Comprehensive Analysis of the Uses of GPU and CUDA in Soft-Computing Techniques

Adhesh Garg*, Diwanshi Gupta*, Parimi Praveen Sahadev*, Sanjay Saxena*

Department of Computer Science & Engineering, IIIT Bhubaneswar, Bhubaneswar, Odisha
Corresponding Author: sanjay@iiit-bh.ac.in

Abstract- Now a day it is quite often seen that Soft-computing approaches are the cast off frequently in Computer and Information Technology. Fuzzy logic, Genetic Algorithms, and Artificial Neural Network are the fundamentals of Soft computing techniques. Several soft computing approaches such as deep neural network require a huge amount of computational power of the CPU to execute a problem and consume lots of time to execute. GPU (Graphics Processing Unit) provides an efficient way to use efficiently all the computational resources in the form of streaming processors. Further, CUDA (Computed Unified Device Architecture) provides a parallel framework specifically designed for GPU. CUDA and GPU are frequently used to implement different soft computing techniques. This article provides a brief introduction of GPU and CUDA and different soft computing techniques. Further, analytical review of the implementation of different soft computing techniques using GPU and CUDA are given.

Keywords- *CUDA, GPU, Soft-Computing, Deep Neural Networks*

I. INTRODUCTION

Soft computing is one the major field of exploration in the field of computer science and technology for the last 3 decades. Soft computing deals with the problems that we do not have a fixed approach or exact solution by the deterministic way such the NP-complete problem. Soft computing includes the machine and deep learning techniques in which the model can be improved for providing a learning rate by training it for the given data [34]

In recent years the advancement of GPU and CUDA by NVidia has led to the solution of many complex problems over time which cannot be calculated over the year as the calculations and problem being too complex or too much time consuming for the CPU. CUDA uses the Multiple cores present in the GPU and the tasks can be run parallel using multithreading and the computation time can be significantly reduced. In the paper, we have examined various research papers where soft computing techniques have been improved with the CUDA and GPU. The rest of the paper is organized in such a manner such that section II gives the introduction of soft computing, section III describes about GPU and CUDA architecture, section IV gives a comprehensive analysis of existing literature of soft computing based on GPU and CUDA, Conclusion is given in section V finally the references are given in section VI.

II. SOFT COMPUTING

Soft computing deals with complex problems that do not have an exact solution using computational methods and algorithms. It utilizes the endurance for dubious nature of problem. Approach to solve a problem using soft computation derives its ingenuity from the potential of human mind to achieve solution with the help of approximation [38]. Soft computing is applied where a direct mathematical formula, no matter how complex, is unable to provide an adequate result. Soft computing has emerged out to be the most sought-after having innumerable applications in the domestic, commercial and industrial sectors [31]. Soft Computing basically comprises of Neural Networks, Fuzzy Logic, Evolutionary Computation and Probabilistic Reasoning as shown in the following figure 1[35]. As the major techniques for soft computing collaborate more advanced techniques are formed like: Fuzzy Neural Networks, Genetic Neural Networks, Fuzzy Evolutionary Algorithms, Genetic Bayesian Networks and many more.

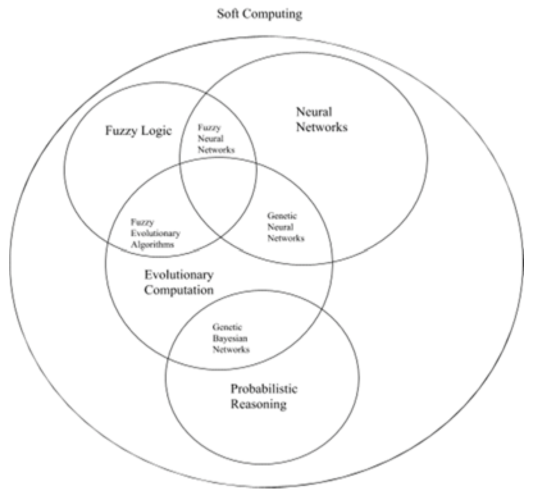

Fig 1: Soft Computing

III. GPU and CUDA

Computer unified Device architecture is a parallel programming platform developed by NVidia to support Multithreading on the Graphical processing unit (GPU) Multicore hardware. CUDA was written using programming languages mainly C and C++. The main purpose of CUDA was to provide the user with a platform for general purpose processing using GPU. [37]

GPU was originally designed for the processing of 3D graphics but over the time GPU evolved from just graphics use to secondary processing option for the user. When the dataset is generally big or there are many lightweight calculations[36].

CUDA is used to run a program in parallel and thus improving its runtime but it is not true in all the cases, as there needs to maintain a balance what task can be calculated in parallel and what should be better done on the single threaded CPU as some of the tasks can be heavy for the GPU threads and in some cases the time of the transition from the GPU to CPU might interfere with the optimal result. GPU computing plays a very vital role. As very computation is needed at the time of computation in the hidden layers [32].

IV. COMPREHENSIVE ANALYSIS OF DIFFERENT LITERATURE

Till now there are several literatures have been done for implementing different soft computing techniques such as fuzzy, genetic and neural network using CUDA on GPU. Here we are going to give the analysis of different current literatures.

1. Nearest neighbor Partitioning and Parallel Nearest Neighboring Partitioning technique in machine learning [1]

Description: Nearest Neighbor Partitioning method is used to find the best and optimal neural network. NNP makes flexible boundaries which helps in making over the previous spherical boundaries which improvise the results but since various stages are in involved in the process due to large data sets the process the very time-consuming. This paper examines the Parallel NNP being run on CUDA with GPU.

Results: When tested NNP parallel on GPU as each of its network classifier on a block and all the processes using multithreading.

Type of GPU: Tesla K10 GPU produced by NVIDIA. There are eight high speed multiprocessors in the GPU Each multiprocessor has thousands of CUDA cores.

2. Hybrid Forecasting Model [2]

Description: Grid which is used to provide energy to our houses is one of a widely used machine. Here Hybrid forecasting model is used as popular deep learning and machine learning approaches are more complex and consume more memory in the process. To enhance the working the HFM model GPU are introduced. To compare the speedup, this will also be run upon the multicore i7 CPU.

Results: The main objective of using the GPU was to split the n step training process parallel through multithreading. It was observed that the process was speed up to 45 times and the process was more work efficient than a single core CPU.

Type of GPU: NVIDIA GPU GTX TITAN X, 3072 CUDA cores with global memory of 12GB, compliant with Maxwell microarchitecture.

3. Image processing using various machine learning and deep learning techniques such as SVM, BIC, CCV, LBP, SHE, PNN, KNN [3]

Description: Agriculture produces the major grain in the world and advancement in the field of computer vision can improve the quality of the grain and the harvest. Here several databases are used for the collection of images regarding the crop insect infestation, microbial infection and discolor grains. The images are used to identify the problems using various techniques of machine learning and deep learning with the help of GPU as the due large datasets multithreading can reduce the processing time and the results can be more efficient.

Results: It is observed that a lot of advancement with regard to computer science in the field of agriculture can take place with techniques such as computer vision and the huge advantage of multicore GPU for easing the process.

4. Genetic Algorithms [4]

Description: Genetic algorithm is a method of solving optimization problems. Image processing requires filters which need to be trained with proper weight and fitness function using a genetic algorithm. This process requires a lot of processing and time so the parallel genetic algorithm is introduced to reduce the processing time.

Results: Parallel implementation of the code reduced the processing time significantly. Direct method was the slowest one among the different Cuda programming approaches then PBM, BBM, SBM.

Type of GPU: Nvidia GeForce 660 GPU

5. Coarse-grained parallelization, fine-grain parallelization [5]

Description: BioHEL is machine learning used for data mining of large datasets and has shown good results on CPU. This paper explores the Technique with large datasets on 2 different GPU. The experiment is carried out in two stages one on highly synthesized data and second on real world problems. The aim is to see the improvement in the time and efficiency of the model on both techniques using both kinds of data.

Results: The experiments show that the in case of coarse-grain approach the optimized configuration between the parallelism and thread work produces the optimal on both the datasets.

Type of GPU: Tesla C2070 with an internal memory of 6 GB and 448 CUDA cores, Tesla C1060 with an internal memory of 4 GB and 30 Multiprocessor.

6. Classification using DARKNET and YOLO [6]

Description: The astronomical data is collected at different telescopes throughout the world. The data is then processed through image processing machine learning techniques in order to classify the data. As these databases are becoming big and big every night. We need better C++ and Python libraries using CUDA GPU based techniques for computation to process these images. Here a method is disused which is also part of AstroCV.

Results: YOLO method with showed a factor of 3 improvement in the results than before due to a different augmentation to the datasets.

7. SVM [7]

Description: Support Vector Machine has been promising algorithm considering it high accuracy in the results but due to the large size of the datasets it was very difficult to train a model.

Here they explored the results using the n-fold cross validation tool to train a model of SVM on GPU.

Results: The parallel method i.e GPUSVM showed significant improvements over the previous data and results are expected to improve even more when using multiple GPU.

Type of GPU: Two Nvidia Tesla C2070 with 6GB of memory and six Nvidia Tesla C2050 with 3GB of memory.

8. Contrastive Divergence [8]

Description: Deep belief networks are widely used in deep learning techniques. Training DBN it difficult considering the large datasets and large number of layers. Here two methods are proposed to make this fast and efficient. Adaptive size technique is applied to choose an appropriate learning rate for the network and training Restricted

Results: It was observed that the speed of training the network was increased up to 46 times.

Type of GPU: Nvidia GeForce GTX 460 and Nvidia GeForce 280.

9. ANN on SVM [9]

Description: Natural disaster such as landslide can be predicted before with the processing of the optical images from the satellite. Here in the experiment images are taken from the google earth to process using machine learning technique SVM and ANN using deep neural networks for landslide recognition. Further, the experiment is conducted using the GPU CUDA to increases the efficiency and processing of the data.

Results: Here the proposed approach to process the optical images have higher accuracy as compared with the previous approach in predicting the area. GPU also helped in achieving higher efficiency and accuracy in the experiment.

Type of GPU: NVidia Tesla K20c GPU with CUDA SDK 7.5

10. Data Clustering, Particle Swarm Optimization, Fish School Search [10]

Description: With the advancement in data science huge volume of data need to process. Clustering is a technique in which a similar type of data is clustered together thus forming clusters. Clustering is itself a NP-type problem. K-Mean and K-Harmonic Mean algorithm are very popular partitioning method for clustering, but they are also not able to give optimal results. Swarm Intelligence method based on the behaviors of the insects and animals have been considered very effective in the field

Results: FSS showed better results than PSO in serial implementation but on parallel implementation on the GPU PSO showed better results than FSS. Processing time was significantly reduced in both the cases and with better results.

Type of GPU: Nvidia GTX 460 with 336 cores.

11. Parallelization of Fuzzy ART algorithm [11]

Description: For the implementation of Fuzzy ART model in CUDA, single dimension blocks and grid are used to simplify vector access.

Results: Peak speed of x57 is achieved for testing as compared to CPU. When a number of categories created is large, relative speed-up between GPU and CPU grows exponentially.

Type of GPU: Different NVIDIA G8X series graphic cards were used.

12. Windowing Scheme ILAS implemented in BioHEL [12]

Description: BioHEL generates sets of rules, each of which is sequentially learnt using GA. Each time the system learns a new rule, it is added to the rule set and all the examples covered by the rule are removed from the training set. In ILAS, the training set is divided into a number of non-overlapping strata. Speed of evolution function on its own: 52.4X. Speed of evolution function's integration with the entire learning process: 58.1X

Results: The speedup varies with the number of windows and for a higher number of windows the speedup tends to decrease.

Type of GPU: For CUDA experiments we used Pentium 4 of 3.6GHz with hyper threading, 2GB of RAM and a Tesla C1060 with 4GB of global memory and 30 multiprocessors. For serial experiments, HPC having each node with 2 quad core processors (Intel Xeon E5472 3.0GHz).

13. Neural Networks training with LMA [13]

Description: The neural network training is carried out by Levenberg-Marquardt Algorithm (LMA), also known as the damped least-squares method. The Filtering and Classification modules use Ensembles composed of five MLP neural networks and 28 input attributes, one hidden layer and one neuron in the output layer.

Results: Multiplication of Jacobian matrix is done using the CUBLAS library from NVIDIA, that implements level 1, 2 and 3 of the existing Basic Linear Algebra Subroutines (BLAS) Library.

Type of GPU: Four types of NVIDIA Graphic cards were used. 1. Athlon 6000 with 2GB RAM, memory size 896MB and a GeForce 260 GTX. 2. Phenom II with 16GB RAM, memory size 4x4096MB and four Tesla c1060.

14. Processing MLPs on GPU [14]

Description: Multi-Layer Perceptron (MLP) is a class of feedforward which consists of (minimum) three layers of nodes: 1. Input Layer, 2. Hidden Layer and 3. Output Layer. To speed-up MLP computation, several MLPs can be processed simultaneously using GPU. Three types of MLP processes were used.

Results: Computation on NVIDIA GPU (Computation time: 17.5 hours) provided 50x speed increase when compared to the highly optimized fastest version of CPU program (8 weeks of computation) for both the GPUs.

Type of GPU: NVIDIA GeForce 8600M GT, NVIDIA GeForce GTX 260

15. Acceleration of SVM with the use of the CSR-GPU-SVM method [15]

Description: SVM is a discriminative binary classifier that is defined by a separating optimal hyperplane. To maximize the utilization of available memory CSR format of SVM is used on GPU. The CUDA functions are based on CSR Vector method, which uses 32-thread warp for multiplying one sparse matrix row by a dense vector.

Results: The use of CSR sparse matrix format allowed us to minimize memory usage, which helps in classifying big sparse collections. The CSR-GPU-SVM is most suitable for big sparse collections.

Type of GPU: For the testing of code, two Nvidia cards were used with 1GB device memory.

1. GeForce 240 (96cores)

2. GeForce GTX 460 (336 cores).

16. SOON and Fuzzy ARTMAP neural networks [16]

Description: Fuzzy ART-based supervised neural networks (NN) are self-organizing algorithms capable of incremental learning. These networks are comprised of an ART_A module, ART_B module and a MapField (MF) which communicates the two ART modules and maps LTM traces to the class selected by an expert. Supervised Orientational invariant Neural Network (SOON) is inspired on ARTMAP theory.

Results: The Fuzzy ARTMAP NN implemented on the GPU performs x100 times faster than equivalent CPU version, while the SOON NN is speeded up by x70 times. Using same patterns, the Fuzzy ART-MAP NN obtains success rate of 48% and SOON of 82% for texture classification

Type of GPU: Dual core 3.2 GHz Pentium 4 with 4 GB RAM, with GPU NVIDIA GeForce GTX285.

17. Implementation of SOMA on GPU [17]

Description: SOMA is modeled after the behavior of intelligent individuals working cooperatively to achieve a common goal. One round of algorithm in SOMA is called migration. To implement SOMA using CUDA, cuSOMA was created and implemented as a C++ class with core components in c-CUDA.

Results: Tests showed high speedups in comparison to CPU implementation of SOMA with highest recorded speedup of 126.9. Data transfer times on overall runtime for large populations grows slower than actual computation time.

Types of GPU:

NVidia Tesla V2075, 448 thread processors at 1150 MHz and Intel Xeon E5607, at 2.26 GHz.

18. Backpropagation using GPU in CUDA [18]

Description: In forward propagation, each of the neurons calculates its output value based on inputs provided by the output values of the neurons that feed it. For each neural network, there is a CUDA function handling the parallel computation of neuron values of that level.

Results: The GPU gain about 7 times speedup than CPU. Calculation of backpropagation on CPU takes several hours while that on GPU in CUDA the computation time is greatly reduced.

Type of GPU: NVIDIA GeForce GTX9800+ , GPU is installed on computer having 2.8GHz Core 2 Duo CPU E7400.

19. Fuzzy-Rough data reduction using SVD [19]

Description: This paper uses Givens Rotations method for the QR method as it is found to be appropriate for full parallel algorithm. Initially, implementation of SVD algorithm with the QR decomposition scheme is done.

Results: A comparison of GPU-parallel version was done with that of the CULA library (set of GPU-accelerated linear algebra routines utilizing NVIDIA CUDA). CULA (using *culaSgesvd* routine) is five times faster than the CPU. For matrices of size 4096×4096, a gain of 97% is obtained in GPU as compared to CULA.

Type of GPU: GPU NVIDIA QuadroK5000, having 1536 CUDA cores, 2:1 Teraflops in single precision, 4GB GDDR5 of memory size and 173 GB/s of memory bandwidth.

20. Implementation of CARMEN on GPU [20]

Description: The paper highlights the working of Adaptive optics on CARMEN architecture and its implementation on GPU using Torch. CARMEN's architecture is a multi-layer perceptron, with a single hidden layer.

Results: Loading data from the main disk is much slower than loading the information from the main RAM. For small networks, this difference could mean an increment between 3 and 5 times, that can be critical for this type of systems.

Type of GPU: Nvidia GeForce GTX, Titan X with CUDA 7.5 and CUDNN v3

21. Pattern classification utilizing Slackmin algorithm [21]

Description: Slackmin algorithm is simple efficient classification model for medium scale data. It has less complexity and fast convergence rate. To overcome the classification deficiency of cuLSlackmin algorithm and the low convergence of the serial kernelized algorithm, so the parallel implementation of algorithm is used.

Results: Proposed cuKslackmin algorithm is efficient for pattern classification in low cost GPUs, since its PPR index is lower than the rest algorithm.

Type of GPU: Nvidia GT440 GPU card. GPU cache memory is configured as 48KB shared memory.

22. SVM Performance Using Large Datasets for Multi-Class Machine Learning Problem [22]

Description: In this parallel implementation is applied because it uses high computational resources and more time due to high complexity.

Results: It gains speed up to 1.41X or 29% performance improvement. Larger datasets help SVM algorithm to get solution near to the optimal solution.

Type of GPU: NVIDIA GeForce GT 720M with 96 built cores. Each core at 775 MHz.

23. Unsupervised Feature Learning Classification With Radial Basis Function Extreme Learning Machine [23]

Description: Unsupervised Feature Learning(UFL) is working better than manually designed ones in tasks such as image classification and object recognition. UFL is has high complexity and slow convergence rate.

Results: Test on MNIST shows that it is better than several art algorithms. Tests on CIFAR-10 shows that it is 20 times faster than CPU version.

Type of GPU: GPU NVIDIA tesla M2075 with 6GB RAM, 448 cores clocked at 1.5 GHz, 32KB shared memory.

24. Computer vision deep learning algorithm [24]

Description: Deep neural network methods have successfully solved many computer vision tasks, machine translation and speech generation problems and even doing it better than humans. GPUs is used to implement them to reduce their time take and increase their potential.

Results: GPUS reduces the number of nodes need to solve deep learning algorithms.

Type of GPU: 2 NVIDIA Kepler K80 dual GPU cards is associated with each node.

25. Deep Learning Classifier for Electrical Network Images Identification [25]

Description: The proposed automatic diagnosis system is used in conducting inspections in electrical power distribution lines instead of manual inspection using image processing algorithms. JSEG segmentation algorithm is done before image identification to improve the performance of deep learning classifier.

Results: In this paper, it is shown that JSEG algorithm will improve the performance of DL network in the identification of thermographic images in pseudo-colour and with GPU and CUDA technologies a fast JSEG method can be implemented.

NVIDIA GeForce GTX 1080 GPU.

26. Classification accuracy of film defects [26]

Description: Visual inspection is one of the main technologies for non-invasive inspection of products. Machine learning techniques are used for classification of defect images. Detecting and classification of defects of industrial products are important for producers to avoid shipping of defective products. They combined the deep neural network with random forest classifier to improve the performance. We also used GPU to accelerate the classification process.

Results: Random forest classification has accuracy of 94.7% and deep convolutional network has precision of 95.8%. by using both of them at same time, the overall precision of 97.1%.

27. Optimized Deep Belief Networks [27]

Description: Deep belief networks (DBNs) are multi-layer neural networks that are based on restricted Boltzmann machines (RBMs). DBNs train each layer in deep learning models and have been applied in various applications. CPUs cannot be achieved the fine grain parallelism in DBN algorithms, which are dominated by massive concurrent operations.

Results: GPU application attains a performance speedup ranging from 14 to 22 in the pre-training process and 12 to 33 times faster in the fine-tuning process compared with conventional CPU methods.

Type of GPU: NVIDIA Tesla K40c.

28. Machine Learning Algorithm for Character Recognition [28]

Description: Pattern recognition is field of machine learning. It is in high demand because of digitalization. It is important to improve the correctness and computational speed of recognition rate. GPUs helps in increasing the computational speed. In this paper, a k-nearest neighbour algorithm is implemented on GPU.

Results: The accuracy was better as the value of k and test dataset increases. The speedup was better as the value of k and test dataset decreases.

Type of GPU: NVIDIA GeForce 940MX with 384 processing cores.

29. Neural Network Algorithm for Handwritten Digit Recognition [29]

Description: Neural Network algorithms have wide use in computing fields. In this method Neural Network algorithm help us to identify the handwritten digit. These algorithms need large computational effort for progress.

Results: The GPU implementation has improvement of 5.96X in training and 8.76X in recognition rate than CPU.

Type of GPU: NVIDIA Graphics card 9600GT

30. Single Kernel Soft Synchronization Technique for Task Arrays [30]

Description: A task array is a 2-dimensional array of tasks with dependency relations. Each task uses the resulting values of some tasks in the left columns, and so it can be started only after these left tasks are completed. CUDA kernel call has a certain overhead, and the running time of a CUDA kernel is determined by a CUDA block that terminates lastly.

Results: SKSS implementation is 1.29-2.11 times faster for the 0-1 knapsack problem, 1.08-1.56 times faster for the summed area table computation, and 1.61-2.11 times faster for the error Diffusion.

Type of GPU: NVIDIA TITAN X. which has 28 streaming multiprocessors with 128 processor cores, 2048 resident threads, 96K-byte shared memory, and 64K 32-bit registers.

V. CONCLUSION

After going through the several literatures, we have concluded that GPUs definitely can be used to speed up other types of computations of different soft computing techniques. GPU computing is all around leveraging the parallelism in the GPU architecture to perform mathematically compute rigorous operations that the CPU is not designed well to handle. A CPU is designed for task switching. It is not designed for high throughput. It needs to switch between multiple tasks and store contexts while switching and take care of the shared memory. It is designed for low throughput and high latency jobs. On the other hand, a GPU is designed for heavy workload and throughput. It is designed for low latency and high throughput jobs. Several researchers and scientists around the world are using GPUs in several soft computing techniques. However, we have concluded deep learning is the field where GPU and CUDA can be fully utilized because deep learning is a field with intense computational requirements and the choice of your GPU will fundamentally determine your deep learning experience. With a good, solid GPU, one can quickly iterate over designs and parameters of deep networks, and run experiments in days instead of months, hours instead of days, minutes instead of hours.

VI. REFERENCES

[1] L Wang, X Zhu, B Yang, J Guo, S Liu, M Li, J Zhu , "A Abraham, Accelerating nearest neighbour partitioning neural network classifier based on CUDA", Engineering Applications of Artificial Intelligence, Vol. 68, 2018, pp. 53–62, doi.org/10.1016/j.engappai.2017.10.023.

[2] I M. Coelho, V N. Coelho, E J. da S. Luz, L S. Ochi, F G. Guimarães, E Rios, "A GPU deep learning metaheuristic based model for time series Forecasting", Applied Energy, Vol.201, 2017, pp.53-62 doi.org/10.1016/j.apenergy.2017.01.003.

[3] D I Patrícioa, R Riederb, "Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review", Computers

and Electronics in Agriculture, Vol. 153, 2018, pp. 69–81 doi.org/10.1016/j.apenergy.2017.01.003.

[4] D Akgün, P Erdoğmuş, "GPU accelerated training of image convolution filter weights using genetic algorithms", Applied Soft Computing, Vol. 30, 2015, pp. 585–594doi.org/10.1016/j.asoc.2015.02.010.

[5] M A. Franco, J Bacardit, "Large-scale experimental evaluation of GPU strategies for evolutionary machine learning", Information Sciences, Vol. 330, 2016, pp. 385-402, doi.org/10.1016/j.ins.2015.10.025.

[6] R.E. González, R.P. Muñoz, C.A. Hernández, "Galaxy detection and identification using deep learning and data augmentation", Astronomy and Computing, Vol. 25, 2018, pp. 103–109, doi.org/10.1016/j.ascom.2018.09.004.

[7] Q Li, R Salman, E Test, R Strack, V Kecman, "Parallel multitask cross validation for Support Vector Machine using GPU", Journal of Parallel and Distributed Computing, Vol. 73, 2013, pp. 293-302, doi.org/10.1016/j.jpdc.2012.02.011.

[8] N Lopes, B Ribeiro, "Towards adaptive learning with improved convergence of deep belief networks on graphics processing units", Pattern Recognition, Vol. 47, 2014, pp. 114–127, doi.org/10.1016/j.patcog.2013.06.029.

[9] Y Liu, L Wu, "High Performance Geological Disaster Recognition using Deep Learning", Procedia Computer Science, Vol. 139, 2018, pp. 529–536, doi.org/10.1016/j.procs.2018.10.237.

[10] A B.S. Serapião, G S. Corrêa, F B. Gonçalves, V O. Carvalho, "Combining K-Means and K-Harmonic with Fish School Search Algorithm for Data Clustering Task on Graphics Processing Units", Applied Soft Computing, Vol. 41, 2016, pp. 290-304, doi.org/10.1016/j.asoc.2015.12.032.

[11] M. Martínez-Zarzuela, F. J. Díaz Pernas, A. Tejero de Pablos, M. Antón Rodríguez, J. F. Díez Higuera, D. Boto Giralda, D. González Ortega, "Adaptive Resonance Theory Fuzzy Networks Parallel Computation Using CUDA", IWANN 2009: Bio-Inspired Systems: Computational and Ambient Intelligence, 2009, pp. 149-156, doi.org/10.1007/978-3-642-02478-8_19

[12] M A. Franco , N Krasnogor , J Bacardit, "Speeding up the evaluation of evolutionary learning systems using GPGPUs", GECCO '10, Proceedings of the 12th annual conference on Genetic and evolutionary computation, 2010, pp.1039-1046, doi.org/10.1145/1830483.1830672

[13] D S Chevitarese, D Szwarcman, M Vellasco, "Speeding Up the Training of Neural Networks with CUDA Technology", ICAISC 2012: Artificial Intelligence and Soft Computing, 2010, pp. 30-38, doi.org/10.1007/978-3-642-29347-4_4

[14] A Guzhva, S Dolenko, I Persiantsev, "Multifold Acceleration of Neural Network Computations Using GPU", Artificial Neural Network - ICANN 2009, pp. 373-380, doi.org/10.1007/978-3-642-04274-4_39

[15] K Sopyła, P Drozda, P Górecki, "SVM with CUDA Accelerated Kernels for Big Sparse Problems", ICAISC 2012: Artificial Intelligence and Soft Computing, 2012, pp. 439-447, doi.org/10.1007/978-3-642-29347-4_51

[16] M. Martínez-Zarzuela, F. J. Díaz Pernas, A. Tejero de Pablos, M. Antón Rodríguez, F. Perozo-Rondón, D. González Ortega, "Fuzzy ARTMAP Based Neural Networks on the GPU for High-Performance Pattern Recognition", IWINAC 2011: New Challenges on Bioinspired Applications, 2011, pp. 343-352, doi.org/10.1007/978-3-642-21326-7_37

[17] M Pavlech, "Self-organizing Migration Algorithm on GPU with CUDA", Soft Computing Models in Industrial and Environmental Applications, 2013, pp. 173-182, doi.org/10.1007/978-3-642-32922-7_18

[18] J Liu, L Guo, "Implementation of Neural Network Backpropagation in CUDA", Intelligence Computation and Evolutionary Computation, 2013, pp. 1021-1027, doi.org/10.1007/978-3-642-31656-2_140

[19] S Cuomo, A Galletti, L Marcellino, G Navarra, G Toraldo, "On GPU–CUDA as preprocessing of fuzzy-rough data reduction by means of singular value decomposition", Springer Berlin Heidelberg, Soft Computing(2018), Vol. 22, 2012, pp. 1525-1532, doi.org/10.1007/s00500-017-2887-x

[20] C González-Gutiérrez, J D Santos-Rodrígue, R Á F Díaz, J L C Rolle, N R Gutiérrez, F J de Cos Juez, "Using GPUs to Speed up a Tomographic Reconstructor Based on Machine Learning", International Joint Conference SOCO'16-CISIS'16-ICEUTE'16, 2016, pp. 279-289, doi.org/10.1007/978-3-319-47364-2_27

[21] G.A. Papakostas, K.I.Diamantaras, T. Papadimitriou, "Parallel pattern classification utilizing GPU- based kernelized Slackmin algorithm", Journal of Parallel and Distributed Computing, Vol. 99, January 2017, pp. 90-99, doi.org/10.1016/j.jpdc.2016.09.001

[22] M A Hay Bin Sulaiman, A Suliman, A Rahim Ahmad, "Measuring GPU-Accelerated Parallel SVM Performance Using Large Datasets for Multi-Class Machine Learning Problem", In Information Technology and Multimedia (ICIMU), 2014 International Conference on IEEE, pp. 299-302, doi.org/10.1109/ICIMU.2014.7066648

[23] D Lam, D Wunsch, "Unsupervised Feature Learning Classification With Radial Basis Function Extreme Learning Machine Using Graphic Processors", IEEE Transactions on Cybernetics , Vol: 47 , Issue: 1 , Jan. 2017, pp. 224 - 231, doi.org/10.1109/TCYB.2015.2511149

[24] V Campos, F Sastre, M Yagues, M Bellver, J Torres, " Distributed training strategies for a computer vision deep learning algorithm on a distributed GPU cluster", Procedia Computer Science, Vol: 108, 2017, pp. 315-324, doi.org/10.1016/j.procs.2017.05.074

[25] F Fambrini, Y Iano, D Gará Caetano, "GPU Cuda JSEG Segmentation Algorithm associated with Deep Learning Classifier for Electrical Network Images Identification", Procedia Computer Science , Vol: 126, 2018, pp. 557-565, doi.org/10.1016/j.procs.2018.07.290 om

[26] H Ando, Y Niitsu, M Hirasawa, H Teduka, M Yajima, "Improvements of classification accuracy of film defects by using GPU-accelerated image processing and machine learning frameworks", International Journal of Latest Trends in Engineering and Technology, Special Issue SACAIM 2016, pp. 425-428 , doi.org/10.1109/NicoInt.2016.15

[27] T Li, Y Dou, L Liang, Y Wang, Q Lv, "Optimized Deep Belief Networks on CUDA GPUs", 2015 International Joint Conference on Neural Networks (IJCNN), 2015, pp. 1-8 ,doi.org/10.1109/IJCNN.2015.7280511

[28] M N Mayekar, S Kuwelkar, "Implementation Of Machine Learning Algorithm For Character Recognition On GPU", In Computing Methodologies and Communication (ICCMC), 2017 International Conference on. IEEE, 2017, pp. 470-474, doi.org/10.1109/ICCMC.2017.8282734

[29] P. Yu. Izotova, N. L. Kazanskiya, D. L. Golovashkina, S. V. Sukhanov, "CUDA Enabled Implementation of a Neural Network Algorithm for Handwritten Digit Recognition", Optical Memory and Neural Networks June 2011, 2011, Vol: 20, Issue 2, pp. 98–106, doi.org/10.3103/S1060992X11020032

[30] S Funasaka, K Nakano, Y Ito, "Single Kernel Soft Synchronization Technique for Task Arrays on CUDA-enabled GPUs, with Applications", 2017 Fifth International Symposium on Computing and Networking (CANDAR) , 2017, pp.11-20, doi.org/10.1109/CANDAR.2017.35

[31] Xiaolu Zhanga, Zeshui Xu, "Soft computing based on maximizing consensus and fuzzy TOPSIS approach to interval-valued intuitionistic fuzzy group decision making", Applied Soft Computing, 2014, pp. 42-56, doi.org/10.1016/j.asoc.2014.08.073

[32] Mohammad Hemmat Esfe, Mohammad Reza Hassani Ahangar, Mousa Rejvani, Davood Toghraie, Mohammad Hadi Hajmohammad, "Designing an artificial neural network to predict dynamic viscosity of aqueous nanofluid of TiO2 using experimental data", International Communications in Heat and Mass Transfer, 2016, pp. 192-196, doi.org/10.1016/j.icheatmasstransfer.2016.04.002