Learning of alphabets using Kohonen's Self organized featured map

Amit Goyal¹, Ankita Lakhanpal² and Shaveta Goyal³
¹²Department of Electronics and Communication Engineering, Maharishi Markandeshwar University, Sadopur, Ambala, Haryana.
³ Department of Electronics and Communication Engineering, RIMT-IET, Mandi Gobindgarh, Punjab.

Abstract
Artificial neural networks are one of the widely used automated techniques and they yield high accuracy. Kohonen maps and Counter Propagation Neural Networks are the most popular learning strategies based on Artificial Neural Networks. The Self Organizing map has attracted an ever increasing amount of interest among researchers and practitioners in wide variety of fields. The SOM has been analyzed extensively, a number of variants have been developed and, perhaps most notably, it has been applied extensively within fields ranging from engineering sciences to medicine, biology, and economics. Kohonen Maps (or Self Organizing Maps) are basically self-organizing systems which are capable to solve the unsupervised rather than the supervised problems, while Counter-propagation Artificial Neural Networks are very similar to Kohonen maps, but an output layer is added to the Kohonen layer in order to handle supervised modeling. In this paper, learning of alphabets with the help of Kohonen Self Organizing Map is described. We use a self-organizing map for feature extraction of alphabets, thus the relative feature of them are automatically defined based on arrangement of the competing units. Our proposed data structure and algorithm reduced the SOM computational complexity by several order of magnitude.

Keywords: Artificial Neural Network, Self Organizing Maps, Unsupervised Learning, MATLAB

1. INTRODUCTION
Kohonen maps (or Self Organizing Maps, SOMs) are one of the most popular learning strategies among the several Artificial Neural Networks algorithms proposed in literature [1]. They were developed in 1982 by Tuevo Kohonen, a professor emeritus of the Academy of Finland. The earliest applications were mainly in the area of engineering. Later the algorithm has become progressively more accepted as a standard data analysis method in a wide variety of fields that can utilize unsupervised learning: clustering, visualization, data organization, characterization, and exploration. The robustness of the SOM algorithm and its appealing visualization effects has also made it a prime candidate in several large-scale information categorization and visualization projects. Self-Organizing Maps are aptly named, “Self-Organizing” is because no supervision is required. Their uses are increasing related to several different tasks and nowadays they can be considered as an important tool in multivariate statistics [2]. SOMs learn on their own through unsupervised competitive learning. “Maps” is because they attempt to map their weights to conform to the given input data. It consists of local data models located at the nodes of the low-dimensional map grid. The models are adapted via competitive learning process. The nodes in a SOM network attempt to become like the inputs presented to them. In this sense, this is how they learn. They can also be called “Feature Maps”, as in Self-Organizing Feature Maps. Retaining principle ‘features’ of the input data is a fundamental principle of SOMs, and one of the things that makes them so valuable. Specifically, the topological relationships between input data are preserved when mapped to a SOM network. The SOM is an unsupervised method for forming a representation of data [3, 4]. It consists of local data models located at the nodes of the low-dimensional map grid. The models are adapted via competitive learning process.

There are two approaches for constructing the SOM for a new data set. One is to modify the SOM algorithm itself. Another possibility is to convert the data into feature vectors so that the comparison and averaging can be done using familiar Euclidean distance and arithmetic averaging. This is the approach taken in the current study, where the SOM is constructed for alphabet recognition. In the current work, the SOM algorithm is very close to the alphabet recognition. The only difference is that instead of single fixed dimensional feature vectors as input and models, the models associated with the SOM nodes are vector sequences with variable lengths [5].

The algorithm for the Self Organizing Map neural network, which is used for alphabet learning, is elaborated in the MATLAB environment.

2. METHODOLOGY
In this paper, two stages of development of the proposed alphabet learning system which include; data acquisition, feature extraction, using Self Organizing neural network are presented. Experiments were performed with 18 alphabet samples (English uppercase) which are created in the MATLAB with the desired dimensions. The results of the proposed system showed better learning performance compared with similar work in literature.
2.1 Character Acquisition

The alphabets in this work are first created in the MATLAB. The dimensions of the characters are adjusted according to requirement. The Self Organizing Map only learns the alphabets of specified dimensions. Alphabets considered were 18 upper case (A-R).

2.2 Feature Extraction

For extracting the feature, the Self Organizing neural network based feature extraction is used. The major advantage of this approach stems from its robustness to small variation, ease of implementation and provides good learning rate. Self Organizing neural network based feature extraction method provides good result even when certain pre-processing steps like filtering, smoothing and slant removing are not considered.

Kohonen Self Organizing Map has a feed-forward structure with a single computational layer of neurons arranged in rows and columns. Each neuron is fully connected to all the source units in the input layer.

![Figure 1: Self Organizing Map Structure.](image)

The aim is to learn a feature map from the spatially continuous input space, in which our input vectors live, to the low dimensional spatially discrete output space, which is formed by arranging the computational neurons into a grid. This mapping is performed by use of competitive learning, often called winner-takes-it-all strategy. The main steps of the training procedure of SOM are presented on the flow chart given in Fig. 2.

![Figure 2: Flow-chart with representation of some of the important steps in the training process.](image)

In training process of Kohonen self-organizing maps the vectors with \(N\) input variables are compared only with the weights of neuron in the Kohonen layer [13]. In each training step, samples are presented to the network, one at a time. For each sample \(x_i\), the most similar neuron (i.e. the winning neuron) is selected on the basis of Euclidean distance [6]. Once the winning neuron \(c\) is found among the neurons in the Kohonen layer, the weights of both winning neuron and the neighborhood neuron are adjusted using suitably selected learning rate \(\eta(t)\) and neighborhood function. The neighborhood function is of rectangular and hexagonal shapes. The neighborhood function includes the learning rate function \(\eta(t)\) which is a decreasing function of time and the function that dictates the form of the neighborhood function. The form of the latter function also determines the rate of change around the winner unit. The neighborhood function can be written as

\[
h_{ci}(t) = \alpha(t) \cdot \exp(- \frac{||r_i - r_c||^2}{2\sigma(t)^2})
\]

in the case of the Gaussian neighborhood function around the winner neuron \(c\).

![Figure 3: Gaussian and Bubble neighborhood functions](image)
A variety of neighborhood functions can be used. We can constrain the neighborhood function to be non-increasing around the winner unit \( c \). One choice for a neighborhood function is to use a Gaussian kernel around the winner neuron as described above. This is computationally demanding as the exponential function has to be calculated, but can well be approximated by the “bubble” neighborhood function. The bubble neighborhood function is a constant function in the defined neighborhood of the winner neuron, that is, every neuron in the neighborhood is updated the same proportion of the difference between the neuron and the presented sample vector. The bubble neighborhood function is a good compromise between the computational cost and the approximation of the Gaussian. In the Figure 3 the two used forms of the neighborhood function are illustrated. In the left, we see the form of a Gaussian (a), on the right the bubble function (b).

The weights in Kohnen self organizing maps are updated as follows

\[
\begin{align*}
    w_{ij}^{new} &= w_{ij}^{old} + \eta(t) [x_i - w_{ij}^{old}] \\
    \end{align*}
\] (2)

The learning rate is non-increasing function \([10]\), which defines the intensity of the changes of the weights during the training process. Some of the commonly used functions, which define the change of learning rate with time, are linear, power, inverse functions. The intensity of correction of weights is determined by the shape of the neighborhood function. The width of the neighborhood defined by the topological distance between the central neuron and farthest neuron affected by the correction, during the training, also decreases.

### 2.3 Learning Algorithm of Self Organizing Map

Assume output nodes are connected in an array (usually 1 or 2 dimension) and the network is fully connected, means all nodes in the input layer are connected to all nodes in the output layer. Use the competitive learning algorithm as follows.

1) Set learning rate \( \eta(t) \) and initialize weight matrix.
2) While stopping condition do step (3) to (7).
3) For each input vector \( x \), do step (4) to (6).
4) For each \( j \), compute the squared Euclidean error distance as follows

\[
D(j) = \sum_i (w_{ij} - x_i)^2, \quad \text{where} \quad i = 1 \ldots n, j = 1 \ldots n
\] (3)

5) Now find index \( j \), when \( D(j) \) is minimum.
6) For all the unit \( j \), update weights as follows

\[
w_{ij}^{new} = w_{ij}^{old} + \eta(t) [x_i - w_{ij}^{old}]
\]

7) Test the stopping condition.

### 3. RESULTS

The performance of the Kohnen Self Organizing Map Neural Network for learning of character has been tested for various alphabets. More than 18 experiments were conducted by varying the values of the parameters namely training rate coefficients and the number of neurons in the Kohnen layer of the Network. A general trend of increase in performance with increase in samples was observed in the experiment. Followings images shows that how the Kohnen Self Organizing Map learns the character “H”.

**Figure 4:** Formation of alphabet H in MATLAB environment.
In the above figures results of creating letter H in MATLAB environment is shown in Figure 4, whereas learning of letter H is shown in Figure 5. When Self Organizing Map learns the letter H, the nodes of the grid matrix come closer at the points where letter H is formed.

4. ANALYSIS
This paper presents our research efforts to develop efficient alphabet learning algorithm based on Kohonen’s self-organizing feature map (SOM) algorithm. In the present work, the alphabets are created in MATLAB environment, then their learning is performed using Kohonen Self Organizing map. In learning process nodes of grid matrix come closer at the points where letter is formed and there is more space between the nodes of grid matrix where the alphabet is not formed. Above results are shown for letter H. By using this technique one can learn any alphabet.

5. CONCLUSION
This work introduce a really easy however effective methodology based on Self Organizing Map to deal with the matter of character learning. The result of this work showed better recognition performances in terms of accuracy and speed compared with existing work in literature. The idea of learning characters using Self Organizing Map neural network has proven its efficiency in terms of robustness. Future work can explore the necessity to integrate into an optimization algorithm to further enhance the performance of the system.

REFERENCES:


AUTHORS

Amit Goyal received the B.Tech and M.Tech degrees in Electronics and Communication Engineering from Punjab Technical University in 2002 and 2010, respectively. During 2002-2010, he worked as Asst. Prof. in Electronics and Communication Department and taught various subjects like Neural Networks & Fuzzy Logic, Wireless & Mobile Communication and Microprocessor & Interfacing etc. He is now Associate Prof. in Maharishi Markandeshwar University, Sadopur, Ambala, Haryana.

Ankita Lakhanpal received the B.Tech degree in Electronics and Communication Engineering Engineering from Punjab Technical University in 2011. She is now doing M.Tech from Maharishi Markandeshwar University, Sadopur, Ambala, Haryana, under the guidance of Mr. Amit Goyal.

Shaveta Goyal received the B.Tech and M.Tech degrees in Electronics and Communication Engineering from Punjab Technical University in 2002 and 2011, respectively. During 2002-2011, she worked as Asst. Prof. in Electronics and Communication Department and taught various subjects like Neural Networks, Digital Signal Processing, Digital Circuit & Logic Design and Multimedia Communication etc. She is now Asstt. Prof. in RIMT-IET, Mandi Gobindgarh, Punjab.