Analysis, Simulation and Optimization of LVQ Neural Network Algorithm and Comparison with SOM

Saeed Talati[1*], MohammadReza Hassani Ahangar[2]
1- Department of Electronic Warfare Engineering, Shahid Sattari University of aeronautical Science and Technology, Tehran, Iran.
Email: saeed.talati@yahoo.com (Corresponding author)
2- Department of Computer Engineering, Imam Hossein University, Tehran, Iran.
Email: mrhasani@ihu.ac.ir

Received: August 2019 Revised: October 2019 Accepted: December 2019

ABSTRACT:
The neural network learning vector quantization can be understood as a special case of an artificial neural network, more precisely, a learning-based approach - winner takes all. In this paper, we investigate this algorithm and find that this algorithm is a supervised version of the vector quantization algorithm, which should check which input belongs to the class (to update) and improve it according to the distance and class in question. To give. A common problem with other neural network algorithms is the speed vector learning algorithm, which has twice the speed of synchronous updating, which performs better where we need fast enough. The simulation results show the same problem and it is shown that in MATLAB software the learning vector quantization simulation speed is higher than the self-organized neural network.

KEYWORDS: Neural Network, Learning Vector Quantization, Self-Organizing Neural Network, Optimization.

1. INTRODUCTION

The human brain, according to many scientists, is a more complex system that has been observed and studied throughout the universe. But this most sophisticated system has neither the size of a galaxy nor the number of components that it has over today's supercomputers. The mysterious complexity of this unique system is due to the many connections that exist between its elements. Something that distinguishes the human brain from all other systems.

The self-conscious and subconscious processes that occur around the geographical reach of the human body are all under the control of the brain. Some of these processes are so complex that no computer or supercomputer in the world can process them. However, research shows that the human brain's manufacturing units are about a million times slower than transistors used in CPU silicon chips.

The very high speed and processing power of the human brain goes back to the very large interconnections that exist between the brain-forming cells, and basically, without these communication links, the human brain is reduced to a normal system as well, which is certainly the case. Will not have the current.

After all, the brain's excellent performance in solving a variety of problems and its high performance, simulating the brain and its capabilities, has made it the most important goal of hardware and software architects. In fact, if there is a day (which is apparently not too far away) that we can build a computer of the size and size of the human brain, there will surely be a major revolution in science, industry, and of course, human life.

In order to simulate the computational behavior of the human brain, since the last few decades that computers have enabled computational algorithms to be implemented, research has been started by computer scientists, engineers and mathematicians, whose work we A branch of artificial intelligence, and under the branch of computational intelligence, is classified as "artificial neural networks". In the field of artificial neural networks, numerous mathematical and software models have been proposed, inspired by the human brain, that are used to solve a wide range of scientific, engineering and practical problems in different fields.

2. THE LEARNING VECTOR QUANTIZATION NEURAL NETWORK ALGORITHM

The learning vector quantization neural network can be understood as a special case of an artificial neural network; more precisely, it applies a learning-based approach - *winner takes all*-. The neural networks consist of a series of layers consisting of simple components called neurons that act in parallel. One of the applications is the data locking mechanism. Of the three types used for a fasting machine, the quantization vector is the learning vector [11 - 12]. In fact, the learning vector maker of the idea of self-organized

neural network has created a cohesive brain. My neural network is organized based on a competition that has the capabilities of data folders. In this case, neurons are the fastest-growing types of data available. This is the weight of communication. In this method, the basic neurons are topology and, depending on the type, have different ontologies. Self-organizing neural networks can separate the data structure that you do not have. In the neural network itself, the conceptual framework of learning is unobserved. There are two neurons in your neural network: three inputs and three competitors. [4] Includes neurons whose inputs compete fully with the neurons that make up the full connectivity.

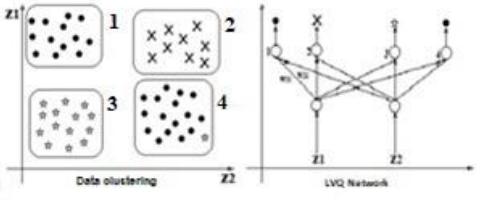

Fig. 1. The output of the learning vector quantization algorithm for the data classification problem.

As can be seen in Fig. 1, each number may be 3, but if 3 neurons cannot compute the data for a given competitor. So in many cases it is necessary for the number of neurons to be greater than the number of vertices. If the number of neurons is too high, the likelihood of a reconciliation will be reduced, which will be due to the performance of the locking states. The neural network algorithm is a vector-based learning algorithm and a competitive algorithm, and in practice it will be seen that this algorithm is also a learning vector compressor. If we want to study the neural network of the learning vector quantization, we must first examine the vector quantization of the learner.

3. VECTOR QUANTIZATION

According to Figure 2, we have a set of inputs named X, which are infinite and, according to the learning vector algorithm, are transformed into a finite set of inputs; Be it.

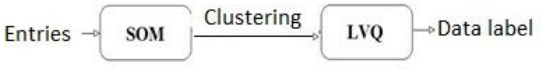

Fig. 2. The learning vector algorithm, x input and i algorithm output

In the figure above, x is an input vector that can have an infinite amount of data in any vector space. i is an index that has a discrete and finite number of quantities. i is associated with a vector such as mi defined on x: i = {1,2,… k} The vector x becomes i under the vector quantization. Collecting data.

4. VECTOR QUANTIZATION BASED ON ENCODER AND DECODER ALGORITHM

To investigate the vector quantization based on the encoder and decoder algorithm we have as follows:

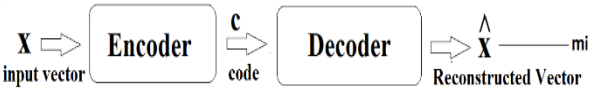

Fig. 3. Vector quantization based on encoder and decoder algorithm.

In the above figure, x is an input vector that has an infinite amount of data in any vector space. In the encoder algorithm x is converted to c, C is a code, and in the encoder algorithm c is converted to x where a discrete enumerable value is actually a vector reconstructed from the initial value of x, and the vector is equivalent to mi; ¬ Improvement is also seen because unlimited vector has become restricted vector. But the problem that actually gets to us is the amount of error we call the quantization error and its value is:

$$e_{(x)} = \|x - \hat{x}\|^2 \qquad (1)$$

In this respect, e(x) is the quantization error value, x is the input vector (infinite value) and x is the vector reconstructed from the input x which has a finite value. The reason for the power of 2 is to make the small errors smaller (ignored) and the larger errors intensified.

$$E = \in \{\|x - \hat{x}\|^2\} = \int \|x - \hat{x}\|^2 . P(x) . d(x) \qquad (2)$$

In this respect E is the value of the quantization error equal to the mathematical expectation of subtracting the magnitude of the input value of the reconstructed output to power 2, which is equal to the product of the sum of the integral subtraction of the input value of the reconstructed output to power 2 in the distribution function. Probability x, in the differential x.

The point to note is that the smaller the value of E (the closer to zero) in the encoder and decoder algorithm, the better the performance.

5. DESIGN AND SIMULATION OF VECTOR QUANTIZATION AND ITS CENTERS

Design and simulation of vector quantization and its centers, the main purpose of the vector quantization algorithm is to convert a series of unlimited members to finite members using the approximation mi. We see the approximation as follows:

Suppose we have three centers with the names (m1, m2, m3) and the value of x is d1 from the center of m1

and d2 from the center of m2 and d3 from the center of m3; Since the value of d1 is smaller than the values of d2, d3, then x will belong to the center of m1 (the winner will win everything according to the above rule).

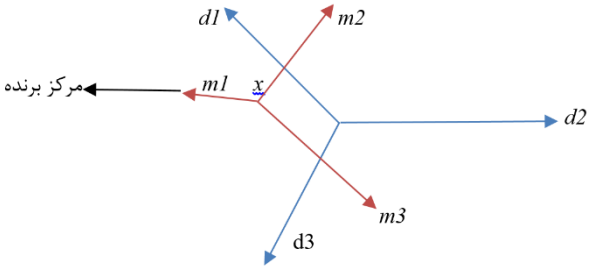

Fig. 4. Centers in Vector Quantization Algorithm and how to win a center.

The above figure shows how to win in theory but how to win mathematically by the following formula:

$$c = \arg\min\|x - mi\| \quad (3)$$

In this respect, c is the winning center, which is equal to the argument of the distance x to the center with the least distance (that center will win).But to see how center boundaries are separated in MATLAB software, the following command is used:

```
X=[0 0
  2 0
  1 2
  0.8 1
  1 -0.5];
voronoi(X(:,1),X(:,2));
xlim([-2 4]);
ylim([-2 4]);
grid on;
axis square;
```

By executing the above command, we will have the following form:

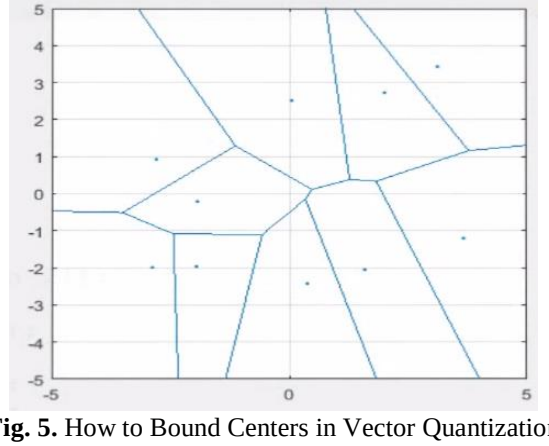

Fig. 5. How to Bound Centers in Vector Quantization Algorithm using MATLAB Software.

6. OPTIMIZATION VECTOR QUANTIZATION

To optimize the vector quantization algorithm we assume that M = {m1, m2, m3, …… mk} are the centers used to classify and cluster these centers, and X = {x1, x2, x3, ……. xt} are temporal data inputs, given the quantization error as:

$$E = \int \|x - \mathrm{m_c}\|^2 . P(x) . d(x) \quad (4)$$

Where, *mc* is the winning center If we define the winner as follows we can claim that the optimization is done in this algorithm:

$$c = \arg\min\|x(\mathrm{t}) - mi\|^2 \quad (5)$$

7. VECTOR QUANTIZATION UPDATE

To update the vector quantization algorithm at time (t + 1) we have:

$$\mathrm{m_{i(t+1)}} = \mathrm{m_{i(t)}} + \Delta \mathrm{m_{i(t)}} \qquad i = 1,2,\dots,k \quad (6)$$

In this respect mi(t + 1) is the updated value and Δmi(t) is the change value of mi(t + 1) relative to mi(t). Which can have the following values in different states:

$$\Delta \mathrm{m_{i(t)}} = \begin{cases} \alpha(\mathrm{t})\left[x(t) - \mathrm{m_{i(t)}}\right], & i = c \arg\min\|x(\mathrm{t}) - mj\|^2 \\ 0 & OtherWise \quad i \neq c \end{cases} \quad (7)$$

The first case is in the state where i is the winning center and the second case is in the case where i is not the winner. Assuming we have two centers m1, m2 where x is d1 from m1 and d2 from m2 so that d1 is <d2 (as in Fig 6) then we have mi(t + 1):

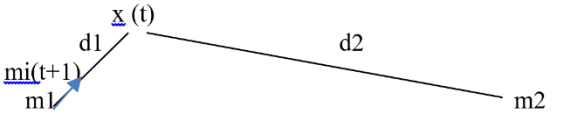

Fig. 6. How to update vector quantization.

In this form m1 is the winning center and the center is updated and mi(t + 1) moves to x(t).
So the updated mode mi(t + 1) is as follows:

$$\mathrm{m_{1(t+1)}} = \mathrm{m_{(t)}} + \alpha_{(\mathrm{t})}\left[x(t) - \mathrm{m_{1(t)}}\right] \quad (8)$$

Where α (t) is the learning rate; and for the general case we have:

$$\mathrm{m_{1(t+1)}} = \mathrm{m_{i(t)}} + \delta \mathrm{c_i}\ \alpha(\mathrm{t})\left[x(t) - \mathrm{m_{1(t)}}\right] \quad (9)$$

In this respect, the existence of δc_i is because only the winning parameter is updated and its values are different for different states with:

$$\delta_{ci=}\begin{cases}1 & i=c\\ OtherWise & i\neq c\end{cases} \quad (10)$$

Now that we have fully explored the vector quantization algorithm, let's look at the learning vector quantization algorithm

8. LEARNING VECTOR QUANTIZATION

This algorithm is a supervised version of the vector quantization algorithm. Where we have to see which x belongs to the class (to update) and we need to improve it according to the distance and class we want. The learning vector quantization algorithm is updated like the nearest class winner or vector quantization. In this algorithm, two situations may occur:

The first case according to Fig.7 teaches m1 correctly when x(t), m1 belong to a class (we create the classification but which class x(t) belongs to is the nature of the problem; here m1 belongs to Class A and m2 belong to Class B).

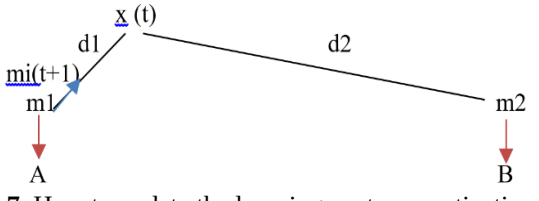

Fig. 7. How to update the learning vector quantization algorithm in the first step.

Secondly, as shown in Fig8, when x(t), m1 does not belong to a class and m1 is not properly trained (here m1 belongs to class B and m2 belongs to class A). m1 Since it is a winner but not in a class that corresponds to x(t), we have to move it far enough so that the class m2 is closer and thus wins.

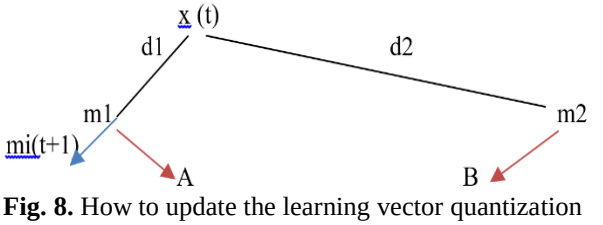

Fig. 8. How to update the learning vector quantization algorithm in the first step.

9. OPTIMIZATION OF LEARNING VECTOR QUANTIZATION ALGORITHM

In the supervised learning vector quantization algorithm, we have:

$$m_{i(t+1)} = m_{i(t)} + \Delta m_i(t) \quad (11)$$

In this respect Δm_i (t) can have the following values:

$$\Delta m_i(t) = \begin{cases}+\alpha(t)\left[x(t)-m_{i(t)}\right] & i=c\\ -\alpha(t)\left[x(t)-m_{i(t)}\right] & i=c\\ 0 & i\neq c\end{cases} \quad (12)$$

The first is when mc and x(t) are in a class; this is an approximation. The second case is when mc and x(t) are not in a class; this is a bypass state.The third case is when the center is not the winning center.

However, it must be: 0<α(t)<1 so that this method converges to the boundaries of the graphs. However, the references suggest: α (t) <0.1. Of course, the appropriate choice α(t) is proved; the LVQ relation has an optimal state as follows:

$$\Delta m_{i(t+1)} = \delta c_i\ f_i(t)\alpha(t)\left[x(t)-m_{i(t)}\right] \quad (13)$$

And in general we will have:

$$m_{i(t+1)} = m_{i(t)} + \delta c_i\ f_i(t)\alpha(t)\left[x(t)-m_{i(t)}\right] \quad (14)$$

And fi is equal to:

$$f_i(t) = \begin{cases}+1\\ -1\end{cases} \quad (15)$$

The first case is when mi(t) and x(t) are in a class; this mode moves to x(t). The second state corresponds to states other than the former; this leads to the departure of x(t).

10. LEARNING VECTOR QUANTIZATION ALGORITHM AND COMPARING IT WITH SELF-ORGANIZING ALGORITHM

In the learning vector quantization algorithm 1 we have:

$$m_{i(t+1)} = m_{i(t)} + \delta c_i\ f_i(t)\alpha(t)\left[x(t)-m_{i(t)}\right] \quad (16)$$

And we have it organized:

$$m_{i(t+1)} = m_{i(t)} + h_i(t)\alpha(t)\left[x(t)-m_{i(t)}\right] \quad (17)$$

Both algorithms are proved by equating the neighborhood functions (δci.fi(t) in LVQ and hi(t) in SOM) that's mean δci.fi(t) = hi(t).

11. COMPARE THE LEARNING VECTOR QUANTIZATION ALGORITHM AND COMPARE IT WITH THE SELF-ORGANIZING ALGORITHM

In the learning vector quantization algorithm compared to the self-organizing algorithm:

1- We don't have a neighborhood structure.

2- There is practically only one neuron moving and we do not interact with neighbors.

3. There is no specific connection between the neighbors.

4- The order does not matter.

12. VECTOR LEARNING ALGORITHM QUANTIZATION OPTIMIZER

We have the learning vector quantization optimized:

$$m_{i(t+1)} = m_{i(t)} + s_{i(t)}\alpha(t)\left[x(t) - m_{(t)}\right] \quad (18)$$

Where, si(t) is the product of δci.fi and the value of si(t) is:

$$s_{i(t)} = \begin{cases} +1 & i = c \\ -1 & i = c \\ 0 & OtherWise \end{cases} \quad (19)$$

The first one is about correct classification and the second one is about incorrect classification. And by converting α(t) to αi(t) for each center an independent learning rate is generated and can change over time.
It should be examined how α(t) changes to optimize the performance of the method. We call mi(t + 1) in mi and x:

$$m_{i(t+1)} = m_i(t) + s_i(t).\alpha(t)\left[x(t) - m_{i(t)}\right] = [1 - s_i(t).\alpha_i(t)]m_{i(t)} + s_i(t).\alpha_i(t)x(t) \quad (20)$$

Where, [1-si(t).α(t)] is the weight of x(t-1) in mi(t) and αi(t) is the weight of x(t). The weight x(t) = αi(t) and the weight x(t-1) = [1-si(t).αi(t)] αi(t-1). If we have: x(t) = x(t-1) We have the principle of induction: x(t) = x(t-1) = x(t-2) …… = x(tk) and we have equality by writing : αi (t) [1-si(t) .αi(t)] αi(t-1) Value αi(t) = αi(t-1) / 1+ Si(t) αi (t-1), and 0 <αi(t) <1.

We must be careful that αi(t) is not greater than 1 because it causes system instability. Based on this equation, changes in αi are also calculated (learn more for negative αs. In references, αi(0) is usually selected from 0.3 to 0.5.

Given the size αi(t) and since all samples have the same weight and impact, the value of αi(t) in different states is equal to:

$$s_i(t) = \begin{cases} 0 & \rightarrow \alpha_{i(t)} = \alpha_{i(t-1)} \\ 1 & \rightarrow \alpha_{i(t)} = \frac{\alpha_{i(t-1)}}{1+\alpha_{i(t-1)}} \\ -1 & \rightarrow \alpha_{i(t)} = \frac{\alpha_{i(t-1)}}{1-\alpha_{i(t-1)}} \end{cases} \quad (21)$$

In the first case, since Si is zero, the value of αi(t) is equal to αi(t-1). The latter reduces αi (because it does not have to converge rapidly and bias the samples) and increases the accuracy of the changes.

In the third case, it increases the αi (because it is completely inaccurate and goes farther away) and increases the rate of change.

13. SIMULATION OF LEARNING VECTOR QUANTIZATION ALGORITHM

The simulation results of the learning vector quantization algorithm are presented in MATLAB software and the following figure shows the simulation results which prove that the simulation of the learning vector quantization method is twice faster than the self-organizing method, but it is also very simple to prove mathematically why Unlike the self-organizing method in the learning vector quantization method, the updating operation is performed twice at the same time, which makes it twice as fast.

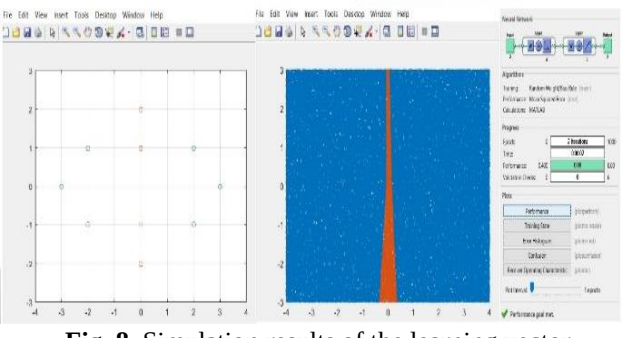

Fig. 8. Simulation results of the learning vector quantization optimization algorithm.

14. CONCLUSION

The neural network of the learning vector quantization can be understood as a special case of an artificial neural network, more precisely, a learning-based approach - it takes everything. In this paper, we investigate this algorithm and find that it is a supervised version of the vector quantization algorithm, which should check which input belongs to which class (to update) and given the distance and class of the case. Comment, let's improve it. In the vector algorithm the learning vector is updated like the nearest vector class or the winner. This algorithm is shown to be faster than comparing the neural network of the learning vector quantization with the self-organized neural network. The simulation results also show the same problem and it is demonstrated that in MATLAB software the learning vector quantization simulator speed is more organized than the neural network itself.

REFERENCES

[1] K. Hesampour, **"Intelligent Selection of Optimum Properties and Separators for Automatic Signal Modulation Detection",** *Yazd University*, 2011.

[2] S. Moghaddam, Goodarz; A. Naseri and S. H. Asadollahi, **"Intelligent Algorithm for Identifying Radar Signals Using Matrix Multiplication and RBF Neural Network"**, *13th Iranian Student*

Electrical Engineering Conference, Tarbiat Modares University, Tehran, 2010.

[3] F. Abdolhossein, Sh. Shafiei**, "An Approach to Machine Learning Algorithms for Artificial Neural Network, MLP Neural Network, RBF Neural Network"**, *Third National Conference on Electrical and Computer Engineering Technology, Tehran, Payam Noor University and Payam Noor University of Tehran*, 2016

[4] Gh. Mohammad Reza, M. Dadgar, **"LVQ Presentation of a Method for Algorithmic Classification of Data",** *2nd Iranian National Congress on New Technologies for Sustainable Development, Tehran, Mehr Arvand Institute of Sustainable Developmen*t, 2014.

[5] J. A. Anderson, M. T. Gately, P. A. Penz, and D. R. Collins**, "Radar signal categorization using a neural network,"** *Proceedings of the IEEE*, Vol. 78, No. 10, pp. 1646–1657, 1990.

[6] A. M. Aziz, **"A Novel and Efficient Approach for Automatic Classification of Radar Emitter Signals,"** *in 2013 IEEE Aerospace Conference*, pp. 1–8, 2013.

[7] B. E. Boser, I. M. Guyon, and V. N. Vapnik**, "A Training Algorithm for Optimal Margin Classifiers,"** *in Proceedings of the fifth annual workshop on Computational learning theory* - COLT '92, pp. 144–152, 1992.

[8] M. Cherniakov, R. S. A. R. Abdullah, P. Jancovic, M. Salous, and V. Chapursky, **"Automatic Ground Target Classification using Forward Scattering Radar,"** *Radar, Sonar and Navigation, IEE Proceedings*, Vol. 153, No. 5, pp. 427–437, Oct. 2006.

[9] C. L. Davies and P. Hollands, **"Automatic Processing for ESM,"** *IEE Proceedings F Communications, Radar and Signal Processing*, Vol. 129, No. 3, p. 164, Jun. 1982.

[10] J. Dudczyk, A. Kawalec, and J. Cyrek**, "Applying the Distance and Similarity Functions to Radar Signals Identification,"** in *2008 International Radar Symposium*, pp. 1–4, 2008.

[11] P. M. Grant and J. H. Collins, **"Introduction to electronic warfare,"** *IEE Proceedings F Communications, Radar and Signal Processing*, Vol. 129, No. 3, pp. 113, Jun. 1982.

[12] J. Han, M. Kamber, and J. Pei, **"Data Mining Concepts and Techniques",** *Third Edit. Morgan Kaufmann Publisher*, pp. 740, 2012.

[13] E.J. Hartman, J.D. Keeler, J.M Kowalski, **"Layered Neural Networks with Gaussian Hidden Units as Universal Approximations,"** *Neural computation*, Vol. 2, No. 2, pp. 210-215, 1990.

[14] J. Liu, J. Lee, L. Li, Z.Q. Luo, and K.M. Wong, **"On-line Clustering Algorithms for Radar Emitter Classification,"** *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 8, pp. 1185–1196, Aug. 2005.

[15] K. Pearson, **"On Lines and Planes of Closest Fit to Systems of Points in Space,"** *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Scienc*e, Vol. 2, No. 11, pp. 559-572, 1991.

[16] M.-Q. Ren, Y. Zhu, Y. Mao, and J. Han**, "Radar Emitter Signals Classification Using Kernel Principle Component Analysis and Fuzzy Support Vector Machines,"** *in 2007 International Conference on Wavelet Analysis and Pattern Recognition*, Vol. 3, pp. 1442–1446, 2007.

[17] E. Świercz, **"Automatic Classification of LFM Signals for Radar Emitter Recognition Using Wavelet Decomposition and LVQ Classifier,"** VVol. 119, No. 4, pp. 488–494, 2011.

[18] V. Vapnik, **"The Nature of Statistical Learning Theory. Springer Science & Business Media",** pp. 314, 2000.

[19] R. G. Wiley, **"Electronic Intelligence: The Analysis of Radar Signals".** Artech House, p. 337, 1993.

[20] Bhattacharya, Gautam, Koushik Ghosh, and Ananda S. Chowdhury. **"An Affinity-Based New Local Distance Function and Similarity Measure for kNN Algorithm."** *Pattern Recognition Letters* 33.3, pp. 356-363, 2012.