A BOOSTING CLASSIFICATION APPROACH BASED ON SOM

Hoa Dinh Nguyen
Học Viện Công Nghệ Bưu Chính Viễn Thông

Abstract **-** Self-organizing map (SOM) is well known for its ability to visualize and reduce the dimension of the data. It has been a useful unsupervised tool for clustering problems for years. In this paper, a new classification framework based on SOM is introduced. In this approach, SOM is combined with the learning vector quantization (LVQ) to form a modified version of the SOM classifier, SOM-LVQ. The classification system is improved by applying an adaptive boosting algorithm with base learners to be SOM-LVQ classifiers. Two decision fusion strategies are adopted in the boosting algorithm, which are majority voting and weighted voting. Experimental results based on a real dataset show that the newly proposed classification approach for SOM outperforms traditional supervised SOM. The results also suggest that this model can be applicable in real classification problems.[1]

Keywords - Self organizing map, learning vector quantization, adaptive boosting, weighted majority voting.

I. INTRODUCTION

The Self-Organizing Map (SOM), which is also known as Kohonen network [1], is an ordered mapping from a set of given multidimensional data samples onto a regular, usually two-dimensional feature space. SOM is based on learning by self-organization which is a process of automatically changing the internal structure of a system. SOM applies the idea of competitive learning and Kohonen rule. During the training process, a data item will be mapped into the node whose parameters are most similar to the data item, i.e., has the smallest distance from the data item in some measurement metric.

Like a codebook vector in vector quantization, the model is then usually a certain weighted local average of the given data items in the data space. But in addition to that, when the models are computed by the SOM algorithm, they are more similar at the nearby nodes than between nodes located farther away from each other on the grid. In this way the set of the models can be regarded to constitute a similarity graph and structured 'skeleton' of the distribution of the given data items.

The SOM was originally developed for the visualization of distributions of metric vectors, such as ordered sets of measurement values or statistical attributes, but it can be shown that a SOM-type mapping can be defined for any data items, the mutual pairwise distances of which can be defined. Examples of non-vector data that are amenable to this method are strings of symbols and sequences of segments in organic molecules [17].

Since it is first introduced about 3 decades ago, SOM has not seemed to lose its attraction. There has been a huge number of International Workshops hold worldwide and dozens of publications by a lot of researchers and scientists in great attempts to experiment SOM on new-arising big data problems such as bioinformatics, textual document analysis, outlier detection, financial technology, robotics, pattern recognition, and much more. So far, many affords and trials have been made to utilize SOM to apply for clustering and classification problems. However, when compared to some other machine learning algorithms, SOM is still not an attractive solution for classification tasks due to its low classification performance results, even though SOM is a simple and easy to implement tool.

This research aims at improving the classification capability of the SOM by introducing a new integration between SOM and learning vector quantization (LVQ) algorithm, called SOM-LVQ model. Additionally, adaptive boosting algorithm (Adaboost) is applied to improve the performance of the system. In this algorithm, sequential SOM-LVQ classifiers are generated then combined together using either majority voting or weighted voting strategies. Weighted voting strategy is a new contribution of this research, in which each base classifier is assigned a weight dynamically based on its node selected as the best matching unit in testing process. Experiments are conducted using a real dataset and experimental results confirm that the newly proposed approach outperform traditional SOM models in solving classification problems.

The structure of this paper is organized as follows. Section 2 provides all background information on SOM and LVQ algorithms. This section also presents the proposed SOM-LVQ model in detailed. Section 3 introduces Adaptive boosting algorithm with two fusion strategies, majority voting and weighted voting. Experimental setup and results are presented in section 4. All discussion and analysis on the empirical performance of the new framework is also included in this section. The paper is concluded in section 5.

Tác giả liên hệ: Nguyễn Đình Hóa
Email: hoand@ptit.edu.vn
Đến tòa soạn: 6/2020, chỉnh sửa: 7/2020, chấp nhận đăng: 7/2020.

II. SELF-ORGANIZING MAP

The self-organizing system in SOM is a set of nodes (or neurons) connected to each other via the topology of, typically, rectangle or hexagon, as illustrated in Figure 1. This set of nodes is called a map. Each neuron has several neighbors (4 or 8 with rectangular topology and 6 with hexagonal topology). In this research, the rectangular topology is utilized, and it is assumed that each neuron has at most 8 neighbors, or fewer if it lies in the edges or corners of the map. Each neuron contains a vector of weights of the same dimension as the input $\boldsymbol{x}$.

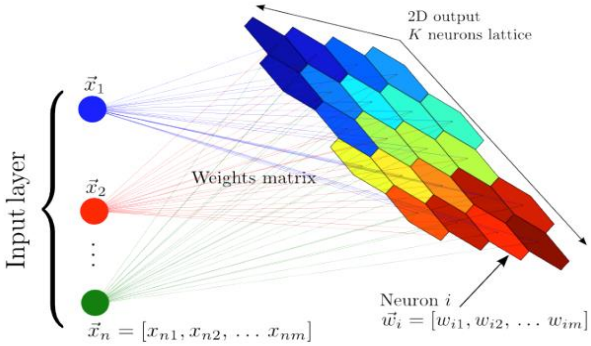

Figure 1: Illustration of an SOM [10]

Let's denote the input vector j as $\boldsymbol{x_j} = [x_{j1}, x_{j2}, \ldots, x_{jm}]^T$, and the weight vector of neuron i as $\boldsymbol{w}_i = [w_{i1}, w_{i2}, \ldots, w_{im}]^T$, $i = 1,2, \ldots n$, where n is the total number of neurons in the map.

At each training step, one randomly selected input vector $\boldsymbol{x}$ from training dataset is introduced to the map. The different between $\boldsymbol{x_j}$ and each neuron in the map is calculated using the Euclidean distance $D(\boldsymbol{x_j}, \boldsymbol{w}_i)$. The neuron having the smallest distance to the sample is called the winning node or the best-matching unit (BMU). The weight vector of the BMU is then updated by a learning rule [2] as:

$$\boldsymbol{w}_i(t) = \boldsymbol{w}_i(t-1) + \alpha(t).D\left(\boldsymbol{x_j}(t), \boldsymbol{w}_i(t-1)\right) \quad (1)$$

Where $\alpha(t)$ is the learning rate, which normally decreases during the training process as $\alpha(t) = \frac{\alpha_0}{1+decayrate*t}$. By this learning rule, the BMU is moved closer to the input sample. In order to facilitate the training process, the BMU's neighbors are also updated. However, only neurons lying inside the BMU's neighborhood are updated. Learning rate and BMU's neighboring radius are decreased after each training iteration. As a result, SOM can be considered as a more flexible version of the K-means algorithm. The learning rule for BMU's neighboring nodes is as follows.

$$\boldsymbol{w}_h(t) = \boldsymbol{w}_h(t-1) + \theta_h(t).\alpha(t).D\left(\boldsymbol{x_j}(t), \boldsymbol{w}_h(t-1)\right) \quad (2)$$

Where $\theta_h(t)$ is neighborhood function determining the number of neighboring neurons being updated at iteration t for $\boldsymbol{x_j}$, and how much they are adjusted. $\theta_h(t)$ is also a decaying function, which can be presented as $\theta_h(t) = e^{\frac{-D(\boldsymbol{w}_i, \boldsymbol{w}_h)^2}{2\alpha(t)^2}}$, where $D(\boldsymbol{w}_i, \boldsymbol{w}_h)$ is the distance from node h to the BMU i. As time (i.e. number of iterations) increases, the neighboring range decreases in an exponential manner and the neighborhood shrinks appropriately. In each iteration, only the winning node and nodes inside its neighborhood have their weights adapted. All other nodes have no change in their weights.

In general, SOM is an unsupervised clustering algorithm and is mainly applied for data clustering problems since each neuron represents one or some patterns of training data. In case the training data is labeled, the labels of the neurons after training process can be assigned based on the labels of the neighboring training samples. However, it is impossible to obtain the optimized classification results. For example, when the unsupervised SOM experiment was conducted to classify Iris dataset, the classification accuracy was only from about 75.0% to 78.35% [3]. There are two main issues that must be solved in order to improve the performance of SOM in supervised classification problems. First, the network parameters should be initialized properly. Second, the SOM learning process should update parameters based on not only inputs but also information from expected outputs.

Supervised SOM

In order to tackle supervised classification problems, traditional SOM must be modified to adapt to the classification tasks. There have been many versions of supervised versions of SOM proposed in the literature. Some supervised SOM solutions has been developed to solve textual document analysis problems [3, 4]. Kurasova [5] introduces an extension of SOM, called WEBSOM to distinguish between different textual document collections. This new kind of supervised SOM can recognize similar documents from the others. Suganthan [4] develops a Hierarchical Overlapped SOM (HOSOM) for handwritten character recognition and has gained very good results. In his approach, an additional neuron layer is added to each winning node of the initial neuron layer, which may cause high computational cost due to the growth of the number of the neuron layers. In order to enable SOM to cover the outlier detection problem, the travelling salesman approach can be used [6]. Additionally, SOM can be combined with KNN to formulate a new version of supervised SOM [7]. Meanwhile, k-means algorithm can be utilized to formulate a simple version of supervised SOM [8].
Kohonen [9] introduced the model "LVQ-SOM", which combine traditional SOM with learning vector quantization (LVQ) algorithm. In this model, each output neuron is assigned one label, and its parameters are adjusted toward the distributed region of training data of the same types.

In this research, a new combination of SOM and LVQ algorithms are proposed, in which the integration order of these two algorithms is different from [9]. Following section presents the principles of LVQ algorithm and the proposed LVQ-SOM model for classification.

Learning vector quantization (LVQ)

LVQ is a supervised neural network learning algorithm used for classification without any topology structure. Each output neuron of LVQ represents a known category of the data. Specifically, each LVQ's winning neuron

represents a subclass and several neurons together create a class [2, 11].

Let $\boldsymbol{x_j} = [x_{j1}, x_{j2}, \dots, x_{jm}]^T$ be an input vector having label T_j, and the weight vector of neuron i be $\boldsymbol{w_i} = [w_{i1}, w_{i2}, \dots, w_{im}]^T$. The neuron i is assigned a label C_i. A five-step LVQ algorithm can be presented as follows.

Step 1: Randomly initialize the weights for neurons

Step 2: Select a random data sample $\boldsymbol{x_j}$ and find its BMU

Step 3: Update the weights of BMU based on the following set of rules:

If $T_j = C_i$ then

$$\boldsymbol{w}_i(t) = \boldsymbol{w}_i(t-1) + \alpha(t).D\left(\boldsymbol{x_j}, \boldsymbol{w}_i(t-1)\right) \quad (3)$$

(the weights of BMU is moved towards the input $\boldsymbol{x_j}$ having the same label)

If $T_j \neq C_i$ then

$$\boldsymbol{w}_i(t) = \boldsymbol{w}_i(t-1) - \alpha(t).D\left(\boldsymbol{x_j}, \boldsymbol{w}_i(t-1)\right) \quad (4)$$

(the weights of BMU is moved away from the input $\boldsymbol{x_j}$ having different label)

Step 4: Update the learning rate $\alpha(t) = \frac{\alpha_0}{1+decayrate*t}$

Step 5: Repeat step 2 to 4

Different from the model "LVQ-SOM" proposed by Kohonen [9], the learning process during LVQ algorithm of this proposed approach does not involve just one winning nodes. Instead, all neighboring nodes of BMU are also updated. Specifically, Step 3 of the LVQ algorithm is modified such that the neighboring nodes of the BMU are updated the same as in modified SOM algorithm. Here, the neighboring function $\theta_j(t)$ of SOM algorithm in equation (2) is used. The revised Step 3 of LVQ algorithm is presented as bellow.

Step 3: Update the weights of BMU i based on the following set of rules:

If $T_j = C_i$ then equation (3) is used to move the weight vector of $\boldsymbol{w}_i$ towards the input $\boldsymbol{x_j}$.

If $T_j \neq C_i$ then equation (4) is used to move the weight vector of $\boldsymbol{w}_i$ away from the input $\boldsymbol{x_j}$.

Update the weights of neighboring nodes h of BMU

$$\boldsymbol{w}_h(t) = \boldsymbol{w}_h(t-1) + k_h.\theta_h(t).\alpha(t).D\left(\boldsymbol{x_j}, \boldsymbol{w}_h(t-1)\right) \quad (5)$$

$$\text{Where } k_h = \begin{cases} 1 & if \quad T_j = C_h \\ -1 & if \quad T_j \neq C_h \end{cases} \quad (6)$$

This revised version of LVQ algorithm helps speed up the learning process and reduce the number of "dead" neurons.

SOM-LVQ model

SOM model is a simple algorithm and is useful for data visualization and clustering problems. Meanwhile, LVQ is useful for classification problems, but its training process can become time consuming and may not converge. This is because LVQ algorithm depends on how the initial weight vectors are arranged. If the neuron is in the middle region of a class that it does not represent, its weight vector may have to travel through a long path to get out of its surrounding region. Because the weights of such a neuron will be repulsed by vectors in the region it must cross. As a result, that neuron may not be able to the region of correct labeled data. This problem can be solved by a proper label assigning strategy. The combination of SOM and LVQ is a promising solution to improve the classifying capability of both SOM and LVQ algorithms.

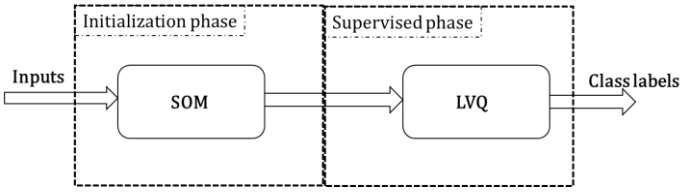

Figure 2: Structure of SOM-LVQ model

Figure 2 illustrates the way that SOM and LVQ algorithms are combined to form a supervised version of SOM. The proposed combined SOM-LVQ model is summarized as a three-stage algorithm as follows.

Stage 1: SOM algorithm is applied to cluster the nodes accordingly with the training data,

Stage 2: Each node of the trained neuron map is initialized a label according to the label of the closest training sample.

Stage 3: The modified LVQ algorithm is applied to train the nodes.

In fact, there is always an overlap between the data of different classes. As a result, there can be one part of missed classified test samples fall into the overlapping regions of different classes. In order to improve the classification accuracy, the input sample needs to be put under different angles in order to exploit all valuable information for the classification process. Specifically, instead of using just one SOM-LVQ model to classify the testing data, several local models can be generated from different portions of the training dataset, which provide several local decisions on the class of the testing data. At the fusion stage, the class having the highest probability among all classes will be decided for the input data. This is what is done in following boosting algorithm.

III. ADAPTIVE BOOSTING ALGORITHM

Boosting algorithms aim at to improve the prediction power by training a sequence of weak models, each compensating the weaknesses of its predecessors. Adaptive boosting, as known as AdaBoost [12], is a boosting algorithm developed for classification problems. The algorithm is based on a set of base learners, each of which is created from a sub set of training data. Initially, each data sample is assigned an equal weight. As a result, in the first iteration, a base learner is generated from a randomly picked training data subset. In each iteration, AdaBoost identifies miss-classified data samples from the base learner in that iteration, then increases their weights (and decreases the weights of correct points, on the contrary), so that the next base learner will focus more on the examples that previous base learners misclassified. Finally, all base learners are combined following a deterministic strategy to create a strong learner which eventually improve the prediction power of the model.

Originally, adaptive boosting algorithm is developed for binary classification problem [14]. However, the classification problem gets more complicated when it comes to multi-class classification. One simple solution to

this is to break down the problem in to several two-class problems. Zhu et al. [15] introduced an algorithm for Adaboost that generalizes the original binary classification to the multi-class problem, called SAMME. Motivated by SAMME algorithm, this research also focuses on multi-classification problem with the use of SOM-LVQ as the base learner.

Adaptive boosting can be applied to any supervised machine learning algorithm. However, it is pointed out by Hastie et al. [13] that Adaboost algorithm works well with weak learners, and decision tree model is especially suited for boosting. Adaboost mainly focuses at reducing bias. The base learners that are often considered for boosting are weak models with low variance but high bias. The most important motivation for the use of low variance but high bias models as weak learners for boosting is that these models are in general less computationally expensive to fit. Indeed, as computations to fit the different models can't be done in parallel, it could become too expensive to fit sequentially several complex models.

SOM-LVQ can be considered as a weak learner since it applies a naïve method (usually, majority voting) to label its nodes, therefore, it often classifies incorrectly samples positioning in the border regions of different classes. In this research, supervised SOM, aka. SOM-LVQ, model is used as a weak learner for the Adaboost algorithm. In Adaboost algorithm, multiple SOM-LVQ models are generated sequentially. Combining the outputs of these models can follow one of pre-determined strategies as bellow.

Majority voting strategy

In this strategy, all base learners have equal weights. Given a test sample, multiple base learners will provide multiple classification answers based on the label of the BMU of each base learner. These answers will be fused to make the final decision as the class label having the most count from all base learners.

Weighted voting strategy

Different from majority voting strategy, in this weighted voting, each base learner is assigned a weight to its answer based on the weight of the BMU in that base learner. Specifically, after training, each node of the SOM-LVQ model is assigned a class label together with a weight determining how confident that node can represent the label of the sample closest to it. This weight is set as the number of times the node is selected as the BMU during training process. If a node never wins during the training, its weight is set to a very small value. At the fusion stage, all weights belonging to each class label is summed up and the class with the highest total weight with be decided.

IV. EXPERIMENTAL RESULTS

Dataset

In this research, the Ecoli dataset collected from the UCI Machine Learning Repository [16] is used. The dataset contains protein localization sites. There are 336 instances with 8 attributes in the dataset. Each sample has a class representing the localization site of protein. The attribute information is given as follows.

1. Sequence Name: Accession number for the SWISS-PROT database
2. mcg: McGeoch's method for signal sequence recognition.
3. gvh: von Heijne's method for signal sequence recognition.
4. lip: von Heijne's Signal Peptidase II consensus sequence score. Binary attribute.
5. chg: Presence of charge on N-terminus of predicted lipoproteins. Binary attribute.
6. aac: score of discriminant analysis of the amino acid content of outer membrane and periplasmic proteins.
7. alm1: score of the ALOM membrane spanning region prediction program.
8. alm2: score of ALOM program after excluding putative cleavable signal regions from the sequence.

There are 8 class labels in the dataset. Those labels are distributed as in Table 1 as follows.

Table 1. The distribution of data samples in the dataset

Class code	Class name	Number of samples
0	cp (cytoplasm)	143
1	im (inner membrane without signal sequence)	77
2	pp (perisplasm)	52
3	imU (inner membrane, uncleavable signal sequence)	35
4	om (outer membrane)	20
5	omL (outer membrane lipoprotein)	5
6	imL (inner membrane lipoprotein)	2
7	imS (inner membrane, cleavable signal sequence)	2

As presented in Table 1, the majority of the samples fall into the first 5 classes. In the experimental results, classification performance of the system for classes 5, 6, 7 can be negligible.

In order to evaluate the performance of the classification model, some metrics are used as follows.

Precision is the number of correct positive samples divided by the number of positive results predicted by the model.

$$precision = \frac{true\ positives}{true\ postives + false\ positives} \quad (7)$$

Recall is the number of correct positive samples divided by the total number of actual positive samples.

$$recall = \frac{true\ positives}{true\ postives + false\ negatives} \quad (8)$$

In any classification model, the accuracy score is an important metric to present the quality of the model. The classification accuracy is simply the rate of correct classifications. However, in this dataset, there is a significant imbalance among all classes of the data, the accuracy is not necessary the precise score to present the performance of the system. Instead, the F1 score can be used here.

F1-score is the harmonic mean of precision and recall.

$$F_1 = 2\,\frac{precision \times recall}{precision + recall} \quad (9)$$

Results and discussions

The dataset is divided into training (with 75% samples) and testing (with 25% samples) subsets randomly. SOM-LVQ base learners are generated with size of 10 x 10 neurons. Table 2 presents the classification results for different setups.

Supervised SOM is generally a modified version of the traditional SOM, in which each node is assigned a label corresponding the class of its closest training sample after training process. This supervised SOM model is widely used in the literature.

Experimental results show that the proposed SOM-LVQ outperforms traditional supervised SOM commonly used in the literature in terms of all performance measurements. Additionally, the boosting algorithm significantly improves the quality of the SOM-LVQ model. This is due to two main reasons.

First, in boosting algorithm, multiple based learners are created sequentially based on the missed classified samples from previous models. This helps base classifiers learn different knowledge from different training subsets, especially the knowledge from the samples that may contain different relationship between inputs and outputs, which results in their missed classifying results.

Seconds, each base learner is created from a small subset of training data. This helps each learner capture different nature characteristics of the data. As a result, when the outputs of all base learners are combined, these different information angles are put into a pool and provide a better decision than if only one classification model is used.

Table 2: Classification results of different model setups.

Model	Model setup	Precision (%)	Recall (%)	F1 (%)
SOM - LVQ	Single	73.4	79.6	75.3
	Boosting (Majority	85.7	83.9	83.9
	Boosting (Weighted voting)	87.4	85.2	86.3
Supervised SOM	Single model	69.5	62.7	64.6

Regarding Adaboost algorithm, the weighted voting works slightly better than majority voting when it utilizes the relationship between each node and the training data during training process. Specifically, if one node is more frequently selected as the BMU during training than other nodes, its weight vector is closely related to the input data, which means it is more relevant to represent the region of its class in the training data. Assigning a classification weight to each node is an effective way to reflect that relationship and helps improve the classification performance. Here, each base classifier does not have one fixed weight. Instead, it has multiple weights corresponding to multiple nodes inside. This dynamic weighting method is designed to adapt with the nature of the data, in which data samples of the same class may have different input distributions.

As expected, the traditional supervised SOM has the worse classification performance since their nodes are just trained to present the clusters of the input data. Each node is assigned with the label of its closest training sample. As a result, the labeled nodes are not representing the region of their respective class regions. SOM-LVQ is much better than supervised SOM since their nodes are arranged by SOM training process, then assigned labels before LVQ training. This helps each node in the model better represent the region of its class. However, if only one single SOM-LVQ is used, its nodes cannot present all possible nature characteristics of the data.

V. CONCLUSIONS

In this research, a new framework to improve the classification capability of SOM is introduced. SOM algorithm is good at presenting the clusters of the input data, while LVQ algorithm is good for the process of moving labeled nodes to its representative class region. The combination of SOM and LVQ algorithm in the proposed method is empirically shown to be effective compared to commonly used supervised SOM. Adaptive boosting algorithm with two different fusion strategies is also proposed in this research. This approach seems to be effective in utilizing the nature information of the data by the creation of multiple base SOM-LVQ models sequentially. Weighting each learner by assigning different weight values to different nodes inside it is a flexible way to present how close the relation between each learner and the input data is. Experimental results show that the new approach significantly improves the classification performance of the SOM structures. In our future work, some more different real applications of the proposed classification framework will be investigated using different real datasets.

REFERENCES

[1] T. Kohonen. "Self-Organized Formation of Topologically Correct Feature Maps", Biological Cybernetics. 43 (1), pp. 59 - 69, 1982.

[2] M.T. Hagan, H.B. Demuth, M.H. Beale, O.D. Jesus. “Neural network design (2nd edition)”, ISBN-10: 0-9717321-1-6, ISBN-13: 978-0-9717321-1-7, 1996.

[3] A. Rauber and D. Merkl, "Automatic Labeling of Self-Organizing Maps: Making a Treasure-Map Reveal Its Secrets", Methodologies for Knowledge Discovery and Data Mining, p. 228 – 237, 1999.

[4] P.N. Suganthan. "Hierarchical overlapped SOM's for pattern classification", IEEE Transactions on Neural Networks, vol. 10, pp. 193-196, 1999.

[5] O. Kurasova, "Strategies for Big Data Clustering", 2014 IEEE 26th International Conference on Tools with Artificial Intelligence, 2014.

[6] P. Stefanovič, O. Kurasova. "Outlier detection in self-organizing maps and their quality estimation", Neural Network World, 28 (2), pp. 105-117, 2018.

[7] L.A. Silva, E.D.M. Hernandez. "A SOM combined with KNN for classification task", Proceedings of the International Joint Conference on Neural Networks, pp. 2368-2373, 2011.

[8] M. Mishra, H.S. Behera. "Kohonen Self Organizing Map with Modified K-means clustering For High Dimensional Data Set", International Journal of Applied Information Systems, pp. 34-39, 2012.

[9] T. Kohonen, "Self-Organizing Maps", 3rd Edition ed., 2000, pp. X-XI.

[10] M. C. Kind and R. J. Brunner, "SOMz: photometric redshift PDFs with self-organizing maps and random atlas", 2013.

[11] E. D. Bodt, M. Cottrell, P. Letrémy, and M. Verleysen, "On the use of self-organizing maps to accelerate vector quantization," *Neurocomputing* , vol. 56, pp. 187-203, 2004.
[12] R. E. Schapire and Y. Freund, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *journal of computer and system sciences*, pp. 119-139, 1996.
[13] T. Hastie, R. Tibshirani and F. Jerome, The Elements of Statistical Learning, 2nd edition, Stanford, California: Springer, 2008, p. 340.
[14] P. Dangeti, Statistics for Machine Learning, Birmingham: Packt Publishing, July 2017.
[15] J. Zhu, H. Zou, S. Rosset and T. Hastie, "Multi-class AdaBoost," *Statistics and Its Interface* , vol. 2, pp. 349-360, 2009.
[16] "UCI Machine Learning Repository," [Online]. Available: https://archive.ics.uci.edu/ml/index.php.
[17] T. Kohonen, P. Somervuo. “How to make large self-organizing maps for nonvectorial data”, Neural Networks, 15 (8-9), pp. 945-52, 2002.

MỘT PHƯƠNG PHÁP NÂNG CAO KHẢ NĂNG PHÂN LOẠI DỮ LIỆU CỦA SOM SỬ DỤNG THUẬT TOÁN BOOSTING

Tóm tắt: Bản đồ tự tổ chức (SOM) được biết đến là một công cụ hữu hiệu trong việc trực quan hóa và giảm kích thước của dữ liệu. SOM là công cụ học không giám sát và rất hữu ích cho các bài toán phân cụm. Bài báo này trình bày về một cách tiếp cận mới cho bài toán phân loại dựa trên SOM. Trong phương pháp này, SOM được kết hợp với thuật toán huấn luyện lượng tử hóa vectơ (LVQ) để tạo thành một mô hình mới là SOM-LVQ. Mộ hình phân loại dữ liệu sử dụng SOM-LVQ được tiếp tục cải tiến bằng cách áp dụng thuật toán tăng cường thích ứng (Adaboost) sử dụng SOM-LVQ làm các bộ phân loại cơ sở. Để kết hợp các kết quả từ các bộ phân loại cơ sở, hai kỹ thuật được áp dụng bao gồm bỏ phiếu theo đa số và bỏ phiếu theo trọng số. Kết quả thử nghiệm dựa trên bộ dữ liệu thực tế cho thấy phương pháp phân loại mới được đề xuất nhằm cải tiến SOM trong nghiên cứu này vượt trội hơn mô hình SOM truyền thống. Kết quả cũng cho thấy khả năng ứng dụng thực tế của mô hình này là rất khả quan.

Từ khoá: Bản đồ tự tổ chức, học lượng tử hoá vector, thuật toán tăng cường, kết hợp theo trọng số.

Hoa Dinh Nguyen earned bachelor and master of science degrees from Hanoi University of Technology in 2000 and 2002, respectively. He got his PhD. degree in electrical and computer engineering in 2013 from Oklahoma State University. He is now a lecturer in information technology at PTIT. His research fields of interest include dynamic systems, data mining, and machine learning.