Implementation of Deep Neural Nets on Microcontrollers for Speech Recognition

Fazal Noor[1†] and Hatem M. El-Boghdadi[2††],

Faculty of Computer and Information Systems, Islamic University of Madinah, Madinah, Saudi Arabia

Abstract
Nature inspired optimization methods and Deep Learning Neural Networks have been proliferated in the literature. The optimization algorithms have been used in variety of problems such as controls, search, estimation, and many other areas whereas the Neural Networks have mainly been used in prediction and classification's problems. In this work, Neural networks with heuristics optimization algorithms are employed to compute the optimum weights. In particular, a Parallel Distributed Bat Algorithm (PDBA) is used to obtain optimum weights of Deep Learning Neural Networks. The algorithms are implemented on an Arduino, an open source microcontroller and an application in control systems is studied using speech recognition. As the neural network process is compute intensive, multiple microcontrollers are proposed in a master-slave configuration. Speech is the fastest way for communications between humans, and extensive research has been done in speech recognition between humans and machines. In this research, Linear Predictive Coding (LPC) is used for extracting speech features and Deep Learning Neural Networks are trained with speech feature samples in a voice-controlled application.

.Keywords:
Robotics, Speech recognition, Bat Optimization Algorithm, Neural Networks,.

1. Introduction

Many real-world optimization problems in the areas of controls, chemistry, biology, engineering, and many other fields require large amounts of computations and simulations. The problem may be formulated as a continuous function and nature inspired optimization methods utilized to obtain optimal solutions. Bat algorithm (BA) is a heuristic optimization algorithm and has been reported to be efficient in providing optimal solutions to continuous nonlinear constrained problems. In cases where the search space is extremely large a PC cluster is useful for large computations [1]. A PC cluster consists of off the shelf machines connected to a fast Ethernet switch. A PC cluster can be thought of as an affordable supercomputer.

Manuscript received April 5, 2025
Manuscript revised April 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.4.5

Super computers are used in many parallel computing applications to solve very complex problems [2]. In this research, there are several objectives and one is to use a parallel bat algorithm for finding the optimum weights of a Neural Network. Warran S. McCulloch and Walter Pitts in 1943 developed a model of an artificial neural neuron with inputs, processor and generation of an output [5]. Since then many researchers have used neural networks in numerous applications, such as aerospace, automobile, military, electronics, financial, industrial, medical, speech, telecommunications, transportation, software, control, signal processing and others. In general, a neural network is an adaptive system where its internal structure consists of neurons (having activation functions) connecting to other neurons by links and its weights are adjusted based on error performance at the output. Once the desired output behaviour is achieved the neural system is said to have been trained and ready for testing it on unseen input data.

2. Literature Review

Many heuristic optimization algorithms appearing in literature were devised by observing nature. These algorithms start with a random population as an input and a fitness function is used for guidance to an optimum solution. The solution need not be exact but close to it and that it gives satisfactory result. On the other hand, Neural Networks are inherently parallel and have been used in numerous applications. Neural Networks are mainly used for prediction or classification type of problems. Incorporating the heuristic optimization algorithms with neural networks were done long time ago. However, with advancement in computational power of processors and reduction in cost of equipment more research is being done with applications in numerous areas, to mention a few are cancer cell detection, speech recognition and drug side effect prediction and/or classification are application areas demanding intensive computations. With open source microcontrollers available on the market, it becomes interesting to implement Neural networks on them and study their performance. In our work, the aim is to implement a neural network on a microcontroller and use nature inspired optimization Bat algorithm to find the optimum weights of the neural network. Our application target is speech recognition using a neural network.

In deep learning the model learns to perform classification directly for example from sound or images, etc. Deep learning is a neural network architecture with many hidden layers which can run in hundreds of layers. Some examples of deep learning are the following: an autonomous vehicle stops or slows down at a pedestrian crosswalk as in fig 1, as counterfeit bank note is put in an ATM machine it calls the security and the culprit is apprehended as in fig 2, another example is a smartphone application provides translation of a foreign street sign instantly. Deep learning is well suited for identification type of problems such as face recognition, speech recognition, sign recognition, etc.

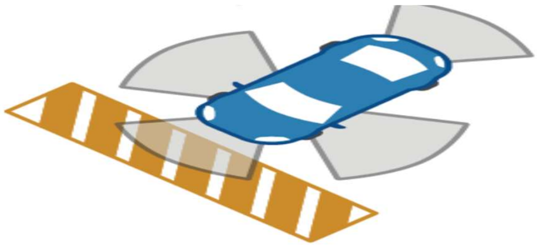

Figure 1 Autonomous car coming to stop at pedestrian crosswalk [3].

Figure 2 ATM machine rejects a counterfeit note [3].

Deep learning neural networks being used in cancer cell detection as in figure 3 below.

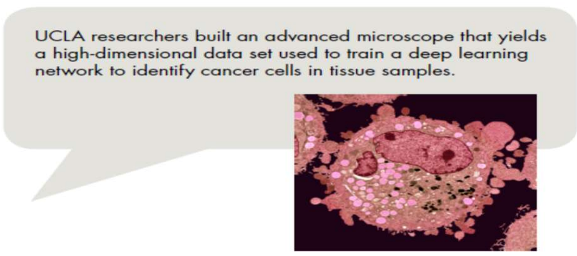

Figure 3 Cancer cell identification [2].

Object recognition have been possible with fairly high accuracy. There are data sets such ImageNet which are available for free and can used to train the neural networks. Fig 4. below shows a neural network trained on images is being able to identify which type of vehicle it is car, truck, … and bicycle.

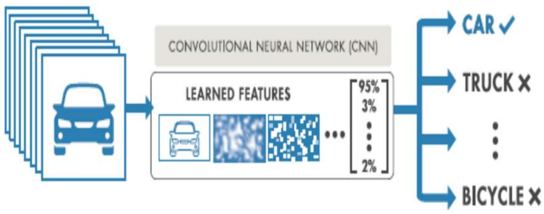

Figure 4 Convolutional neural network [5].

With increased computing power it is possible to accelerate training of huge amounts of data and reducing training time from months or weeks, to days or hours. AlexNet is being used to perform new recognition tasks and was trained with 1.3 million images of high resolution and is capable of recognizing 1000 different objects. Figure 5 shows a deep neural network can be developed for nonlinear processing using simple elements or processors operating in parallel. The human brain is similar to it. The deep neural network consists of an input layer, many hidden layers, and an output layer. The layers are interconnected via nodes (called neurons). The network learns from the inputs, we do not have any idea what features the network learns itself.

Suppose an input image is given as shown in figure 5 below it goes through Convolutional network and output is classified indicating the image contains a flower, a cup, car, and a tree.

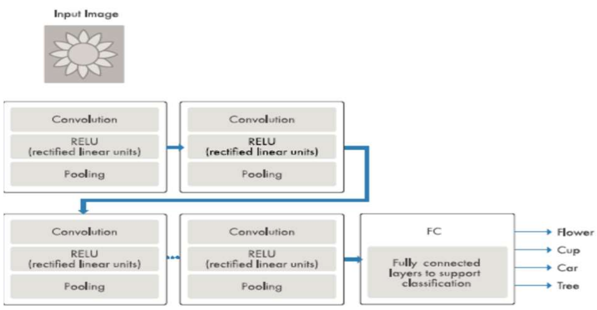

Figure 5 Convolutional NN for classification [10].

Convolutional neural networks (CNN) are one of most popular algorithms for deep learning with images and video. The input is first convolved, then pooled, and then sent through a rectified linear unit as shown below in Fig 6. The 3 operations are repeated over and over again with tens or hundreds of layers and each layer learns new different features. After feature detection the 2nd phase is the classification as shown in figure below. The final layer is using a softmax function for the classification output.

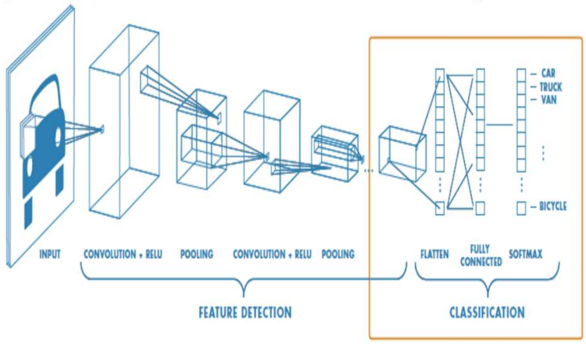

Figure 6 First part is feature detection and Second part consists of classification [11].

There is a difference between deep learning and machine learning as seen in fig 7. Deep learning is a type of machine learning. In machine learning the features are manually extracted for example from an image. Whereas with deep learning, the images are fed into a neural network and features are learned automatically. For deep learning the learning phase may require hundreds of thousands or up to millions of images for better results. Since this phase is computationally intensive it is better to have extremely fast CPUs, or GPUs or PC clusters, etc.

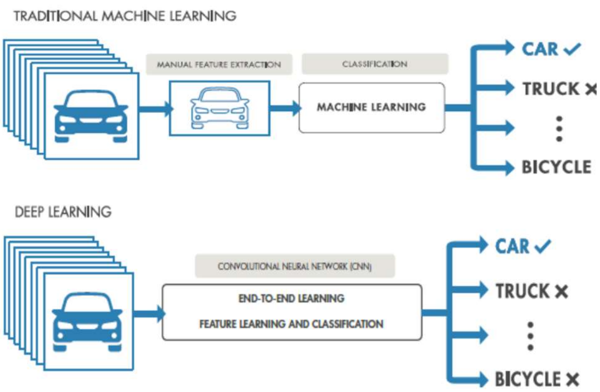

Figure 7 Traditional machine learning and bottom using deep learning [11].

The table below shows the relations between machine learning and deep learning.

Table 1: Comparison between machine learning and deep learning.

Machine Learning	Deep Learning
+ Good results with small data sets	– Requires very large data sets
+ Quick to train a model	– Computationally intensive
– Need to try different features and classifiers to achieve best results	+ Learns features and classifiers automatically
– Accuracy plateaus	+ Accuracy is unlimited

As an example of Deep learning is the AlexNet which is trained using more than a million images and is commonly used for image classification. It can classify images in a thousand different classes. Figure 8 below is a sample of AlexNet's classifications.

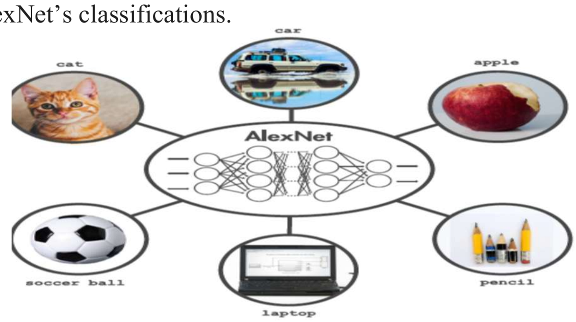

Figure 8 AlexNet trained with over million images and can classify 1000 objects [12].

Deep learning requires intensive computations and therefore the selection of resources is critical as it can take hours, weeks, months or even years to obtain results due to huge amounts of data and depending on computational power. For computation there are several options, CPU based, GPU base, PC cluster based, and cloud based. CPU based is best for simple pretrained Neural Networks and readily available option. A Graphical Processing Unit (GPU) can reduce network training time from weeks to days or days to hours depending on the data and problem at hand. Multiple GPUs are often used to further speed up the processing time. A PC cluster based is a good option as if you're getting a supercomputer at an affordable price. It is useful in applications which can be run in parallel and demands less communication time compared to computation time. The Cloud based is best as you don't have purchase any equipment nor have to setup or configure the hardware. One example of practical study is in Cancer diagnostics with Deep Learning and Photonic time stretch. Cancer patients usually receive chemotherapy and must on a regular basis undergo CT and PET scans to check on the progress of treatment. A method called Flow cytometry is used to identify tumor cells flowing in the blood and is considered a game-changer in cancer treatment. However, in this method extremely high volumes of data is generated in fact 100 gigabytes of data per second. For a single experiment, in which every cell in a 10-milliliter blood sample is imaged at almost 100,000 cells per second, the system generates from 10 to 50 terabytes of data [2]. Figure 9 below shows a block diagram of cancer cell detection at UCLA photonics lab. As it involves big data, it takes more than a week to complete image processing and machine learning process. They used parallel computing with 16 processors to accelerate and reduce the time required to complete the analysis from a week to half a day.

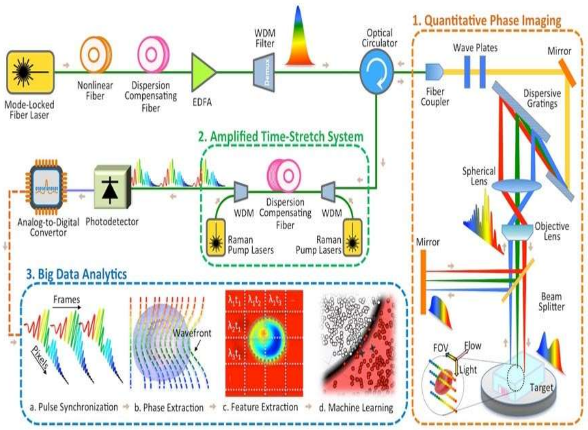

Figure 9 Blood Analysis for cancerous cells.

Similarly, speech recognition requires many samples of recorded voice to train a neural network.

3. Optimization Method – Bat Algorithm

Xin-She Yang in 2010, developed a new meta-heuristic optimization algorithm based on Bat's echolocation behavior. The bats transmit pulses of sound waves and listen to the reflected echoes of sound waves. From the reflected waves the bats are able to differentiate between prey and surroundings. The bat algorithm developed by Yang depends on the following three assumptions,

1. The bats use echolocation to calculate distance and be able to differentiate between prey and objects in the background.

2. Each bat is flying randomly with velocity at position x and with a frequency f. The frequency may have varying wavelength lambda, and loudness A0 to search for prey.

3. Assumption that loudness changes from a large positive value for A0 to a minimum constant value Amin.

The Bat's algorithm is based on the hunting behavior of bats. Bats use echolocation to detect prey, avoid obstacles and locate their resting location. The bats emit high-pitched sounds and interpret their echoes to determine the distance and direction of targets. The Bat algorithm has the following main features, automatic zooming via loudness and pulse emission rates, parameter control, frequency tuning. It has the following advantages, ability to solve efficiently wide range of nonlinear optimization problems with optimal solutions. The Bat algorithm has proven to provide solutions in a variety of applications, namely, Engineering design, Protein Structure prediction, Classification of genes, PID controllers, and Neural Networks. In this research, we use the Parallel Distributed Bat Algorithm (PDBA) in obtaining the optimum weights of a Deep Neural Network [10], The parallel bat algorithm is based on the inherent behavior of each bat echolocation. Every bat is flying independently with its own frequency, velocity, and location. Therefore, a straight forward parallel method would be to have number of processors equal to the number of flying bats. Each processor executes in a parallel and distributed fashion. However, when there are limited number of processors, then a group of X bats or solutions will be assigned to each processor. In a parallel distributed algorithm, each worker node works on a split population (i.e. solutions) depending on the size of the cluster. The parallel-distributed bat algorithm using the Master-Worker model is summarized as follows:

Parallel Models: Master-Slave Model

Parallel-Distributed Bat Algorithm – Master – Worker (PDBA-MW)

Initialize every_send_time to 50

For (iteration = 1 to Max_iteration)

IF (Worker Node) then

For (i=1 to newpopsize) Note: newpopsize is N / total_workers

Perform Steps 1 to 5 of sequential algorithm

End

If (iterations equals every_send_time) then

1. Each worker_k sends its best candidate solution to the Master.
2. Receive the best from Master node.

Else if (Master Node) then

For (k = 1 to Total_Workers)

1. Receive best_k from each worker.
2. Compare and choose best among them.
3. Send to each worker the best.

ENDIF

The process is terminated when desired accuracy is achieved or maximum number of iterations has been reached.

In the PDBA the master may be used to perform primarily the exploration part and workers primarily perform the exploitation part more than exploration of the search space. Other scenarios are possible. The main objective in the PD method is to reduce the communication time as much as possible compared to the computation time.

4. Artificial Neural Networks

Artificial Neural networks (ANN) are inspired by the biology of human brain, which consists of neurons connected to other neurons via axons. It has been shown human brain consists of 10^11 neurons. An example of how neurons are connected via axons and dendrites are shown in the figure 10 below. The dendrites act as inputs receiving signals from the sensors as electrical impulses and are processed by neurons.

In an ANN there are several layers of nodes and each node has an activation function which is computed and its output is passed to next layer of neurons connected to it. Each link connecting the nodes have a weight value associated to it.

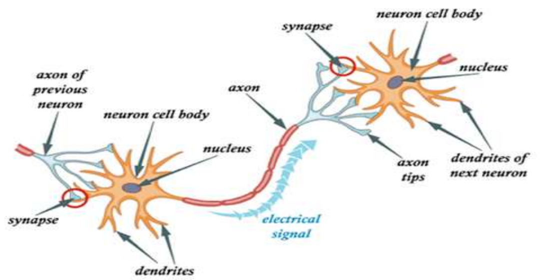

Figure 10 Biology of Human Brain cells.

A Neural network consists of 2 topologies, a feedforward and a feedback. In a feedforward NN as the name implies flow is one way without any feedback loop. These networks are beneficial in pattern generation, pattern classification, and pattern recognition problems.

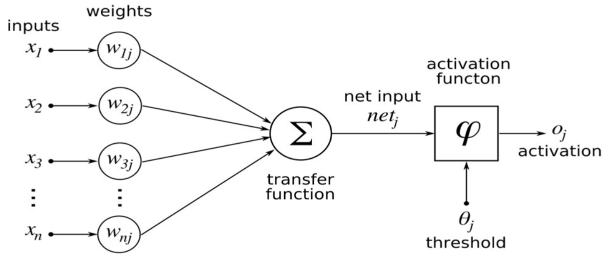

Figure 11 Neuron with Activation function.

Table 2: Common Activation Functions used in Neural Networks.

Name	Plot	Equation	Derivative (with respect to x)	Range
Identity		$f(x)=x$	$f'(x)=1$	$(-\infty,\infty)$
Binary step		$f(x)=\begin{cases}0 & \text{for } x<0\\ 1 & \text{for } x\ge 0\end{cases}$	$f'(x)=\begin{cases}0 & \text{for } x\ne 0\\ ? & \text{for } x=0\end{cases}$	$\{0,1\}$
Logistic (a.k.a. Sigmoid or Soft step)		$f(x)=\sigma(x)=\frac{1}{1+e^{-x}}$ [1]	$f'(x)=f(x)(1-f(x))$	$(0,1)$
TanH		$f(x)=\tanh(x)=\frac{(e^{x}-e^{-x})}{(e^{x}+e^{-x})}$	$f'(x)=1-f(x)^2$	$(-1,1)$
ArcTan		$f(x)=\tan^{-1}(x)$	$f'(x)=\frac{1}{x^2+1}$	$\left(-\frac{\pi}{2},\frac{\pi}{2}\right)$
ElliotSig[9][10][11] Softsign[12][13]		$f(x)=\frac{x}{1+\lvert x\rvert}$	$f'(x)=\frac{1}{(1+\lvert x\rvert)^2}$	$(-1,1)$
Inverse square root unit (ISRU)[14]		$f(x)=\frac{x}{\sqrt{1+\alpha x^2}}$	$f'(x)=\left(\frac{1}{\sqrt{1+\alpha x^2}}\right)^3$	$\left(-\frac{1}{\sqrt{\alpha}},\frac{1}{\sqrt{\alpha}}\right)$
Inverse square root linear unit (ISRLU)[14]		$f(x)=\begin{cases}\frac{x}{\sqrt{1+\alpha x^2}} & \text{for } x<0\\ x & \text{for } x\ge 0\end{cases}$	$f'(x)=\begin{cases}\left(\frac{1}{\sqrt{1+\alpha x^2}}\right)^3 & \text{for } x<0\\ 1 & \text{for } x>0\end{cases}$	$\left(-\frac{1}{\sqrt{\alpha}},\infty\right)$

Neurons are connected in each layer and weights assigned to the links with an arrow indicating the flow of information, see figure below. A function at each neuron is then computed with inputs and weights multiplied and summed. The network is trained with data and desired outputs, in the training phase the weights are adjusted incrementally such the desired output is achieved. There are alternative methods to make the deep learning neural networks learn, namely, reinforcement learning, supervised learning, and unsupervised learning. In reinforcement learning, Neural network observes the environment, makes a decision, and weights are adjusted if the observation is not correct. Supervised learning consists of an instructor type guiding the NN with the correct answers and adjustments to the weights are done by observing the loss function result. In unsupervised learning, no instructor is used to guide the NN with an example data set.

In the feedback topology neural network usually uses a back-propagation method with the feedforward part. In the back-propagation method weights are adjusted according to derivatives of a loss function. The main problem with this method is convergence to local optimum. Figure below shows the algorithm and a block diagram of back-propagation method.

<u>Algorithm for Deep Learning Neural Networks:</u>

1. Initialization; Set weights matrix to random numbers.
2. Forward propagate by using weights and activation functions.
3. Error Function: Compute the error function sum of squares of absolute error
4. Optimize weights using differentiation of the loss error function or use optimization technique.
5. Perform back-propagation of errors to the hidden layers.
6. Update the weights.
7. Iterate steps 2 to 6 until convergenc

The main objective is to design a NN for prediction of spoken words in control systems as in Fig 12. Let F_{jk}^{L} be the LPC feature inputs, where L is the NN layer number, j represents data and k represents input. For Deep Learning NN, the number of hidden layers are greater than two.

$$F^1 = \begin{bmatrix} f_{11}^1 & f_{12}^1 \\ \vdots & \vdots \\ f_{N1}^1 & f_{N2}^1 \end{bmatrix} \tag{1}$$

The weight matrix at layer 1 is represented by W^1, where the notation W_{qr} represents the link weight from node q to node r.

$$W^1 = \begin{bmatrix} W_{11} & \cdots & W_{14} \\ \vdots & \ddots & \vdots \\ W_{21} & \cdots & W_{24} \end{bmatrix} \tag{2}$$

Let H be the value matrix obtained by multiplying the input features F by the weights W matrix

$$H_1^1 = F\,W$$

$$H_1^1 = [Fi1W11 + F1W21 \quad Fi2W12 + Fi1W22 \quad Fi1W13 + Fi2W23 \quad Fi1W14 + Fi2W24] \tag{3}$$

Let A(v) be the activation function matrix, where the sigmoid function is used at each node, $s(v) = 1/(1 + e^{-v})$ [9]

$$S^1(v) = \begin{bmatrix} s_1(v1) & s_1(v2) & \cdots & s_1(v4) \\ \vdots & & \ddots & \vdots \\ s_N(v1) & s_N(v2) & \cdots & s_N(v4) \end{bmatrix} \tag{4}$$

Let G^2 be the output obtained by function matrix S(v) multiplied by the weight matrix W^2 at layer 2.

$$G^2 = S^2(v)\,W^2 = \begin{bmatrix} G_{11}^2 & G_{12}^2 & \cdots & G_{14}^2 \\ \vdots & & \ddots & \vdots \\ G_{N1}^2 & G_{N2}^2 & \cdots & G_{N4}^2 \end{bmatrix} \tag{5}$$

where W^2 is the weight matrix at layer 2.

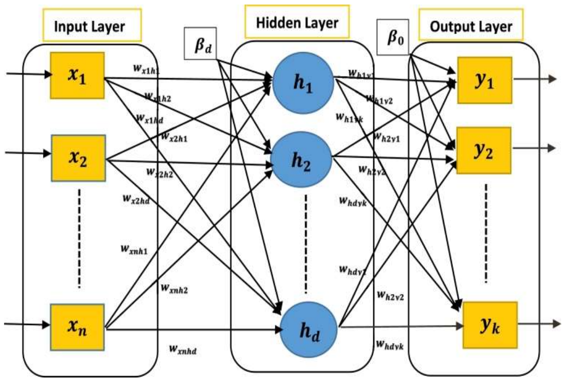

Figure 12 Diagram of a Neural Network with N inputs, and K outputs.

The performance of the neural network may use the squared error function

$$Error = -\sum_{i=1}^{4} 0.5\, x\, (target_i - actual_i)^2 \tag{6}$$

or the cross entropy.

$$CE = CE(target, output) = -\sum_{i,z=1}^{i,z=4} t_{iz} \log(o_{iz}) \tag{7}$$

where t is the target and o is the actual output of trained neural network.

The optimal weights in the backpropagation method are computed by the gradient descent of total error, Error, as given below,

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial z_k}\; x\; \frac{\partial z_k}{\partial f}\; x\; \frac{\partial f}{\partial w_i} \tag{8}$$

The cross entropy, CE, is obtained by taking the partial derivative of CE with respect to the weights and is given by,

$$\frac{\partial CE}{\partial w_i} = \frac{\partial CE}{\partial z_k}\; x\; \frac{\partial z_k}{\partial f}\; x\; \frac{\partial f}{\partial w_i} \tag{9}$$

After taking partial derivatives and evaluating, the old weights are computed as following,

$$W_{new} = W_{old} - delta * \nabla_W CE \tag{10}$$

Where delta represents a small step value. The new value of weights does not guarantee it is better than the old weight, it might be necessary to re-adjust weights according to error convergence criteria.

Neural networks are inherently parallel as the connections of elements in each layer. A neural network is trained for a particular input to map to a specific target output. The output of a NN often have binary values, however when implementing a speech recognition system, the problem is that there is overlapping of classes. In such a case it is better to use a probabilistic neural network in which the output values are numbers between 0 and 1, and that the sum of outputs equals one.

There are 3 approaches to weights updating of a NN, online mode, batch mode, and stochastic mode. In the online mode, after each training sample the weights are updated. In the batch mode, the weights are updated after all samples in the training set are run. In the stochastic mode, the updates of weights are done in mini-batches.

Practical aspects of Neural Networks. In practice, the backpropagation algorithm is used to update the weights of a Deep Learning NN as it performs well in challenging problems. The training of a neural network is an art of balancing between the learning (training dataset) and generalization by checking its performance on new unseen inputs to a NN. In practice backpropagation is used as a popular method due to its simplicity, and computational efficiency. When implementing the backpropagation method, it is important to look at the following parameters; the 3 different modes of updating of weights, randomizing the input samples, initialization of weights, choice of activation functions (e.g. Sigmoid or Relu or others), choice of target representations (e.g. 0, 1 or -1 1), and choice of learning rate.

5. Application to Speech Recognition

In this section, an application to speech recognition is presented. Speech is one of the most efficient ways to communicate and convey information by humans. Now a days, speech recognition systems are used to replace input devices such as mouse and keyboards. Speech recognition is also a very desirable way to communicate with machines such as robotics.

There are many methods used for speech recognition [13]. The voice signals are sampled at 8000 directly from the microphone and the Linear Prediction Coding (LPC) is used for extracting 13 features from a voice signal see fig 13. Many samples of a particular word for neural network to recognize are collected and stored in matrices of arrays. These samples are then used to train a Deep Learning Neural Network to recognize the word spoken. After training is done and optimum weights of the Neural Network determined, the NN network is then tested to check for satisfactory results. Fig 14 shows the NN trained for different group of selected words. The Bat algorithm with the Backpropagation method were used to find the optimum weights of the Neural Networks. If the results are not satisfactory then the Deep Learning NN is re-trained. Our experimental results show the highest recognition rate achieved is 96 % and lowest is 88%. This indicates higher accuracy to 99 % is desirable for speech recognition applications.

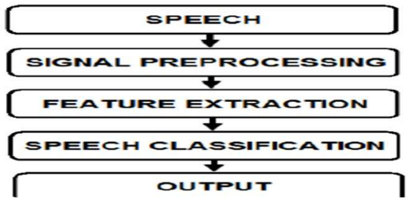

Figure 13 Speech recognition process.

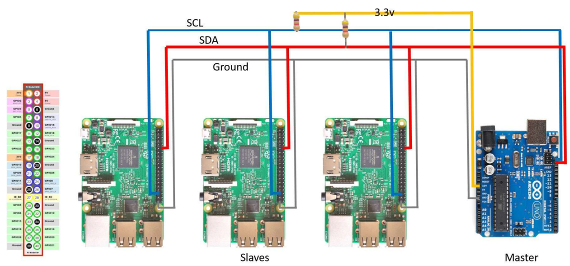

Figure 14 Each slave has a NN trained for a different group of selected words.

Speech Recognition Method

Step 1. Speak clearly the words to be trained by neural network.

a. Around 13 features of the spoken word are extracted.
b. Due to memory limitation, only 10 words were used in the experiment.

Step 2. Run the neural network with weights and calculate the outputs and error.

a. Neural network with backpropagation and evolutionary Bat Algorithm were implemented on Arduino microcontrollers

Step 3. Test the neural network on microcontrollers. Speak into the microphone the words to be tested.

The following 14 words were tested, clear, open, close, door, window, left, right, up, down, stop, go, on, off, and motor. The figures 15 to 28 below are of words tested with their values of 13 LPC features and 6 samples of each word spoken. Fig 29 show the error per iterations for N generations. Table 3, shows the percentage of time the machine recognized the words correctly.

Table 3: Percentage of time the machine recognized the words correctly.

	Trained Words	Percent Correct (%)		Trained Words	Percent Correct (%)
1	Clear	93 %	8	up	89 %
2	Open	91 %	9	down	94 %
3	Close	94 %	10	stop	93 %
4	Door	90 %	11	go	92 %
5	Window	88 %	12	on	89 %
6	left	91 %	13	off	96 %
7	right	93 %	14	motor	91 %

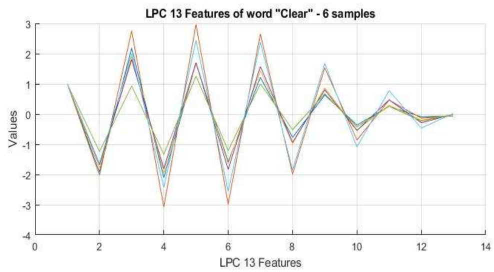

Figure 15 LPC 13 features of word Clear

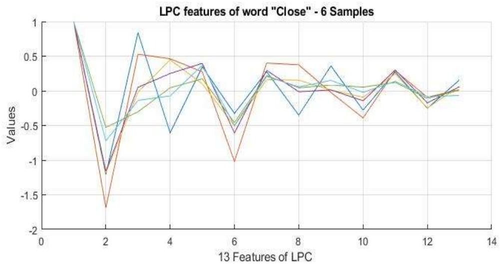

Figure 16 LPC features of word Close

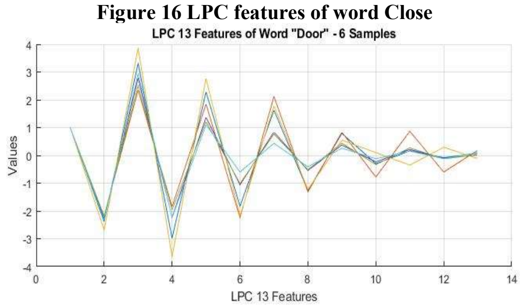

Figure 17 LPC features of word Door

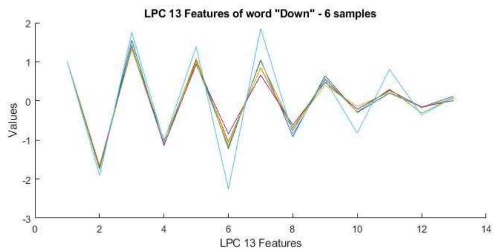

Figure 18 LPC featurs of word Down

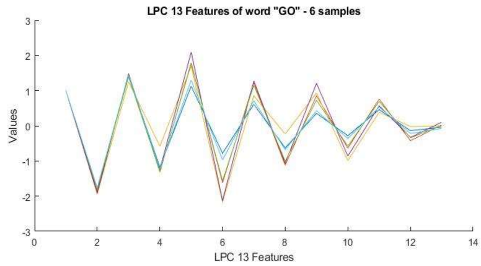

Figure 19 LPC featurs of word Go.

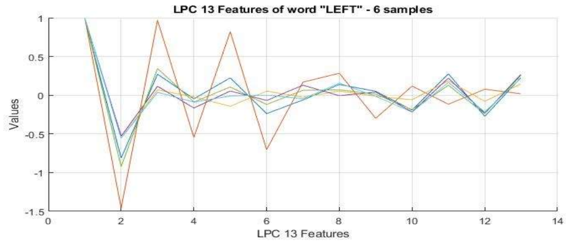

Figure 20 LPC featurs of word Left.

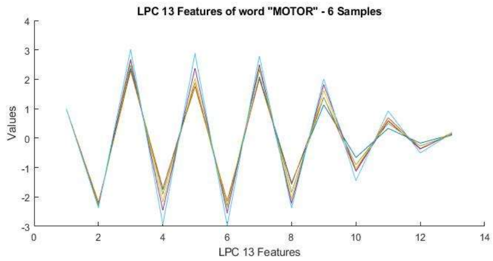

Figure 21 LPC featurs of word Motor

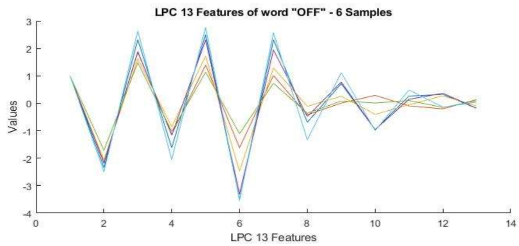

Figure 22 LPC featurs of word off.

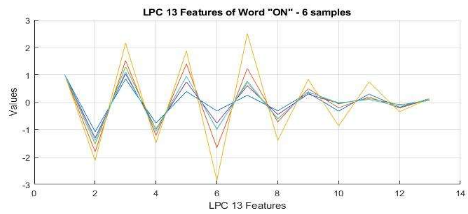

Figure 23 LPC featurs of word on.

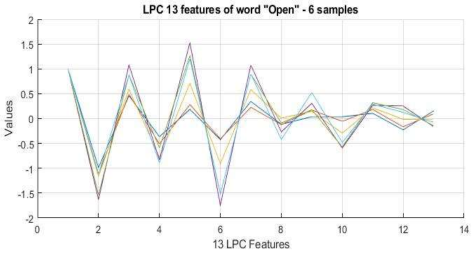

Figure 24 LPC features of word Open.

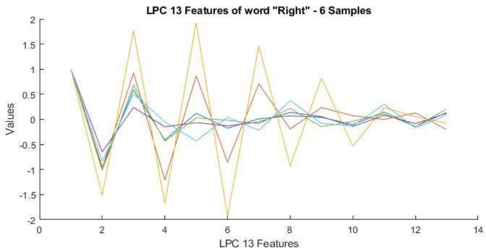

Figure 25 LPC features of word Right.

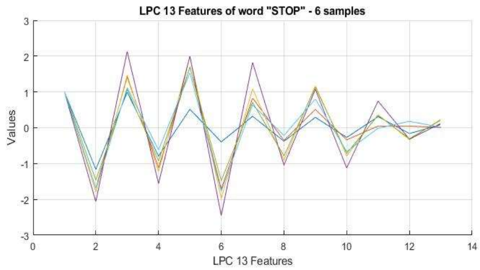

Figure 26 LPC features of word Stop.

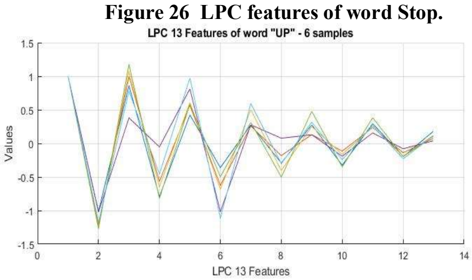

Figure 27 LPC features of word up.

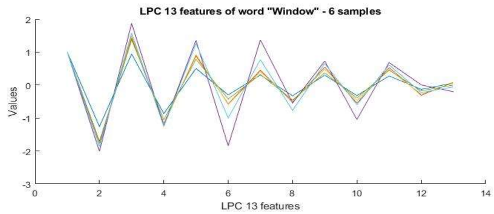

Figure 28 LPC features of word Window.

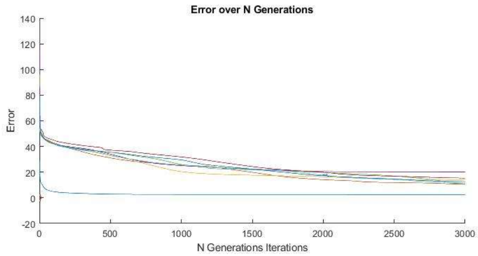

Figure 29 Error per iterations for N generations.

6. Conclusion

The theoretical study was performed in parallelizing the deep learning neural networks and in using parallel Bat's algorithm. It was observed the computational convergence in general increases with the number of processors employed. The practical part consisted of implementing neural networks on several Arduino microcontrollers. The weights of the neural networks were computed via combination of Backpropagation method, and Bat's algorithms. An application to speech recognition was implemented on Arduino microcontrollers and the performance of the deep neural network was observed. Three Arduino microcontrollers each having a neural network algorithm with selected words were tested. The performance was measured by observing the accuracy to speech commands given to the microcontrollers.

It was observed that high level of classification accuracy by the neural network of spoken words were achieved and therefore making it attractive for voice-controlled applications in robotics. In speech recognition, it is desirable to have accuracy near 99% for voice-controlled applications. Increasing the number of words to be recognized dynamically is a subject for future research.

Acknowledgments

This research was supported by the Takamul research grant #25/40 from the Deanship of Scientific Research at Islamic University of Madinah. The authors are also grateful to the Faculty of Computer Science and Information Systems at Islamic University of Madinah for their support in this research.

References

[1] Martin T. Hagan and Howard B. Demuth, Neural Network Design, 2nd Edition, ebook.

[2] Bahram Jalali, Claire Lifan Chen, and Ata Mahjoubfar, Cancer Diagnostics with Deep Learning and Photonic Time Stretch University of California, Los Angeles

[3] R.P. Lippmann, Pattern Classification using neural networks, IEEE Communications Magazine, vol 27, issue 111, Nov pp. 47-50, 1989.

[4] D. Zipser, R. A Andersen, "A Back Propagation Programmed Network that Simulates Response Properties of a Subset of Posterior Parietal Neurons", *Nature*, vol. 331, pp. 679-684, 1988.
[5] L. W. Chan, F. Fallside, "An Adaptive Training Algorithm for Back-Propagation Networks", *Comp. Speech and Language*, vol. 2, pp. 205-218, 1987.
[6] G. Nagy, "Candide's Practical Principles of Experimental Pattern Recognition ", *IEEE Trans. on Pattern Anal. and Machine Intel.*, vol. PAMI-5, no. 2, pp. 199-200, 1983,
[7] M. F. Tenorio, W. T. Lee, "Self Organizing Neural Networks for the Identification Problem" in Advances in Neural Information Processing Systems, CA, San Mateo:Morgan Kauffman, vol. 1, pp. 57-64, 1989.
[8] Fazal Noor, Pattern recognition using Neural Networks, Project, Islamic University of Madinah, 2018.
[9] Xiaolei Ma, et.al, Parallel Architecture of Convolutional Bi-Directional LSTM NeuralNetworks for NetworkWide Metro Ridership Prediction, IEEE Transaction on Intelligent Transportation Systems, pp 1-11, 2018
[10] Fazal Noor, Abdulghani Ibrahim, Mohammed Mostafa, Performance of Parallel Distributed Bat Algorithm using MPI on PC Clusters, Annals of Emerging Technologies in Computing, Jan 1 2020.
[11] James Mora, Dec 19, 2017, https://www.electronicdesign.com/embedded-revolution/chirp-microsystems-sticks-sonar-chips-detect-world-around-them, Accessed Nov 2018.
[12] Thomas Kopinski, Fabian Sachara, Uwe Handmann, A Deep Learning Approach to Mid-air Gesture Interaction for Mobile Devices from Time of Flight Data, Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services, pp 1-9, Hiroshima, Japan, Nov 28 to Dec 1, 2016.
[13] Yu Sang, Laixi Shi, and Yimin Liu, Micro Hand Gesture Recognition System Using Ultrasonic Active Sensing, IEEE Access, Vol 6, pp. 49339 – 49347, 2018.
[14] Chris J. Wellekens, Introduction to Speech Recognition Using Neural Networks, ESANN' proceedings-European Symposium on Artificial Neural Networks, Belgium, 22-24, April 1988.
[15] X.-S. Yang, "A New Metaheuristic Bat-Inspired Algorithm," Studies in Computational Intelligence, Springer Berlin, pp. 65-74, 2010.
[16] Syed Misbahuddin and Fazal Noor, "Hands-on Workshop on Parallel Processing", University of Hail, Saudi Arabia,: Syed Misbahuddin, 2007.
[17] D.-W. Huang and J. Lin, "Scaling Populations of a Genetic Algorithm for Job Shop Scheduling Problems using MapReduce," presented at the 2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), Indianapolis, IN, 2010.
[18] Y. Xin-She "Bat algorithm for multi-objective optimization," Internal Journal of BioInspired Computation, vol. 3, pp. 267-274, 2011.
[19] F. Wang, P. L. H. Yu, and D. W. Cheung, "Combining Technical Trading Rules Using Parallel Particle Swarm Optimization based on Hadoop," presented at the International Joint Conference on Neural Networks (IJCNN), Beijing, China, 2014.
[20] W. Zhao, H. Ma, and Q. He, "Parallel k-means clustering based on Mapreduce," vol. 5931, pp. 674-679, 2009.
[21] Barry Wilkinson and Michael Allen, Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers, North Carolina at Charlotte: Prentice Hall, 2005.
[22] K. Khan, A. Nikov, and A. Sahai, "A Fuzzy Bat Clustering Method for Ergonomic Screening of Office Workplaces," in Third International Conference on Software, Services and Semantic Technologies S3T 2011. vol. 101, ed: Springer Berlin Heidelberg, pp. 59-66. 2011.
[23] Eduardo R. Hruschkaaand Nelson F.F. EbeckenbCOPPE, "researchgate," 27 May 2002. [Online]Available:https://www.researchgate.net/publication/220571471_A_ genetic_ algorithm _for _cluster_analysis. [Accessed 3 Mar 2019.
[24] Chandra R., L. Dagum D. Kohr. D. Maydan, J. McDonald and R. Menon, Parallel Programming in OpenMP, San Francisco, CA: Margon Kaufmann Publishers, 2001.

Fazal Noor received the B. Eng. and M. Eng. degrees in Electrical and Computer Engineering from Concordia University, Montreal, Canada in 1984 and 1986, respectively. He received his Ph.D. from McGill University, Montreal, Canada in 1993. Currently, he is with the Faculty of Computer and Information Systems at Islamic University of Madinah, Saudi Arabia. He has published numerous papers in IEEE, ACM, Springer and various international journals and conferences. He has been a reviewer for IEEE, Elsevier, Springer, and various other journals. His research interests are in AI, FANETS, Neural Networks, Embedded Systems, Signal Processing, Network Security, Optimization Algorithms, and Parallel and Distributed computing.

Hatem M. El-Boghdadi is a professor of Computer Engineering at Cairo University, Giza, Egypt since 2015. He received his M.Sc. and Ph.D. in Computer Engineering form Assiut University and Louisiana State University, in 1994 and 2003 respectively. Currently, he is on leave at the Islamic University of Madinah, Saudi Arabia. His research interests include Parallel Architectures and Algorithms, Network-on-Chips, Reconfigurable Computing.