BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **22**, No 2
Sofia • 2022 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2022-0015

Enhancing the Speed of the Learning Vector Quantization (LVQ) Algorithm by Adding Partial Distance Computation

Orieb AbuAlghanam[1], *Omar Adwan*[2,4], *Mohammad A. Al Shariah*[3], *Mohammad Qatawneh*[4]

[1]*Department of Networks and Information Security, Faculty of Information Technology, Al-Ahliyya Amman University, Amman, Jordan*
[2]*Department of Computer Science, Faculty of Information Technology, Al-Ahliyya Amman University, Amman, Jordan*
[3]*Department of Data Science and Artificial Intelligence, Al-Ahliyya Amman University, Amman, Jordan*
[4]*Department of Computer Science, King Abdulla II School for Information Technology, University of Jordan, Amman, Jordan*
E-mails: o.abualghanam@ammanu.edu.jo adwanoy@ammanu.edu.jo m.sharaiah@ammanu.edu.jo mohd.qat@ju.edu.jo

***Abstract*:** *Learning Vector Quantization* (*LVQ*) *is one of the most widely used classification approaches. LVQ faces a problem as when the size of data grows large it becomes slower. In this paper, a modified version of LVQ, which is called PDLVQ is proposed to accelerate the traditional version. The proposed scheme aims to avoid unnecessary computations by applying an efficient Partial Distance* (*PD*) *computation strategy. Three different benchmark datasets are used in the experiments. The comparisons have been done between LVQ and PDLVQ in terms of runtime and in result, it turns out that PDLVQ shows better efficiency than LVQ. PDLVQ has achieved up to 37% efficiency in runtime compared to LVQ when the dimensions have increased. Also, the enhanced algorithm* (*PDLVQ*) *shows clear enhancement to decrease runtime when the size of dimensions, the number of clusters, or the size of data becomes increased compared with the traditional one which is LVQ.*

***Keywords*:** *Classification, LVQ, partial distance computation, PDLVQ, SOM.*

1. Introduction

In the context of smart cities, smart environments, and smart campus the size of data become huge. Nowadays, data science is currently regarded as one of the hottest research areas. The massive growth of big data has become a problem that must be addressed. Machine learning is one of the most essential methods for gaining value from data [1, 2]. Moreover, it has been used in different fields [3, 4].

Nowadays, data science is considered as one of the trending hot research areas. The huge development of big data becomes an issue that needs to deal with. Machine

Learning (ML) is considered as one important way for making benefit from this data. Moreover, the rapid technological advancements and large data production need to upgrade or change conventional techniques [5]. ML is the study that aims to use the computer algorithm that developed its behaviour over time based on the experience. Moreover, ML is considered to be a component of artificial intelligence that is used to classify or predict future situations [6]. Nowadays, increasing the requirement for classifications is considered one of the most frequent tasks in machine learning. The research in this field has increased significantly. Thus, during the last few years the fast development and increase of the volume of data becomes an important issue [7] so, solution of some problem in a reasonable time with a huge dataset is quite an issue [8, 9].

Classification has been used in many fields such as intrusion detection systems and image processing [10, 11]. Even more in the medical field, especially in image recognition or disease classification. In medical images, we need to deal with millions or billion pixels per picture to recognize or diagnose particular diseases. Thus, a huge number of computational processes are needed at the same time and the run time should be taken into consideration also [12-15].

Learning Vector Quantization (LVQ) classifier is one of the most intuitive prototypes-based classification models and it is an adaptive data classification technique that uses training data with the desired class of information to classify data [12, 16]. LVQ has two layers the first layer is a linear layer with supervised training and the second layer is a competitive layer that pre-processes the dataset and finds cluster centres [17]. Fig.1 shows the architecture of LVQ which uses unsupervised data clustering algorithms to adjust weights only on the basis of input patterns when no knowledge about the desired outcomes is provided. In this case *x* represents the input dimensions and the *y* represent the number of label classes.

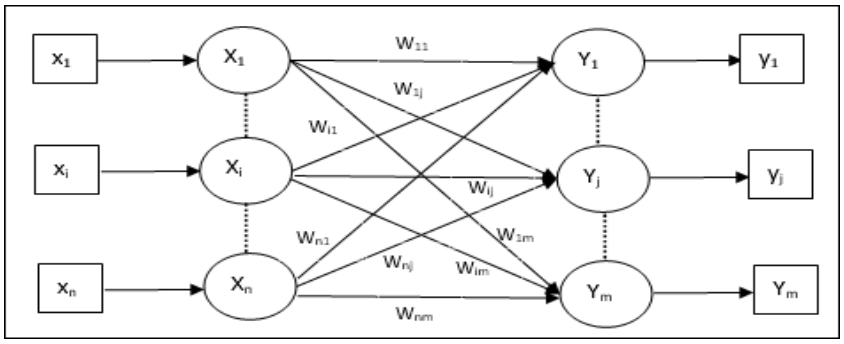

Fig. 1. The architecture of learning vector quantization LVQ [18]

LVQ is considered severely limited by the high computational complexity that makes it extremely slow [19]. This is because of the Euclidian distance that is used in calculations, this part of the algorithm takes more time and more complex computation especially when the dimensions are increased [19].

In this paper, a new modified version of LVQ has been presented, which is called PDLVQ. The modified version aims to decrease the computation time that has been used in traditional LVQ and accelerate the overall runtime. This is done by adding a Partial Distance (PD) and selecting the important computations only.

The rest of the paper is organized as follows. Section 2 introduces the background and some related works. The proposed architecture design and settings are presented in the Section 3. Section 4 presents the results and analysis. Finally, we conclude the paper in the Section 5.

2. Background and related works

A brief background is presented in this section for learning vector quantization and partial distance as presented in the subsections below.

2.1. Learning vector quantization

LVQ is an extension of the Kohonen neural network method [19], which has been moved from an unsupervised neural network in Kohonen Self-Organizing Maps (SOM) to a supervised LVQ neural network. LVQ can be viewed as a neural classifier paradigm that has the same architecture of SOM without topological structure and consists of output layers that are designed for several classification categories [22, 23]. The input vector that has been inserted in LVQ to classify is known as the reference vector or codebook [24].

Nowadays classification is important in the big data world, therefore several studies have developed the traditional techniques to cope with massive development of these kind of data. In [25] a design is presented of the meta classifier ensemble with sampling and feature selection for multiclass imbalanced data has been investigated to improve the ensemble classifier through data-level approach (sampling and feature selection). Moreover, [26] proposes a new variant of Grey Wolf Optimization (GWO), called Inertia Motivated GWO (IMGWO). The aim of IMGWO is to establish better balance between exploration and exploitation, traditionally an Artificial Neural Network (ANN) with backpropagation.

The LVQ network works well for many problems, but it has two drawbacks: a dead neural that never does anything useful and how the initial weight vectors are arranged. The first problem can be solved using a “conscience” mechanism, which makes a neuron that wins a lot of times to have the ability to let others win too.

The second problem is that the LVQ depends on how the initial weight of a vector is distributed [24, 27], which may cause the vector to travel through a region of a class that it does not represent in order to get to the right region which consumes more calculations [27]. Because the weights of such a neuron will be repelled by vectors in the region it has to cross, this might never be accomplished. Thus, it may never properly classify the region to which it is being attracted.

In literature, LVQ has been used in different fields such as security, medical, agriculture and many others [30, 29]. In [31] an Anomaly based Detection Analysis for Intrusion Detection is proposed for big data. The LVQ Algorithm is suggested to process a large volume of data to detect the attacks. Moreover, in [32] a multimodal biometric based on LVQ analyze the images in two kinds of biometrics which are face and fingerprint. Pattern generation and pattern matching is used in their model to enhance the result of classification. The comparison result has been done between

Gaussian Mixture Model (GMM) and the proposed model, namely LVQ and the LVQ shows better performance compared to GMM for different metrics.

The use of LVQ is widely spread in the image-processing field. In [33] a face recognition from the webcam for digital images is proposed. The proposed technique is a hybrid of the LVQ-Algorithm with Self Organizing Kohonen. The result of the hybrid technique performs better results than using LVQ and Kohonen separately. Also, LVQ is used in pattern recognition as [34] an Artificial Neural Network LVQ, and Gray Level Co-occurrence Matrix (GLCM) are used as a method to classify the real batik. The proposed method extracts textural features and obtains the pattern value of batik to recognize the real batik.

LVQ has an important role in the agriculture field; it has been used for the early detection of diseases. In [35] authors present a method for detecting and classifying tomato leaf disease using a Convolutional Neural Network (CNN) model and the LVQ-Algorithm. The proposed method has been applied on 500 images of tomato leaves with four symptoms of diseases. In spite of LVQ limitations, it has been noticed that LVQ is used in many images' classification, image recognition and security fields. Therefore, the purpose of this paper concerns a study of slowness problem in calculating the Euclidean distance [36] that is used in LVQ when using a large number of dimensions.

2.2. Partial distance

PD is one of the most popular strategies, which is used to reduce the computational complexity in distance calculation and many researchers use PD logic to accelerate the algorithm. [37] introduces the effect of ordering the codebook on the efficiency of the partial distance search algorithm for Vector Quantization [38] This study shows that the algorithm's computational complexity can be furthermore reduced by ordering the code vectors according to the sizes of their related clusters. Vector quantization has become very popular in multiple areas but the utilization of the vector quantization is strongly limited by the computational complexity of the encoding process which makes it exponentially expensive for large-size codebooks [39].

To reduce the computational complexity of the minimum distortion search in a vector quantization encoding process, the code vector can be rejected on the basis of partial distance without completing the total distance computation, to maximize the savings offered by the partial distance search by ordering the code vectors according to the sizes of their related clusters. According to the study, the explicit ordering of the code vectors shows more valuable computational savings than the partial distance method, the efficiency has improved by arranging the code vectors in the codebook in a way that the sizes of their related clusters are in decreasing order [37]. The proposed partial strategy accelerates the code search in VQ-based coding technique and it can be adopted to save time cost on best-match codeword in a codebook. As observed in the experimental results the time required in the proposed scheme of encoding an input image is at least twice faster than the speed of a full search.

The proposed schemes in [40, 41] show a new strategy to accelerate the k-Means clustering algorithm in order to utilize the algorithm in clustering and

overcome the high computational complexity. This study suggests accelerating the k-Means clustering algorithm by avoiding many needless distance calculations by utilizing the Partial Distance. PD logic, the PD Algorithm allows early termination of the distance calculation between a data point (vector) and a cluster center by introducing a premature exit condition in the search process. Based on the experimental results the proposed algorithm provides better efficiency when applied to different datasets in all cases [41].

An accelerating k-Prototypes Algorithm based on partial distance is proposed in [42]. The proposed algorithm avoids distance computations in order to find the shortest distance between an object and a cluster without computing distances for all attributes. The experiment results show the proposed k-Prototypes Algorithm outperforms the initial k-Prototypes Algorithm. Moreover, in [43] a neural system that solves the problem of fuzzy clustering of distorted observations is shown.

3. Proposed model

In this paper, a new modified version of LVQ is proposed, which is called PDLVQ. The model being proposed aims to accelerate the LVQ by combining PD computation with LVQ in order to decrease unnecessary calculations that are used in LVQ. Moreover, these changes have been made to reduce the time taken by LVQ to run without affecting its consistency or accuracy of classifications.

PD uses conditions that allow early termination of distance calculations. Moreover, PD computation works efficiently on high dimensional datasets and it reduces the exhaustive search and computational complexity. This is done by allowing early termination of the distance calculation between a data point and a cluster center by providing early exit conditions in the search process [42].

Let $C = c_{ij}$ where $i = 1, ..., N$ be a set of cluster centers of size N, where (c_{ij}, $j = 1, ..., K$) is a K dimensional data point (vector).

For a given data point $X = (x_j, j = 1, ..., K)$, it is required to find the cluster center with the minimum distance from the set C under the squared error distance measure defined as shown in the next equation where D_i represent the dimension of the sample. For example, if we have two vectors each one has 4 dimensions $A(x_1, x_2, x_3, x_4)$ and $B(y_1, y_2, y_3, y_4)$, then the distance between those two vectors will be represented by the Euclidian distance as follows:

$$D_i(A, B) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2 + (x_4 - y_4)^2}. \quad (1)$$

The major challenge is when a vector's dimension becomes large such as ten, sixteen, or even more. Despite the applicability at applying Euclidean distance this will also take a long time for computations. The proposed PDLVQ uses one of the following three cases to eliminate the unnecessary computations:

- **Best case.** This case is used when the calculations of the new data point are greater than the $\boldsymbol{D_i}$; in this case not all dimensions will be calculated. The time complexity is $O(a)$.
- **Average case.** This case is used when the $\boldsymbol{D_i}$ is neither the longest nor the shortest distance, the estimation of the new data point will be the same as the $\boldsymbol{D_i}$. In

this case, half the number of dimensions will be calculated so, the time complexity is $O(n/2)$.

- **Worst case.** When the $\boldsymbol{D_i}$ has been calculated, this takes the longest distance, which means that $\boldsymbol{D_i}$ will take all dimensions into consideration for new data point estimation and the time complexity is $O(n)$.

Numerical example. This section presents a difference between LVQ and PDLVQ using equations.

Suppose the dataset has three classes which are *X*, *Y* and *Z*,

$$X= \{a_1, a_2, a_3, a_4, a_5,\ldots,a_n\},$$
$$Y= \{b_1, b_2, b_3, b_4, b_5,\ldots,b_n\},$$
$$Z= \{c_1, c_2, c_3, c_4, c_5,\ldots,c_n\}.$$

If any record has been entered into the LVQ model the record will be determined using Euclidean distance and compared with each class as shown in the next equations (2)-(4). On the other hand the competition cost depends on the number of classes and the dimension of the input. Let assume that $A= \{z_1, z_2, z_3, z_4, z_5,\ldots, z_n\}$ is an input record need to be classified. Using LVQ the result will be calculated as follows:

(2) $D_i = \sqrt{(z_1 - a_1)^2 + (z_2 - a_2)^2 + (z_3 - a_3)^2 + (z_4 - a_4)^2 + \ldots\ldots + (z_n - a_n)^2}$,

(3) $D_i = \sqrt{(z_1 - b_1)^2 + (z_2 - b_2)^2 + (z_3 - b_3)^2 + (z_4 - b_4)^2 + \ldots\ldots + (z_n - b_n)^2}$,

(4) $D_i = \sqrt{(z_1 - c_1)^2 + (z_2 - c_2)^2 + (z_3 - c_3)^2 + (z_4 - c_4)^2 + \ldots\ldots + (z_n - c_n)^2}$.

However, in the PDLVQ many unnecessary computations will be eliminated based on the heuristic function that tunes the parameter α. PDLVQ reduces the number of points to be computed in each vector or input based on the assumption of α. If α is assumed to be less than D_i then PDLVQ will check the value in each point so if it finds that $(z_1 - a_1)^2 + (z_2 - a_2)^2 + (z_3 - a_3)^2 > \alpha$ then $\ldots + (z_n - a_n)^2$ will be ignored and so on. In this case, the dimension of the vector using Euclidean distance will be reduced and this will reduce the computational cost and the run time.

As shown in the next equations, the calculations depend on the value of α. It can be noticed that in each formula the number of points is changed to a lesser number of points:

(5) $$D_i = \sqrt{(z_1 - a_1)^2 + (z_2 - a_2)^2 + (z_3 - a_3)^2},$$

(6) $$D_i = \sqrt{(z_1 - b_1)^2 + (z_2 - b_2)^2 + (z_3 - b_3)^2 + (z_4 - b_4)^2},$$

(7) $$D_i = \sqrt{(z_1 - c_1)^2 + (z_2 - c_2)^2}.$$

As shown in the flowchart of PDLVQ in Fig. 2, the model being proposed has three major parts, where data are being preprocessed, tuning for D_i and PDLVQ mode, which is a modified version of LVQ. Irrelevant and redundant data, as well as noisy and unreliable data, should be removed during the data-preprocessing step because of their effect on the accuracy of the model being built. Also, in this phase, cleaning, normalization, transformation, function extraction, and selection should be carried out for the examples of data preprocessing being considered.

Decreasing the computational complexity depends on selecting the best value of D_i, which helps to reach the first case or at least the second one. This is done based on the preprocessing phase after a general look for the classes and the dimensions for the entered dataset. Inside the classification model, which is PDLVQ the value of D_i

will be updated automatically and the learning rate will be enhanced based on the dataset records.

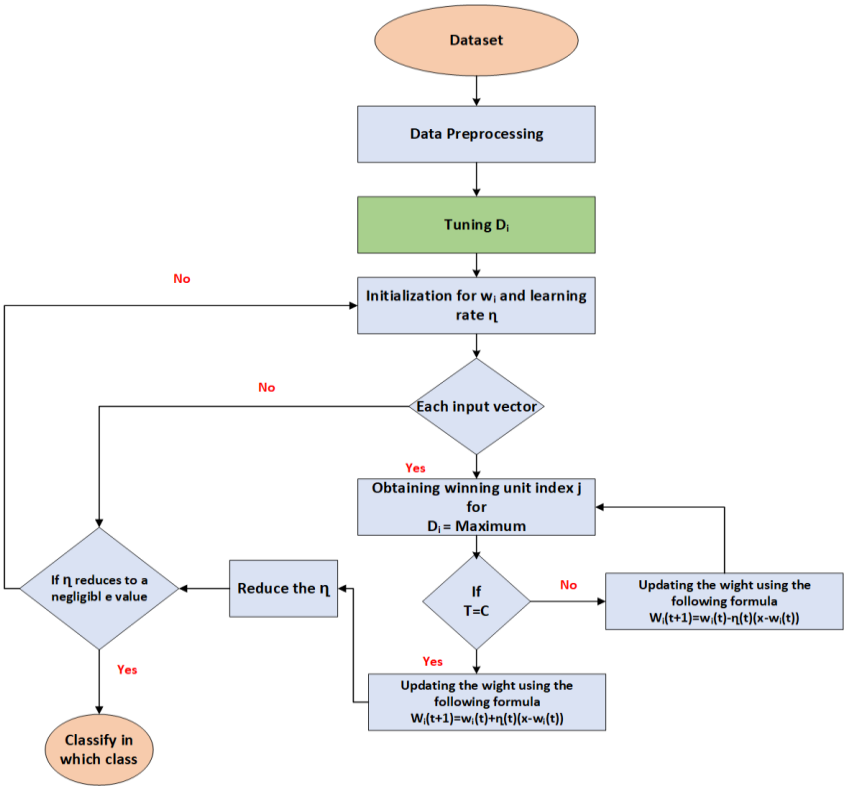

Fig. 2. Flowchart for PDLVQ Algorithm

4. Experiments and results

4.1. Dataset

In this paper, three benchmark datasets have been collected from UCI Repository of Machine Learning Databases [44] which are Iris, Ecoli and Pima. Table 1 shows the number of classes and dimensions for each one. The number of classes refers to how many classes can the data be classified to. In Iris three classes exist which are Iris setosa, Iris virginica and Iris versicolor. Moreover, the number of dimensions represents the number of features which are in Iris Sepal length, Sepal width, Petal, length and Petal Width. The size of data represents the number of records or the number of samples it contains.

Table 1. The characteristics of the three benchmark datasets

Dataset	Number of classes	Number of dimension	Size
Iris	3	4	150
Ecoli	8	7	336
Pima	2	8	768

4.2. Analysis results

We empirically confirm the efficiency of the modified version PDLVQ compared to LVQ. The experiments have been carried out in terms of runtime for each dataset based on different scenarios. Table 2 shows the runtime of LVQ and PD LVQ for the original datasets without any changes in the number of clusters, dimensions and records. The results show that the efficiency of PDLVQ is better and it takes less runtime for the three datasets compared to LVQ. It can be noticed that the efficiency of PDLVQ is enhanced when the size of the dataset increases.

Table 2. Runtime for LVQ vs PDLVQ for original datasets

Dataset			LVQ	PDVQ	Efficiency
Iris	C	3	0.17	0.11	0.05
	D	4			
Ecoli	C	8	0.33	0.22	0.11
	D	7			
Pima	C	2	0.37	0.23	0.14
	D	8			

The experiments have been carried out using two separate scenarios: the first, in which we increased the number of clusters for each dataset and determine the runtime in each case, after that a comparison has been done to calculate the efficiency of PDLVQ compared to LVQ. In the second scenario, we increased the dimensions in each dataset to determine the runtime that is needed in the modified version (PDLVQ) and LVQ. Accordingly, a comparison has been done to determine the efficiency between PDLVQ and LVQ.

The evaluation of the proposed model has different scenarios – for example the dimensions have been expanded using feature extraction to show the performance of our model, how it works when the number of features increase. Moreover, the number of classes are expanded to evaluate the performance of the proposed mode. In Iris dataset three original clusters are existing and in order to show the efficiency of the proposed model when the number of clusters increases, we split the data into several clusters.

Table 3 shows the runtime of LVQ and PD LVQ for different scenarios. It can be noticed that PDLVQ shows better efficiency compared with LVQ, when the number of clusters and dimensions of the datasets is increased. The efficiency of PDLVQ in Iris reached 24% when the number of clusters has been changed to 24 clusters and reached 32% when the dimensions for Iris have been expanded to be 16 dimensions.

PDLVQ shows better performance when the number of clusters, dimensions and the size of data is increasing. This has been verified when Ecoli dataset has been chosen, where the results show that PDLVQ needs less runtime compared to LVQ. Moreover, it can be noticed that when the number of clusters and dimensions of the datasets increase or when the size of the database is increased the enhancement for PDLVQ gives better efficiency.

PDLVQ achieved in Ecoli 30% and 35% efficiency compared with LVQ when the number of clusters changed to be 32 and the dimensions have been increased to be 28. On the other hand, the enhancement in Pima dataset is greater than in Ecoli

and Iris where the achieved efficiency is 35% and 37% when the number of clusters has been increased to 32 and the dimensions are 34. The higher efficiency has been reached in Pima when the dimensions have been changed to 34 and the size of the data became larger than the previous ones.

Table 3. The experimental results for different datasets at different number of clusters and dimensions

Iris					
Number of clusters	3	6	9	12	24
Runtime LVQ	0.17	0.33	0.46	0.57	0.81
Runtime PDLVQ	0.11	0.22	0.30	0.39	0.57
Efficiency	0.05	0.11	0.16	0.18	0.24
Dimension	2	4	6	8	16
Runtime LVQ	0.13	0.27	0.38	0.56	0.89
Runtime PDLVQ	0.07	0.15	0.21	0.35	0.57
Efficiency	0.06	0.12	0.17	0.21	0.32
Ecoli					
Number of clusters	2	4	8	16	32
Runtime LVQ	0.23	0.35	0.47	0.75	0.98
Runtime PDLVQ	0.14	0.21	0.33	0.54	0.68
Efficiency	0.09	0.14	0.16	0.21	0.30
Dimension	3	7	14	21	28
Runtime LVQ	0.16	0.31	0.45	0.67	1.12
Runtime PDLVQ	0.08	0.16	0.26	0.43	0.77
Efficiency	0.08	0.15	0.19	0.24	0.35
Pima					
Number of clusters	2	4	8	16	32
Runtime LVQ	0.25	0.41	0.57	0.84	1.07
Runtime PDLVQ	0.14	0.24	0.37	0.60	0.72
Efficiency	0.11	0.17	0.20	0.24	0.35
Dimension	4	8	16	24	34
Runtime LVQ	0.36	0.43	0.58	0.91	1.24
Runtime PDLVQ	0.24	0.25	0.37	0.65	0.87
Efficiency	0.12	0.18	0.21	0.26	0.37

It can be noticed that the PDLVQ achieves better efficiency using the same dataset when the number of clusters has been increased. Moreover, when the number of dimensions has been increased the runtime that is needed in PDLVQ is always less than in the traditional version of LVQ. Thus, Our PDLVQ algorithm decreases the time complexity by reducing unnecessary distance computation by using partial distance computation instead of distance computations of all attributes between an object and a cluster center.

Fig. 3 and Fig. 4 show the runtime for LVQ and PDLVQ in Iris when the number of clusters and the dimensions has been increased. Also, Fig. 5 and Fig. 6 show the runtime when Ecoli has been chosen to show the runtime for the LVQ and the

modified version which is PDLVQ when the number of cluster and dimensions have been increased. Finally, Fig. 7 and Fig. 8 show the runtime for LVQ and PDLVQ in Pima dataset.

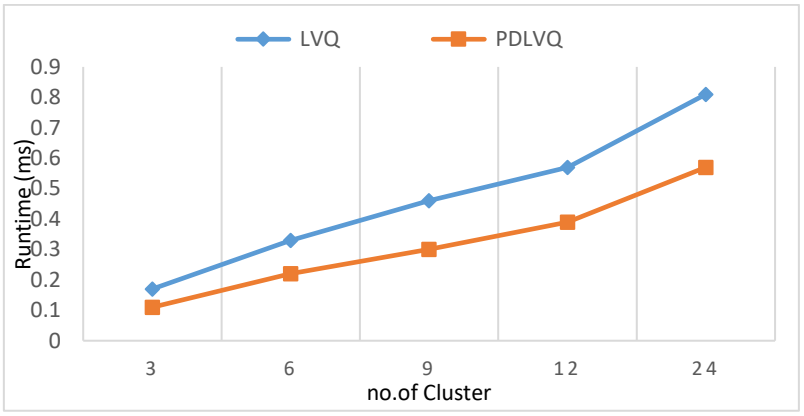

Fig. 3. Runtime for Iris dataset in different number of clusters

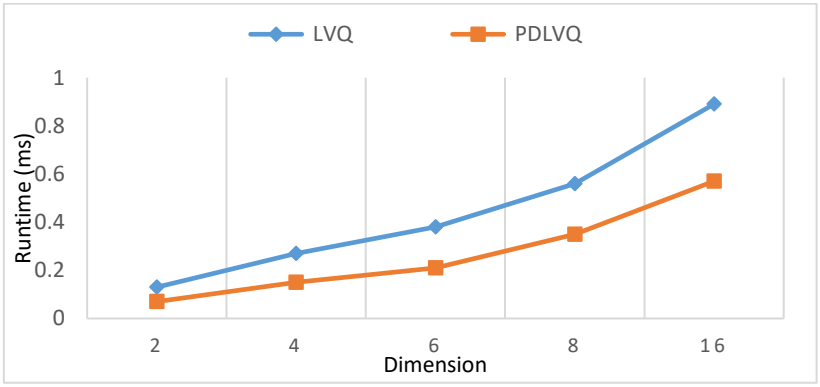

Fig. 4. Runtime for Iris dataset in different dimensions

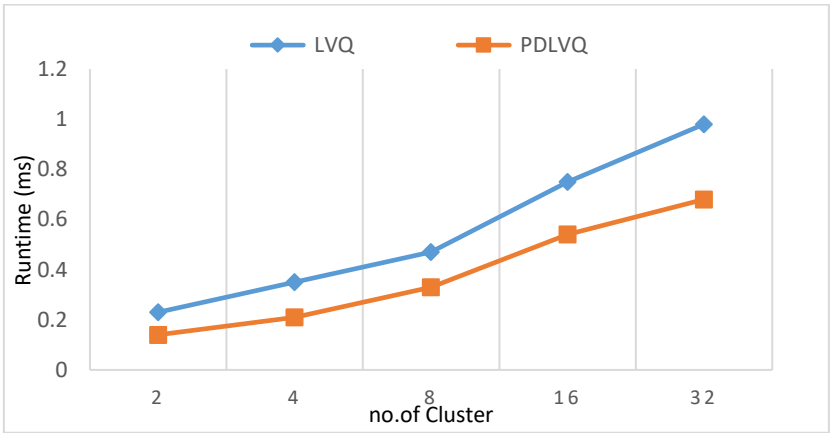

Fig. 5. Runtime for Ecoli dataset in different number of clusters

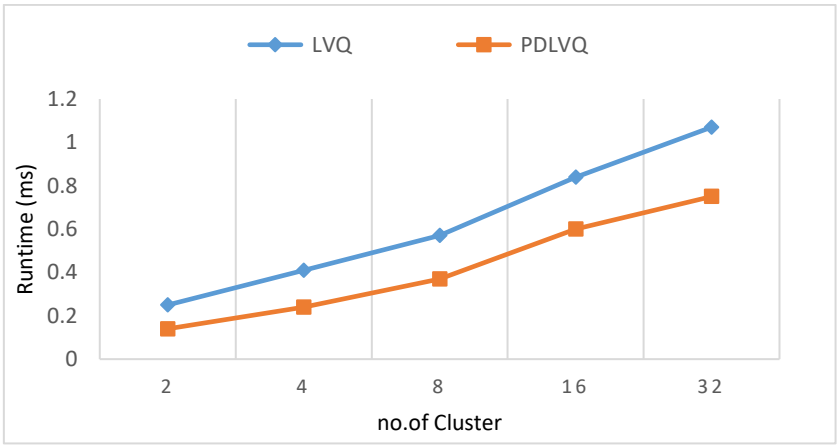

Fig. 6. Runtime for Ecoli dataset in different dimensions

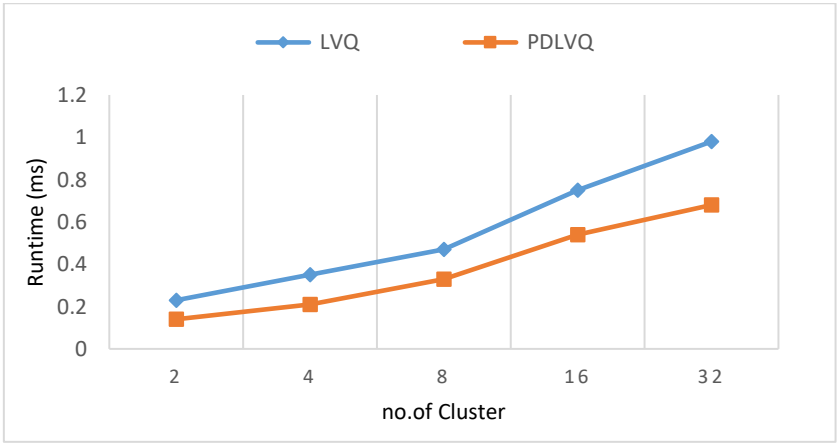

Fig. 7. Runtime for Pima dataset in different number of clusters

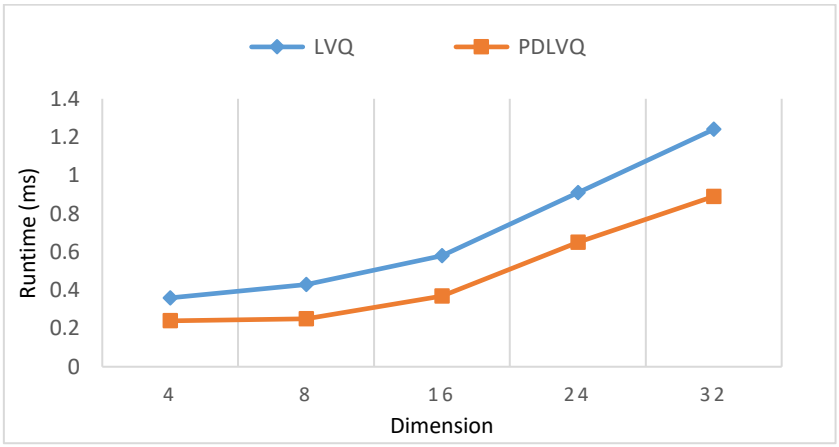

Fig. 8. Runtime for Pima dataset in different dimensions

5. Conclusion

LVQ uses Euclidean distance in order to classify any input vector for the corresponding appropriate class. These calculations take a long time, which increases the computational complexity of the LVQ due to its high dimensionality. This paper proposes a modified version of LVQ which is called PDLVQ to speed up LVQ. The proposed technique is based on Partial Distance Computation, which uses an effective partial distance strategy to prevent many unnecessary distance calculations. When used on different datasets these datasets have been chosen based on many differences between each other, such as the number of clusters, dimensions and the size of datasets. The experiment results show that PDLVQ outperforms the LVQ in all scenarios and in all datasets. The runtime of PDLVQ compared to the LVQ Algorithm has been enhanced by approximately 37% when the dimensions of Pima have been increased to be 32 dimensions. Moreover, when the number of clusters has been increased in Pima the runtime that has been taken in PDLVQ outperform LVQ by 35%. Also, it has been noticed that when the dataset has a high number of clusters or dimensions PDLVQ is giving better results to decrease the runtime than LVQ. Also, it can be noticed that when the modified version of LVQ which is PD LVQ is used in large data sizes this gives a clear enhancement compared with a smaller data size. Finally, we conclude that adding Partial Distance computation to the Learning Vector Quantization algorithm improves the algorithm by saving the time that is wasted in unnecessary computations in the traditional version.

References

1. A r t e l t, A., B. H a m m e r. Efficient Computation of Counterfactual Explanations of LVQ Models. – arXiv preprint arXiv:1908.00735, 2019.
2. H a s h e m, I. A. T., I. Y a q o o b, N. B. A n u a r, S. M o k h t a r, A. G a n i, S. U. K h a n. The Rise of "Big Data" on Cloud Computing: Review and Open Research Issues. – Information Systems, Vol. **47**, 2015, pp. 98-115.
3. H u a n g, W., H. W a n g, Y. Z h a n g, S. Z h a n g. A Novel Cluster Computing Technique Based on Signal Clustering and Analytic Hierarchy Model Using Hadoop. – Cluster Computing, Vol. **22**, 2019, No 6, pp. 13077-13084.
4. C h o i, T. M., S. W. W a l l a c e, Y. W a n g. Big Data Analytics in Operations Management. – Production and Operations Management, Vol. **27**, 2018, No 10, pp. 1868-1883.
5. A k h i a t, Y., Y. M a n z a l i, M. C h a h h o u, A. Z i n e d i n e. A New Noisy Random Forest Based Method for Feature Selection. – Cybernetics and Information Technologies, Vol. **21**, 2021, No 2, pp. 10-28.
6. T c h a m o v a, A., J. D e z e r t, N. B o c h e v a, P. K o n s t a n t i n o v a, B. G e n o v a, M. S t e f a n o v a. A Study on Human Learning Ability during Classification of Motion and Colour Visual Cues and Their Combination. – Cybernetics and Information Technologies, Vol. **21**, 2021, No 1, pp. 73-86.
7. M a d h u m a l a, R. B., H. T i w a r i, V. C. D e v a r a j. Virtual Machine Placement Using Energy Efficient Particle Swarm Optimization in Cloud Datacenter. – Cybernetics and Information Technologies, Vol. **21**, 2021, No 1, pp. 62-72.
8. K u m a r, K. D i n e s h, E. U m a m a h e s w a r i. HPCWMF: A Hybrid Predictive Cloud Workload Management Framework Using Improved LSTM Neural Network. – Cybernetics and Information Technologies, Vol. **20**, 2020, No 4, pp. 55-73.

9. Y a z i c i, M., S. B a s u r r a, M. M. G a b e r. Edge Machine Learning: Enabling Smart Internet of Things Applications. – Big Data and Cognitive Computing, Vol. **2**, 2018, No 3, pp. 26.
10. K a d e n, M., M. L a n g e, D. N e b e l, M. R i e d e l, T. G e w e n i g e r, T. V i l l m a n n. Aspects in Classification Learning – Review of Recent Developments in Learning Vector Quantization. – Foundation of Computing and Decision Sciences, Vol. **39**, 2014, No 2, pp. 79-105.
11. W u, K. L., M. S. Y a n g. Alternative Learning Vector Quantization. – Pattern Recognition, Vol. **39**, 2006, No 3, pp. 351-362.
12. M e l i n, P., J. A m e z c u a, F. V a l d e z, O. C a s t i l l o. A New Neural Network Model Based on the LVQ Algorithm for Multi-Class Classification of Arrhythmias. – Information Sciences, Vol. **279**, 2014, pp. 483-497.
13. D e v i, K. J., G. B., M o u l i k a, K. S r a v a n t h i, K. M. K u m a r. Prediction of Medicines Using LVQ Methodology. – In: Proc. of International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS'17), IEEE, 2017, pp. 388-391.
14. B l a i e c h, A. G., K. B e a n K h a l f a, M. B o u b a k e r, M. H. B e d o u i. LvQ Neural Network Optimized Implementation on FPGA Devices with Multiple-Wordlength Operations for Real-Time Systems. – Neural Computing and Applications, Vol. **29**, 2018, No 2, pp. 509-528.
15. P u t r a, D. S., Y. U. W. W e r u. Pattern Recognition of Electromyography (EMG) Signal for Wrist Movement Using Learning Vector Quantization (LVQ). – In: Proc. of IOP Conference Series: Materials Science and Engineering, Vol. **506**, 2019, No 1, pp. 12-20.
16. S h e i k h A b d u l l a h, S. N. H., F. A. B o h a n i, B. H. N a y e f, S. S a h r a n, O. A l A k a s h, R. I q b a l H u s s a i n, F. I s m a i l. Round Randomized Learning Vector Quantization for Brain Tumor Imaging. – Computational and Mathematical Methods in Medicine, 2016.
17. B i e h l, M., A. G h o s h, B. H a m m e r. Learning Vector Quantization: The Dynamics of Winner-Takes-All Algorithms. – Neurocomputing, Vol. **69**, 2006, No 7-9, pp. 660-670.
18. G h o s h, A., M. B i e h l, B. H a m m e r. Performance Analysis of LVQ Algorithms: A Statistical Physics Approach. – Neural Networks, Vol. **19**, 2006, No 6-7, pp. 817-829.
19. M o k b e l, B., B. P a a s s e n, F. M. S c h l e i f, B. H a m m e r. Metric Learning for Sequences in Relational LVQ. – Neurocomputing, Vol. **169**, 2015, pp. 306-322.
20. T z a n a k o u, E. M. Supervised and Unsupervised Pattern Recognition: Feature Extraction and Computational Intelligence. CRC Press, 2017.
21. K o h o n e n, T. Improved Versions of Learning Vector Quantization. – In: Proc. of IJCNN International Joint Conference on Neural Networks, IEEE, 1990, pp. 545-550.
22. N a o u m, R. S., Z. N. A l-S u l t a n i. Learning Vector Quantization (LVQ) and k-Nearest Neighbor for Intrusion Classification. – World of Computer Science and Information Technology Journal (WCSIT), Vol. **2**, 2012, No 3, pp. 105-109.
23. L e u n g, K. M. Learning Vector Quantization. Department of Computer and Information Science, Polytechnic University, 2009.
24. P a n d y a, A. S., R. B. M a c y. Pattern Recognition with Neural Networks in C++. CRC Press, 1995.
25. S a i n i n, M. S., R. A l f r e d, F. A h m a d. Ensemble Meta Classifier with Sampling and Feature Selection for Data with Imbalance Multiclass Problem. – Journal of Information and Communication Technology, Vol. **20**, 2021, No 2, pp. 103-133.
26. K u m a r, N., D. K u m a r. An Improved Grey Wolf Optimization-Based Learning of Artificial Neural Network for Medical Data Classification. – Journal of Information and Communication Technology, Vol. **20**, 2021, No 2, pp. 213-248.
27. B l a c k, T., D. B. F o g e l, Z. M i c h a l e w i c z. Evolutionary Computation 1: Basic Algorithms and Operators. CRC Press, 2018.
28. K i m, K.-S., I. H a n. The Cluster-Indexing Method for Case-Based Reasoning Using Self-Organizing Maps and Learning Vector Quantization for Bond Rating Cases. – Expert Systems with Applications, Vol. **21**, 2001, No 3, pp. 147-156.
29. A b u A l g h a n a m, O., L. A l b d o u r, O. A d w a n. Multimodal Biometric Fusion Online Handwritten Signature Verification Using Neural Network and Support Vector Machine. – Transactions, Vol. **7**, 2021, No 8.
30. A b u a l g h a n a m, O., M. Q a t a w n e h, W. A l m o b a i d e e n. A Survey of Key Distribution in the Context of Internet of Things. – Journal of Theoretical and Applied Information Technology, Vol. **97**, 2019, No 22, pp. 3217-3241.

31. S a l m a n, M., D. H u s n a, S. G. A p r i l i a n i, J. G. P i n e m. Anomaly Based Detection Analysis for Intrusion Detection System Using Big Data Technique with Learning Vector Quantization (LVQ) and Principal Component Analysis (PCA). – In: Proc. of International Conference on Artificial Intelligence and Virtual Reality, 2018, pp. 20-23.
32. A r u l k u m a r, V., P. V i v e k a n a n d a n. An Intelligent Technique for Uniquely Recognising Face and Finger Image Using Learning Vector Quantisation (LVQ) Based Template Key Generation. – International Journal of Biomedical Engineering and Technology, Vol. **26**, 2018, No 3-4, pp. 237-249.
33. M a w e n g k a n g, H., S. E f e n d i, S. Hybrid Learning Vector Quantization (LVQ) Algorithm on Face Recognition Using Webcam. – In: Proc. of IOP Conference Series: Materials Science and Engineering, Vol. **420**, 2018, No 1, pp. 012126.
34. Y u n i a r n o, E. M., M. H. P u r n o m o. Indonesian Batik Image Classification Using Statistical Texture Feature Extraction Gray Level Co-Occurrence Matrix (GLCM) and Learning Vector Quantization (LVQ). – Journal of Telecommunication, Electronic and Computer Engineering (JTEC), Vol. **10**, 2018, No 2-3, pp. 67-71.
35. S a r d o g a n, M., A. T u n c e r, Y. O z e n. Plant Leaf Disease Detection and Classification Based on CNN with LVQ Algorithm. – In: Proc. of 3rd International Conference on Computer Science and Engineering (UBMK), IEEE, 2018, pp. 382-385.
36. A m e z c u a, J., P. M e l i n, O. C a s t i l l o. New Classification Method Based on Modular Neural Networks with the LVQ Algorithm and Type-2 Fuzzy Logic. Springer, 2018.
37. P a l i w a l, K. K., V. R a m a s u b r a m a n i a n. Effect of Ordering the Codebook on the Efficiency of the Partial Distance Search Algorithm for Vector Quantization. – IEEE Transactions on Communications, Vol. **37**, 1989, No 5, pp. 538-540.
38. V a n V e e n, R., V. G u r v i t s, R. V. K o g a n, S. K. M e l e s, G. J. de V r i e s, R. J. R e n k e n, M. B i e h l. An Application of Generalized Matrix Learning Vector Quantization in Neuroimaging. – Computer Methods and Programs in Biomedicine, Vol. **197**, 2020, pp. 105708.
39. N o w a k o v á, J., M. P r í l e p o k, V. S n á š e l. Medical Image Retrieval Using Vector Quantization and Fuzzy S-Tree. – Journal of Medical Systems, Vol. **41**, 2017, No 2, pp. 1-16.
40. Y a n g, C. H., S. J. W a n g. Accelerating VQ-Based Codeword Search on the Basis of Partial Search Strategy. – Computer Standards & Interfaces, Vol. **28**, 2005, No 2, pp. 231-240.
41. A l-Z o u b i, M., A. H u d a i b, A. H u n e i t i, B. H a m m o. New Efficient Strategy to Accelerate k-Means Clustering Algorithm. – American Journal of Applied Sciences, Vol. **5**, 2008, No 9, pp. 1247-1250.
42. K i m, B. A Fast k-Prototypes Algorithm Using Partial Distance Computation. – Symmetry, Vol. **9**, 2017, No 4, pp. 58.
43. S h a f r o n e n k o, A., A. D o l o t o v, Y. B o d y a n s k i y, G. S e t l a k. Fuzzy Clustering of Distorted Observations Based on Optimal Expansion Using Partial Distances. – In: Proc. of 2nd IEEE International Conference on Data Stream Mining & Processing (DSMP'18), IEEE, 2018, pp. 327-330.
44. B l a k e, C., C. Merz. UCI Repository of Machine Learning Databases. Bib Sonomy, 1998.
45. **http://www. ics. uci. edu/~ mlearn/MLRepository**

Received: 01.03.2022; Second Version: 23.03.2022; Accepted: 29.03.2022 (fast track)