Parallalizable Deep Self-Organizing Maps for Image Classification

Chathurika S. Wickramasinghe, Kasun Amarasinghe, Milos Manic
Virginia Commonwealth University, Richmond, Virginia
brahmanacsw@vcu.edu, amarasinghek@vcu.edu, misko@ieee.org

Abstract— **Self-organizing Maps (SOMs) are neural network architectures which are used to learn from unlabeled data. Despite being proven to be useful in many areas, SOMs have limited capability of performing high level feature abstraction due to its shallow structure. As a solution, deep self-organizing maps (DSOM), have been proposed. DSOMs enable multiple levels of abstraction in unsupervised learning with a hierarchical deep structure. However, training of DSOMs is computationally expensive, limiting its usability. This paper presents a DSOM architecture that is easily parallelizable and hence more computationally efficient (PD-SOM). The presented architecture has three main advantages: 1) Unsupervised learning based image classification, 2) High-level feature abstraction and 3) Less computationally expensive training while maintaining high accuracy levels. The PD-SOM architecture was implemented on the benchmark MNIST hand written character dataset. To test the robustness and the generalization capability of the presented PD-SOM architecture, testing was done with: 1) small training sets and 2) varying degrees of noise. It was shown that the presented architecture consistently outperformed the previously proposed DSOM while showing ~18% decrease in training time for the MNIST dataset.**

Keywords—Self Organizing Map (SOM); Deep Self Organizing Map (DSOM); MNIST; Image classification; Deep Learning

I. INTRODUCTION

Self-organizing Maps (SOM) are unsupervised learning architectures introduced by T.Kohonen [1]. SOMs are based on the winner-take-all algorithm and possess the capability of creating spatially organized representations of input patterns [2] [3] [4] [5]. Furthermore, SOMs have the capability of compressing the information while preserving the most important relationships of primary data items, which can be considered as a process of abstraction.

SOMs have been proven to be suitable for visualization of high dimensional data and in the exploratory phase of data mining [6] [7]. Therefore, SOMs have been successfully used in a multitude of areas including speech recognition, robotics, process control and telecommunication [1] [2] [3] [4] [6] [7]. Further, it has been shown that SOMs have a better capability of revealing the overlapping structure in clusters when compared to other traditional cluster analysis techniques such as partitive clustering and K-means [8].

Even though SOMs have been proven to extremely useful in visual classification tasks, SOMs have a limited capability of high-level feature abstraction due to its shallow structure [9]. SOMs have other limitations such as fixed network size and not guaranteeing a minimal solution [9]. Further, it has been shown that SOMs are incapable of clearly representing the hierarchical relationships that exist in input data [9]. Therefore, traditional SOMs lack the capability of identifying highly important features that exist in data [10] [9] [11].

In order to find solutions to these limitations, research on different architectures of SOMs have been conducted. As a solution for the static structure, [9] and [12] proposed a growing hierarchical SOM (GHSOM) architecture. GHSOM is a neural network model composed of multiple layers where each layer consists of several independently growing SOMs. In [13], the authors proposed a Multilayer Self Organizing Map architecture (HSOM), another hierarchal SOM architecture developed to address the problem of static map size. In this model, each layer depends on the outputs of its previous layer. In a more recent attempt, Deep Self-Organizing Maps (DSOMs) were proposed as a SOM architecture with hierarchical feature abstraction ability [14]. This model consisted of multiple layers of self-organizing map layers and sampling layers. The authors attempted at incorporating the concepts in Deep Learning with the concept of SOM to create SOMs with high-level feature abstraction capability.

In this paper, we present an extension of this work. The implementation of DSOM presented in [14] was able to achieve improved classification accuracies over supervised version of traditional SOM. However, the presented DSOM is computationally expensive and it has the same problem of static map size as in traditional SOMs. To our best knowledge, the work presented in [14], is the only research work carried out on DSOM and there is no further work on improving the capabilities and computational time on DSOMs.

Therefore, this paper presents a easily parallelizable Deep Self Organizing Maps architecture (PD-SOM) with the main intention of improving the computational time of DSOMS while retaining performance. The presented PD-SOM has three main advantages: 1) Unsupervised learning based image classification, 2) High-level feature abstraction and 3) Less computationally expensive training while maintaining high accuracy levels. In addition to the main advantages, PD-SOM partially overcomes the static map size problem that exist in traditional SOMs and DSOMs. The presented PD-SOM architecture was tested on the MNIST hand written character recognition dataset. It has to be noted that the goal was not to improve the state-of-the art classification accuracy for the MNIST dataset. Convolution neural networks (CNNs) and Recurrent Neural Networks based architectures have achieved the best classification accuracies for the MNIST dataset [15]

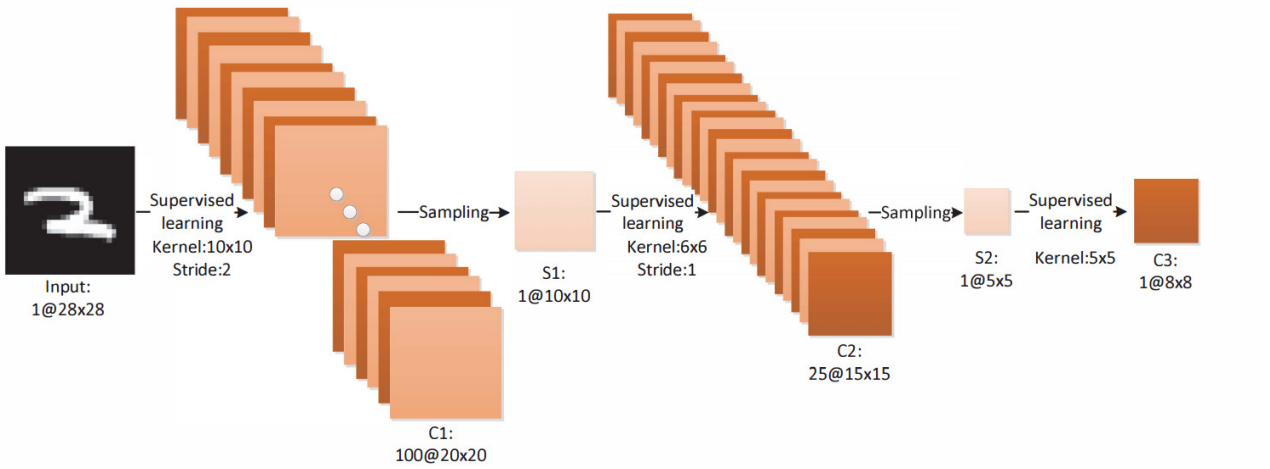

Figure 1: Linear DSOM used for Handwritten Character Recognition by Liu et al [14]

[16] [17]. However, one limitation of those algorithms is that they require labeled data for training the classifiers [18]. The presented PDSOM architecture employs unsupervised learning. Therefore, it has to be noted that the goal of this work is to achieve highest possible accuracy while training with unlabeled data in a computationally efficient manner, not to improve the state-of-the art classification accuracy for the MNIST dataset.

The rest of the paper is organized as follows. Section II provides a detailed review to the DSOMs. Section III describes the proposed parallelizable PD-SOM architecture. Experimental setup and result discuss in section IV. Section V discusses the conclusions proposed 2D DSOM architecture.

II. DEEP SOM

This section briefly introduces Deep Self Organizing Maps.

Deep Self Organizing Maps (DSOMs) were first proposed by Liu et al [22]. DSOMs can be seen as a combination of concepts: traditional supervised SOMs and Convolution Neural Network (CNNs) [14]. In Convolution Neural Networks (CNNs), each unit in a layer receives inputs from a set of units located in a small neighborhood in its previous layer [19], [20]. CNN implements the idea of the local receptive field, such that it can extract elementary visual features such as edges, end points and corners. These extracted features are combined by subsequent layers in order to detect higher order features. The idea of DSOMs is similar to CNN where a SOM in a higher level layer is able to learn more abstract information than the SOM in its previous layer. The last layer of a DSOM contains one map which stores the information needed for classification. Figure 1 shows a three layer DSOM architecture which has been used for handwritten character recognition by Liu et al [22]. DSOM has shown significant accuracy improvements compared to the traditional supervised SOM.

DSOM provides a high level of feature abstraction relative to traditional SOM. In regular SOM, it maps entire observation space into a single 2D grid of neurons whereas in DSOM, the input image is processed in patches.

The DSOM architecture consists of self-organizing map layers and sampling layers, arranged in a deep hierarchical structure. A self-organizing layer is made up of multiple SOMs, with each SOM focuses only on one local region (patch) of the input pattern (see Figure 2). Patch size is defined as the local region of input pattern where a particular SOM map performs its learning. Stride defines how many pixels the two sub-regions are apart from each other. Figure 2 illustrates how winning neuron indexes of all the maps are organized into another 2D grid in next layer, which is called sampling layer. It acts as the input pattern to the second SOM layer. These self-organizing layers and sampling layers can be concatenated one after the other, which will allow the deep hierarchical structure (Figure 3). Such that, local information will be gathered, forming more global information in higher layers.

Input pattern can be represented as X where $X=\{x_1,\ldots x_N\}$ where N represent the number of the input patterns. The training of the DSOM is carried out with the following steps.

Step 1: Weight initialization: Randomly initialize all the weights in the network.

Step 2: SOM Layer: Select a random input pattern(X) from the training dataset. As in Figure 2, assuming the size of input or output of sampling layer is M x M, the self-organizing layer or the SOM layer uses multiple maps $\{C_{1,1}, C_{1,2}, \ldots, C_{Nmaps*Nmaps}\}$ to model the input where. Each map focuses on a K x K sub region (patch) from input pattern/sampling layer output. Let's assume the stride as s. Number of maps can be calculated as

$$N_{map} = ceil\left(\frac{M-K}{S}\right)+1 \quad (1)$$

where the ceil(*) calculates the smallest integer upper bound and M represents the number of pixels on one side.

As mentioned, each patch is trained using a separate SOM with T number of neurons. So the number of SOM maps will be equal to (N_{map} x N_{map}). Each map contains T nodes/ neurons. Assuming each patch from position {p,q} can be modelled by

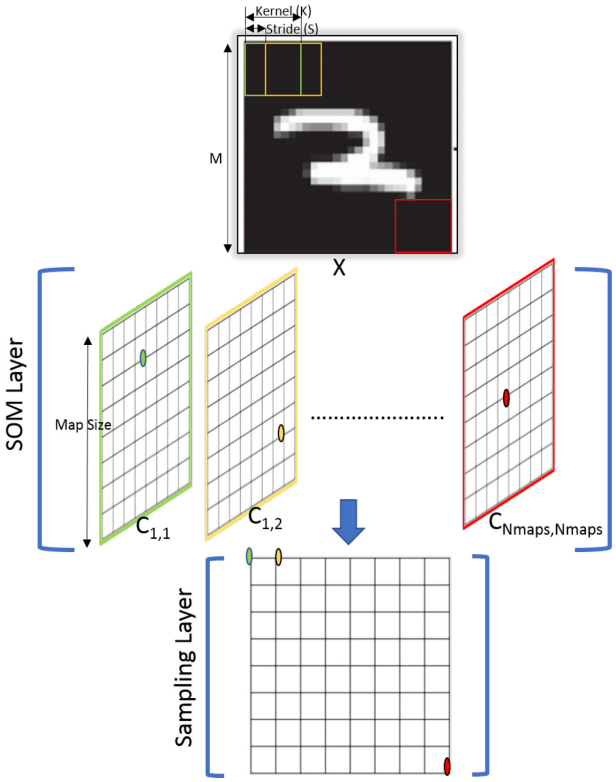

Figure 2: Sampling layer creation in DSOM

a map with set of neuron weights denoted as $\{w^l_{1,p,q}, \ldots, w^l_{T,p,q}\}$. Here *l* indicates index of the layer. Similar to a traditional SOM, the patch $x_{p,q}$ extracted from the input first updates the map by finding the best matching unit/winning neuron (BMU) j^* as follows:

$$j^* = \arg\min_j \|x_{p,q} - w^l_{j,p,q}\|^2 \tag{2}$$

Once the BMU is found, weights in the neighbourhood of neuron j* can be updated as follows:

$$w^i_{j,p,q}(t+1) = \quad + \eta(t)\alpha(t,j,j^*)\left(xp,q - {}^{l}_{w_{j,p,q}}(t)\right) \tag{3}$$

$$j \in N_j{}^*$$

where, N_{j^*} defines the neighbourhood region, and *n(t)* is the learning rate at epoch t. The learning rate can be changed through the epochs as follows:

$$\eta(t) = 0.49(1 - t/\text{epoch}) + 0.01 \tag{4}$$

In order to calculate the neighbourhood, the following equation can be used [25].

$$\alpha(t, j, j^*) = \exp\left(-\frac{\|\mathbf{r}_{j^*} - \mathbf{r}_j\|}{2\delta\, t^2}\right) \tag{5}$$

Step 3: Sampling Layer: Once the BMUs are calculates for all the patches, sampling layer combines information of maps from the preceding SOM layer. The BMU for each patch is extracted and placed in a 2D grid (sampling layer) at the patch index. (See Figure 3).

Step 4: Repeat Step 2-3: The sampling is a unique representation of the input image and it becomes the input for the next SOM layer. This process is repeated until the last layer.

Step 5: Repeat 2-4: These steps are repeated until the maximum number of iterations (epochs) has been reached or until specified convergence criteria are met.

As same as in SOMs, structure of the map can be defined by neighbourhood relationships between adjacent neurons. During the training weight vector in the map in the neighbourhood of the BMU are updated such that they will move closer to the input vector. Once the SOM is trained, in the final trained SOM, each neuron is assigned a class. The assignment is carried out with respect to neuron hits.

III. PD-SOM for Image Clasification

This section elaborates the presented PD-SOM architecture.

The main objective of the presented PD-SOM is to achieve reduced computation times while improving accuracy and generalization capability. The main distinction of PD-SOM from the earlier presented DSOM architecture is the use of multiple parallel SOMs of different sizes in the SOM layer whereas DSOM uses only one SOM in the SOM layer.

When designing a SOM, selection of proper map size is very important to avoid adverse effects such as overfitting to the training data [21] [22]. It has been shown that if map size is too small, it might not explain important differences that should be detected [23]. If map size is too large, there is a possibility of overfitting to the training data [24]. Furthermore, different classes have different number of features that defines them. Therefore, a single map size may not be adequate to capture the distinguishing features for all classes. For example, in the MNIST dataset, each character has different features that distinguish it from other characters such as, number of segments, left-right density ratio, bottom-up density ratio [25]. It is apparent that the number of features associated with one letter is differ from another and selecting a single map size that can represent all classes very well can be extremely difficult. Therefore, a method of combining features extracted from maps of different sizes can potentially represent features in all classes.

As mentioned, a map which is too large can result in over-fitting to the training data set and if the map is too small, it may result in under-fitting. However, a smaller map can learn low granular (high-level) features and a large map can learn highly granular (low-level) features. Therefore, the use of two different sized maps together in parallel and the combination of their information into a single 2D map in the sampling layer, results in generalized information than using one single map. It provides a balance between learning too much information (overfitting) or too less information (under-fitting) to the proposed PD-SOM.

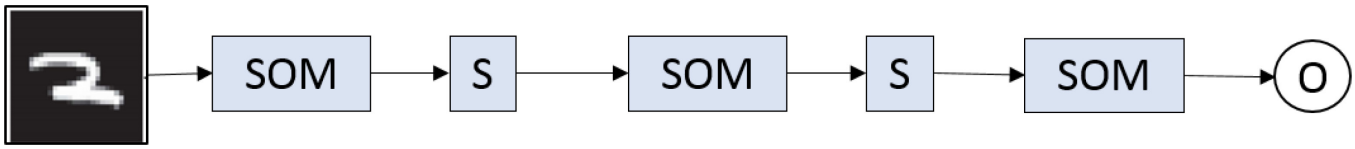

Figure 3: Linear DSOM architecture

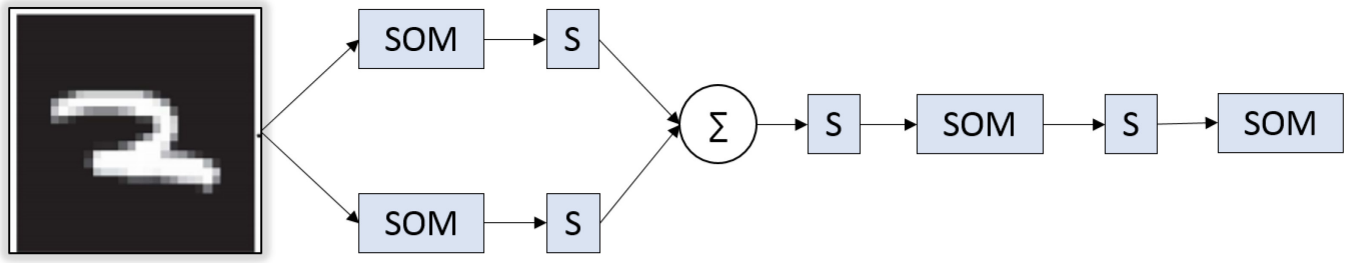

Figure 4: Proposed PD-SOM architecture (Type 1)

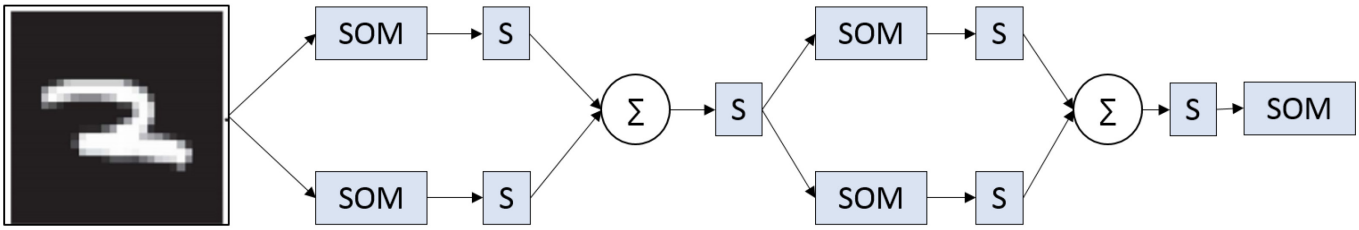

Figure 5: PD-SOM architecture - possible extension (Type 2)

The linear DSOM proposed in [14] does not solve the static map size problem in traditional SOMs since only a single sized map is used in a single layer. Therefore, in linear DSOM, a patch of the image is learned only using a single map size limiting the learning capability of the SOM. Conversely, the presented PD-SOM partially alleviates the static nature by using different sized maps in the same layer. Therefore, PD-SOM has more flexibility in learning features of different resolutions in the same layer. For instance, if a small map size and a large map size is used in parallel, the smaller map can learn the high-level features and the larger map can learn the low level-features. In the case of small map missing a certain feature, there's a probability where the larger map can catch it. Therefore, the PD-SOM architecture has a better probability of capturing more features that represent a certain class without overfitting to training data over the previously proposed linear DSOM.

Figure 3 illustrates the linear DSOM architecture that was presented in [14]. Figure 4 and Figure 5 illustrate two types of the proposed PD-SOM architecture. Training of the PD-SOM can be carried out in the following steps.

Step 1: Weight initialization: Randomly initialize all the weights in the network.

Step 2: SOM Layer: Step 2 from the previous section is used in parallel on the parallel SOM layers

Step 3: Sampling Layer: In this step, the outputs from the Parallel SOMs are combined to create the sampling layer. A separate sampling layer is created for each of the parallel SOMs using the same method described in the previous section. Then the parallel sampling layers are combined together to make a single sampling layer. This is carried out using a simple summation process. I.e. the 2D matrices are added together to create a single 2D matrix, which is the combined sampling layer. Each of the parallel sampling layers are unique representations of the input image patch and the summation of unique representations creates another unique representation.

Step 4: Repeat Step 2-3: The combined sampling layer becomes the input for the next SOM layer. This process is repeated until the last layer.

Step 5: Repeat 2-4: These steps are repeated until the maximum number of iterations (epochs) has been reached or until specified convergence criteria are met.

Figure 5 illustrates another possible architecture of the PD-SOM. It illustrates a possible extension where the model can be made deeper by adding another layer of parallel SOMs. Therefore, the model can be made deeper by mixing the two ideas of linear DSOM and PD-SOM.

IV. Experimental Results and Discussion

This section discusses the experimental setup and the results of the experiments.

All the DSOM and PD-SOM architectures were carried out using MNIST: the benchmark hand written character recognition dataset [26]. Training was performed as unsupervised i.e. input vector contained only image descriptors

TABLE 1: THE ARCHITECTURE OF THE LINEAR DSOM

Layer	Nmap	MapSize	Patch(k)	Stride
Layer 1	100	-	10*10	2
Layer 2	25	15*15	6*6	1
Layer 3	1	8*8	5*5	1

without class information. After the training was performed, cluster identification was performed based on neuron hits.

The training dataset contained 3000 images whereas testing dataset contained 10000 images. A smaller training set was used to reduce the training time and to evaluate the generalization capability of the DSOM architectures. In the MNIST dataset, each image is the size of 28*28 pixels. For the proposed architectures, the performance were evaluated using different map sizes in SOM layers as well as by introducing different noise levels into validation set.

Basic hyper-parameters such as patch size, stride, and kernel sizes were selected based on the work by Liu et al [14]. In order to compare the two architectures, patch size and stride was kept as constant values. In this study, only square maps with the size of one side ranging from 12 to 28 were used, as they have been shown the best accuracies [14]. The number of epochs during the training was set to 5 in order to reduce the experiment time.

A. Linear DSOM architecutre proposed by Liu et al.

The linear DSOM architecture was implemented with the parameters set identical to [14]. Table 1 presents the architecture details of the linear DSOM. As mentioned the map sizes ranged from 12x12 to 28x28. For brevity, only results for the best four models are presented (See Table 2). Different map sizes were tested to observe the effect of the map size on classification accuracy. It was observed that if the number of neurons in the map is low, then the mean accuracy was low because the small map learned an insufficient set of features. It has to be noted that only the map sizes for the first layer were changed, and the map sizes for the second and third layers were kept constant. Liu et al. observed that accuracy increases with the map size from 6x6 to 15x15 and noticed that map sizes beyond that point didn't result in a significant improvement in accuracy and only resulted in heavier computation times [14]. Therefore, the second layer map size was selected as 15x15. Similarly, the third layer size was set to 8x8.

In order to test the robustness of the algorithm to noise, testing was carried out with different noise levels introduced to test data. As expected, results show that when noise level was increased, the testing accuracy dropped in all models and it was noticed that there was no specific map size which performed significantly better than the rest.

B. Presented PDSOM architecture

The presented PD-SOM architecture was implemented with two layers depth and width (Figure 4). In this version of the implementation, two SOMs were used in parallel in the SOM layer and two layers were used in terms of depth. Table III shows architecture details of the two layered PD-SOM. When selecting the map sizes for the parallel SOM layer, two different test cases were considered. 1) Same sized SOMs in parallel; 2) Different sized SOMs in parallel. In the second case, one SOM was set to have a constant map size and the other one was changed to better identify the changes in performance relative to different map sizes.

TABLE 2: THE ARCHITECTURE OF THE PORPOSED PD-SOM

Layer	Nmap	MapSize	Patch(k)	Stride
Layer 1	100	-	10*10	2
Layer 2	1	8*8	5*5	1

In **Test Case 1**, it was observed that using two same sized SOMs in parallel, did not improve the classification accuracy when compared to the linear DSOM. Further, it was noticed that it sometimes had an adverse effect on classification accuracy. This could be the result of learning of redundant features in the same sized maps. Further, use of same sized maps can result in the two maps learning contrasting features in the same granularity which prevents creating a unique feature map for a class.

In **Test Case 2,** four models were tested with a range of different map sizes. Table IV presents the results obtained for the four best models. As mentioned, one map of the parallel SOMs was kept constant and the second map was varied. It was noticed that all models performed consistently until the noise level was 20% and noticed an accuracy drop when the noise level was increased to 40%. Further, it was noticed that all the models were able to achieve similar accuracies which were slightly better than the more computationally expensive linear DSOM. PD-SOM with two different maps sizes partially solves the static map size problem encountered by the traditional SOM and linear DSOM, since different sized maps can learn features of different granularities.

Table V presents the overall mean accuracies obtained for four best models of DSOM and PD-SOM. It was observed that even though training accuracies are almost similar for both DSOM and PD-SOM, testing accuracy and performance on noisy data is better in PD-SOM relative to DSOM. According to the mean accuracies in Table 5, the PD-SOM shows 2% accuracy improvement for testing and 3% accuracy improvement for noisy data sets. In addition to the improvements in accuracy, PD–SOM showed around 18% reduction in computation time relative to the three layered DSOM architecture. Therefore, the experiment provides evidence to the fact that the PD-SOM can achieve better accuracies with reduced training times. Further, it should be noted that this was tested on a relatively small training dataset (3000 images). For a larger dataset, this has the potential of saving a considerable amount of time.

V. CONCLUSIONS

This paper presented a novel architecture for a parallelized version of a Deep Self-Organizing Map (PD-SOM). This work was an extended analysis and an improvement of the work proposed by Liu et al [14]. The presented PD-SOM provides three main advantages: 1) Unsupervised learning based image classification, 2) High-level feature abstraction for SOM and 3) Less computationally expensive training while maintaining high accuracy levels. The presented PD-SOM was implemented and

TABLE 3: RESULTS OBTAINED FOR LINEAR DSOM ALGORITHM

Model	Layer 1 MapSize	Train Accuracy	Test Accuracy	Noise level				
				2	5	10	20	40
1	18*18	85.68	81.876	82.032	81.9876	81.878	81.056	73.09
2	20*20	84.68	79.396	77.83	77.89	77.76	77.248	69.104
3	22*22	85.52	81.44	78.568	78.532	78.346	77.776	69.714
4	24*24	85.88	79.158	83.235	83.05	83.15	81.61	71.85

TABLE 4: RESULTS OBTAINED FOR THE PD-SOM

Model	Layer1		Train Accuracy	Test Accuracy	Noise level				
	SOM1	SOM2			2	5	10	20	40
1	20*20	14*14	86.04	82.234	81.82	81.624	81.676	81.382	73.874
2	20*20	16*16	85.24	81.966	81.555	81.2	81.065	80.8875	74.22
3	22*22	14*14	85.72	82.88	82.874	82.9	82.73	80.93	74.362
4	22*22	16*16	84.72	81.34	81.32	81.23	80.81	80.72	73.25

TABLE 5: RESULTS COMPARISON BETWEEN THE PD-SOM AND LINEAR DSOM

Architecture	Train Accuracy	Test Accuracy	Noise level					Computation Time
			2	5	10	20	40	
DSOM (3 layered)	85.44	80.4675	80.41625	80.3649	80.2835	79.4225	70.9395	3747.4s
PD-SOM (2 layered)	85.43	82.105	81.89225	81.7385	81.5703	80.9798	73.9265	3076.6s

tested on the benchmark MNIST handwritten character dataset. The performance of PD-SOM was compared to the linear DSOM in [14] in terms of computation time and classification accuracy. Further, to test the algorithm's robustness to noise, PD-SOM and linear DSOM were tested on varying degrees of noise. Experimental results showed that the PD-SOM reduced the computation time by 18% while improving the classification accuracy. Further, PD-SOM performed well on noisy data making it a viable candidate architecture for further research. From the experimental results it can be concluded that the PD-SOM with different map sizes provides more flexible and robust learning while reducing computation time compared to the linear DSOM. As future work, the presented PD-SOM architecture will be tested on more datasets including CASIA (Chinese handwritten character dataset) and will be extended in depth and width to observe the effects in computation time and accuracy.

VI. REFERENCES

[1] T. Kohonen, "The self-organizing map," *Neurocomputing,* vol. 21, no. 1-3, pp. 1-6, 1998.

[2] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE,* vol. 78, no. 9, pp. 1464-1480, 1990.

[3] T. Kohonen, "Self-Organization of Very Large Document Collection : State of Art," *Springer,* vol. 1, pp. 63-74.

[4] T. Kohonen, S. Kaski, K. Lagus, J. Salo jarvi, H. Jukka, V. Paatero and A. Saarela, "Self organization of a massive document collection," *IEEE Transactions on Neural Networks,* vol. 11, no. 3, 2000.

[5] X. Lin, D. Soergel and G. Marchionini, "A self-organizing semantic map for information retrieval," *Proceeding:SIGIR '91 Proceedings of the 14th annual international ACM SIGIR conference SIGIR '91 Proceedings of the 14th annual international ACM SIGIR conference ,* pp. 262-269, 1991.

[6] J. Vesanto and E. Alhoniemi, "CLustering of the Self organizing map," *IEEE transactions on Neural networks,* vol. 11, no. 3, pp. 586-600, 2000.

[7] T. Kohenen, E. Oja, O. Simula, A. Visa and J. Kangas, "Engineering Application Of Self Organizing Map," *Proceedings Of IEEE,* vol. 84, no. 10, pp. 11772-11781, 1996.

[8] C. Budayan, I. Dikmen and M. T. Birgonul, "Comparing the performance of traditional cluster analysis, self-organizing maps and fuzzy C-means method for strategic grouping," *Expert Systems with Applications,* vol. 36, no. 9, pp. 11772-11781, 2009.

[9] A. Rauber, D. Merkl and M. Dittenbach, "The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data," *IEEE Transactions on Neural Networks,* vol. 13, no. 6, pp. 1331-1341, 2002.

[10] M. Attik, L. Bougrain and F. Alexandre, "Self-organizing Map Initialization," *Artificial Neural Networks: Biological Inspirations (ICANN),* pp. 357-362, 2005.

[11] M. Dittenbach, A. Rauber and D. Merkl, "Uncovering hierarchical structure in data using the growing hierarchical self-organizing map," *Neurocomputing,* vol. 48, no. 1-4, pp. 199-216, 2002.

[12] D. Alahakoon, S. Halgamuge and B. Srinivasan, "Dynamic self-organizing maps with controlled growth for knowledge discovery," *IEEE Transactions on Neural Networks ,* vol. 11, no. 3, pp. 601 - 614, 2000.

[13] J. Lampinen and E. Oja, "Clustering properties of hierarchical self-organizing maps," *Journal of Mathematical Imaging and Vision,* vol. 2, no. 2-3, p. 261–272, 1992.

[14] N. Liu, J. Wang and Y. Gong, "Deep Self-Organizing Map for Visual Classification," *International Joint Conference on Neural Networks (IJCNN),* 2016.

[15] M. Defferrard, X. Bresson and P. Vandergheynst, "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering," *Advances in Neural Information Processing Systems 29 (NIPS 2016),* 2016.

[16] M. Liang and X. Hu, "Recurrent Convolutional Neural Network for Object Recognition," *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 3367-3375, 2015.

[17] A. Graves, Supervised sequence labelling with recurrent neural networks, Springer, 2012.

[18] Y. LeCun1, Y. Bengio and . G. Hinton, Deep learning, Nature, 2015.

[19] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, no. 11, 1998.

[20] M. Zeiler and R. Fergus, "Visualizing and Understanding Convolution Networks," *European Conference on Computer Vision, ECCV 2014: Computer Vision – ECCV 2014,* pp. 818-833, 2014.

[21] C. Sungzoon and C. Keonhoe , "Evolution of neural network training set through addition of virtual samples," in *Evolutionary Computation, Proceedings of IEEE International Conference*, 1996.

[22] . D. Hunter, H. Yu, P. Michael S. , J. Kolbusz and W. Bogdan M. , "Selection of Proper Neural Network Sizes and Architectures—A Comparative Study," *IEEE Transactions on Industrial Informatics,* vol. 8, no. 2, pp. 228 - 240, 2012.

[23] Y.-S. Park, R. Céréghino, A. Compin and S. Lek, "Applications of artificial neural networks for patterning and predicting aquatic insect species richness in running waters," *Ecological Modelling,* vol. 160, no. 3, pp. 265-280, 2003.

[24] L. Steve , G. C. Lee and A. C. Tsoi, "What Size Neural Network Gives Optimal Generalization? Convergence Properties of Backpropagation," *Technical Report,* 1996.

[25] L. Heutte, T. Paquet, J. V. Morea, Y. Lecourtier and C. Olivier, "A structural/statistical feature based vector for handwritten character recognition," *Pattern Recognition Letters,* vol. 19, no. 7, pp. 629-641, 1998.

[26] Y. Lecun and C. Cortes, *Handwritten digit database (MNIST),* 2010.

[27] X. Jin, B. W.Wah, X. Chen and Y. Wang, "Significance and Challenges of Big Data Research," *Big Data Research,* vol. 2, no. 2, pp. 59-64, 2015.

[28] K. Jarrett, K. Kavukcuoglu, M. Ranzato and Y. LeCun, "What is the Best Multi-Stage Architecture for Object Recognition".

[29] A. Coates, H. Lee and A. Ng, "An Analysis of Single-Layer Networks in Unsupervised Feature Learning," in *Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011.