333 ± 0,000	0,893 ± 0,075	0,961 ± 0,025
60	0,979 ± 0,021	0,328 ± 0,011	0,900 ± 0,052	0,956 ± 0,028
70	0,983 ± 0,020	0,328 ± 0,011	0,901 ± 0,056	0,948 ± 0,037
80	0,989 ± 0,014	0,331 ± 0,008	0,913 ± 0,055	0,968 ± 0,022
90	0,996 ± 0,009	0,333 ± 0,000	0,925 ± 0,045	0,949 ± 0,031
100	0,993 ± 0,013	0,333 ± 0,000	0,935 ± 0,032	0,965 ± 0,035

learning. Second, there is a weak impact of the labeled data percentage. Third, the performance of the proposed method in incremental learning is very good. This shows that our algorithm is robust and depends less on the initial training data even if the initial model assumption is not consistent with the real data space structure.

4. Conclusion

In this paper, a new learning approach of incremental neural network improved by SVM is defined. This network is built automatically without any a priori information on the number of neurons. It ensures the both classification and clustering, it is able to learn new data without forgetting or relearning the old ones. We introduce the type of neuron to assure the incremental case, where the inactivated neurons represent the old data and the neurons are activated by the new data. We used the cluster posterior probabilities of classes to create a relation between the clusters and the classes. We use the SVM to improve and accelerate the INNCC classification during its training. The experiments on synthetic datasets and real datasets demonstrate the incremental performances. The experiments on synthetic datasets and real datasets demonstrate the incremental performances.

Acknowledgments

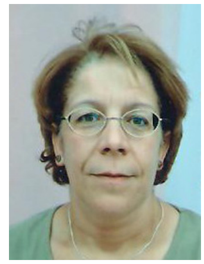
The authors would like to thank the anonymous reviewers whose valuable comment and suggestions have helped to considerably improve the quality of paper.

References

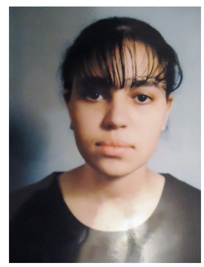
[1] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Englewood Cliffs, 1988.
[2] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley Interscience, New York, 2000.
[3] O. Chapelle, B. Schölkopf, A. Zien, Semi-Supervised Learning, vol. 2, MIT Press, Cambridge, MA, 2006.
[4] S. Basu, A. Banerjee, R.J. Rooney, Semi-supervised clustering by seeding, in: Proceedings of the 19th International Conference on Machine Learning, 2002, pp. 19–26.
[5] A. Hsu, S.K. Halgamuge, Class structure visualization with semi-supervised growing self-organizing maps, Neurocomputing 71 (2008) 3124–3130.
[6] F. Wang, Changshui. Zhang, Robust self-training semi-supervised learning, Neurocomputing 70 (2007) 2931–2939.
[7] H. Gan, N. Sang, R. Huang, X. Tong, Z. Dan, Using clustering analysis to improve semi-supervised classification, Neurocomputing 101 (2013) 290–298.
[8] Zhenning Wun, Huaguang Zhang, Jinhai Liu, A fuzzy support vector machine algorithm for classification based on a novel PIM fuzzy clustering method, Neurocomputing 125 (2014) 119–124.
[9] C. Zhang, S. Xiang, F. Nie, Y. Song, Non-linear dimensionality reduction with relative distance comparison, Neurocomputing 72 (2009) 1719–1731.
[10] I.A. Maraziotis, A semi-supervised fuzzy clustering algorithm applied to gene expression data, Pattern Recognit. 45 (2012) 637–648.
[11] W. Chen, G. Feng, Spectral clustering: a semi-supervised approach, Neurocomputing 77 (2012) 229–242.
[12] Y. Yang, Y. Yang, H. Tao Shen, Y. Zhang, X. Du, and X. Zhou, Discriminative nonnegative spectral clustering with out-of-sample extension, IEEE Trans. Knowl. Data Eng., 1760–1771, (2013).
[13] S. Grossberg, Competitive learning: from interactive activation to adaptive resonance, Cognit. Sci. 11 (1987) 23–63.
[14] S. Haykin, Neural Networks: A comprehensive foundation, Prentice Hall, New Jersey, 1999.
[15] D. Specht, Probabilistic neural networks for classification, mapping, or associative memory, in: Proceedings of IEEE International Conference on Neural Networks, vol. 1, July 1988, San Diego, CA, pp. 525–532.
[16] T. Kohoen, Self-organized formation of topologically correct feature maps, Biol. Cybern. 43 (1982) 59–69.
[17] T.M. Martinetz, Competitive Hebbian learning rule forms perfectly topology preserving maps, J. ICANN (1993) 427–434.
[18] T.M. Martinetz, S.G. Berkovich, K.J. Schulten, Neural-gas network for vector quantization and its application to time-series prediction, IEEE Trans. Neural Netw. 4 (4) (1993) 507–552.
[19] B. Fritzk, A growing neural gas network learns topologies, Adv. Neural Inf. Process. Syst. (NIPS) (1995) 625–632.
[20] J. Bruske, G. Sommer, Dynamic cell structure learns perfectly topology preserving map, Neural Comput. 7 (1995) 845–865.

[21] Y. Prudent and A. Ennaji, An incremental growing neural gas learns topologies, in: Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 05), Montreal, Canada, 2005, pp. 1211–1216.
[22] F. Shen, O. Hasegawa, An incremental network for on-line unsupervised classification and topology learning, Neural Netw. 19 (2006) 99–106.
[23] F. Shen, T. Ogura, O. Hasegawa, An enhanced self-organizing incremental neural network for online unsupervised learning, Neural Networks 20 (2007) 893–903.
[24] A. Hebboul, M. Hacini, F. Hachouf, An incremental parallel neural network for unsupervised classification, in: Seventh international workshop on systems, signal processing and their applications, WOSSPA, 2011, pp. 400–403.
[25] M. Tscherepanow, M. Kortkamp, M. Kammer., A hierarchical ART network for the stable incremental learning of topological structures and associations from noisy data, Neural Netw. 24 (2011) 906–916.
[26] F. Shen, H. Yu, Y. Kamiya, O. Hasegawa, An online incremental semi-supervised learning method, J. Adv. Comput. Intell. Band Intell. Inform. 14 (2010) 593–605.
[27] V.N. Vapnik, Statistical Learning Theory, John Wiley and Sons, New York, 1998.
[28] J. Shawe-Taylor, S. Sun, A review of optimization methodologies in support vector machines, Neurocomputing 74 (2011) 3609–3618.
[29] T.M. Cover, P.E. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory, IT13 1 (1967) 21–27.
[30] A. Gionis, H. Mannila, P. Tsaparas, Clustering aggregation, ACM Trans. Knowl. Discov. Data 1 (1) (2007) 1–30.
[31] C. Blake, E. Keogh, C.J. Merz, UCI Repository of Machine Learning Databases, Department of Information and Computer Science, University of California, Irvine, CA, 1998. ⟨http://www.ics.uci.edu/mlearn/MLRepository.html⟩.
[32] T. Joachims, Transductive inference for text classification using support vector machines, in: International Conference on Machine Learning (ICML), 1999. ⟨http://svmlight.joachims.org/⟩.

Amel Hebboul is assisting master at the University of Constantine 3, in Algeria. She received the degree of engineer and magistére in computer science from University of Constantine 1 in 2005 and 2008. She is currently a Ph.D. candidate at the University of Constantine 2. Her research interests are pattern recognition, machine learning, and neural network.



Fella Hachouf is a professor at the University of Constantine I, in Algeria with automatic and robotic laboratory. She received the degree of engineer in electrical engineering from national polytechnic school of Algiers in 1983, “Diplôme des études approfondies” (1984) and “Doctorat-ingenieur” (1988) in image processing from University of sciences and techniques, Rouen, France. In 2004, Dr. Hachouf obtained “Doctorat d'état” from university Mentouri of Constantine. Her research interests include image processing, pattern recognition quality assessment, enhancement, fuzzy logic and neural networks for image analysis.



Amel Boulemnadjel is a Ph.D. candidate in the university of Constantine 1, Algeria. She received the degree of engineer and magistére in electronique from University of Constantine 1 in 2007 and 2009. Her research interests are the subspace clustering algorithms, active learning and image analysis.