Optimal Convolutional Neural Network Architecture Design Using Clonal Selection Algorithm

Ali Al Bataineh and Devinder Kaur

***Abstract*—In this paper, the Clonal Selection Algorithm (CSA) is implemented to determine the optimal architecture of a convolutional neural network (CNN) for the classification of handwritten character digits. The efficacy of CNN in image recognition is one of the central motives why the world has woken up to the effectiveness of deep learning. During training, an optimal CNN architecture can extract complex features from the data that is being trained; however, the ideal architecture of a CNN for a specific problem cannot be determined by some standard procedure. In practice, CNN architectures are generally designed using human expertise and domain knowledge. By using CSA, optimal architecture of CNN can be determined autonomously through evolution of hyperparameters of the architecture for a given dataset. In this work, proposed methodology is tested on EMNIST dataset which is an enhanced version of MNIST dataset. The results have proven that the CSA based tuning is capable of generating optimal CNN architectures. Through this proposed technique, the best architecture of CNN for a given problem can be self-determined without any human intervention.**

***Index Terms*—Convolutional neural network, clonal selection algorithm, EMNIST dataset.**

I. Introduction

Machine learning is an application of artificial intelligence that deals with the study of algorithms that can learn from data. It enables systems to learn from historical data without being explicitly programmed. It is an exciting area of research that can drive the future of technology [1]. Machine learning has become an active area of research for the last two decades. Famous applications of machine learning include speech recognition, image recognition, self-driving cars and medical diagnosis, etc. These different applications of machine learning are impacting human lives in a positive manner.

Machine learning algorithms generally consist of two phases: training and prediction. In training phase, goal of any machine learning model is to learn those parameters of the model which best describes the overall training dataset. In prediction phase, learned model is used to make predictions. These predictions can then be used for many different real-world applications [2]. Machine learning algorithms can be categorized into two broad categories depending upon the nature of available dataset: supervised and unsupervised. Supervised algorithms are those, which require corresponding output labels in addition to input data for training, whereas unsupervised algorithms only require input data for training.

Neural networks are a set of Machine learning algorithms, which model complex relationships in data using a multilayer representation. A neural network is made up of number of individual units which are called neurons. These neurons are arranged in multiple layers which are connected to each other through these neurons. Neurons of one layer are connected to neurons of another layer through weighted edges. Weights of these edges are learned during training phase using training dataset. Data is passed from input layer to output layer of the network through these neurons. Neurons in each layer perform simple mathematical operations using learned weights and transmits the information to all the other neurons connected to it [3], [4].

Increase in computational power has increased the interest of researchers in deep neural networks, where the structure of network become more complex by adding more layers into the network. These deep neural networks are capable of solving wide range of problems adequately that could not be solved before such as Image classification [5]-[7].

A special class of deep neural networks is the convolutional neural networks, which are usually referred to as CNNs or ConvNets. CNNs are a special kind of artificial neural networks, which have a grid-like architecture and proposed in a paper by Yann LeCun in 1998 [8]. He used CNNs to classify handwritten digits and was able to achieve it through his first CNN which was called LeNet-5. For image classification, CNN is the state-of-art technique due to its grid-like architecture and used by several big players like Facebook, Google and Amazon etc. for various applications related to image classification. CNNs use a mathematical process called convolution, which is a special type of linear transformation. Unlike conventional neural networks, CNNs use convolution operation to obtain an intermediate output (feature). This intermediate output is then provided as input to the next layer. In a CNN architecture, this is done in at least one of its layers. A typical CNN architecture comprises of layers of different types, such as convolution, pooling and fully connected. In order to design such CNN architecture, users have to make various design decisions. The architecture of CNN needs to determine the types and number of layers, ordering of these layers, and hyperparameters of each layer. Hyperparameters of each layer include filter size for the convolution layers, activation type for each layer, pool size, and filter size for pooling layers. The vast number of architectures that can be generated based on these selections makes it difficult for an exhaustive manual search. While there has been some work going on automated discovery of

Manuscript received May 27, 2019; revised October 11, 2019.

The authors are with the EECS Department, the University of Toledo, Toledo, OH 43606 USA (e-mail: ali.albataineh@utoledo.edu, devinder.kaur@utoledo.edu).

doi: 10.18178/ijmlc.2019.9.6.874

best neural network architectures, new CNN architectures are still primarily proposed by researchers based on their problem's domain knowledge and intuition obtained from experimentation [9].

Lately, there have been notable research attempts to implement bio-inspired computing algorithms to evolve one or more aspects of convolutional neural networks (CNNs). The majority of the trial involving the evolution of CNN has centered on the network weights and network architecture. Clonal selection algorithm (CSA), which takes inspiration from human immune system, is a bio-inspired optimization technique, which has been recognized in machine learning to tackle the optimal solutions for several complex optimization problems in various areas.

In this paper, the Clonal Selection Algorithm (CSA) is implemented to automate the process of CNN's architecture selection. CSA is inspired by natural biological systems, and makes use of genetic operators such as selection, cloning, and mutation to find globally optimal solutions. The goal of the CSA algorithm is to discover the CNN architecture that best describes a given dataset without human intervention.

In the proposed CSA algorithm, the initial set of hyperparameters chosen for a CNN architecture is referred to as a foreign body known as an antigen. In order to fight the antigen, human immune system generates several antibodies.

Similarly, several antibodies are generated to fight the antigen in CSA algorithm. The affinity score of each antibody is computed, and clones are generated for each antibody. The antibody with the best affinity serves as the antigen for the second generation and the process is repeated until the best antibody is found. The best antibody depends upon the problem being solved. For a classification problem like handwritten character digits classification, the best antibody represents the set of hyperparameters that gives the maximum classification accuracy.

The paper is organized as follows. In Section II, the EMNIST dataset, which is used to evaluate the CNN architecture is introduced. In Section III, the architecture and the operations of a CNN are discussed. CSA is explained in section IV. Section V explains the implementation of proposed method to generate CNN architectures using CSA. Section VI discusses the results obtained. Section VII presents conclusion of the work.

II. EMNIST Dataset

The EMNIST is enhanced MNIST dataset that is the "Hello World" of pattern recognition in machine learning and deep learning. The "NIST" stands for National Institute of Standards and Technology, the institute that originally gathered this dataset and the "M" stands for "modified" [10]. Ease-of-use, accessibility and moderate size of MNIST dataset contributed to its widespread adoption.

The EMNIST dataset comprises of set of handwritten digits, lowercase and uppercase letters derived from NIST Special Database 19 [11]. It has a set of handwritten digits (0–9), (a-z) and (A-Z) being converted from 128 × 128 into 28 × 28 pixel pictures that directly matches the MNIST dataset as shown below in Fig. 1.

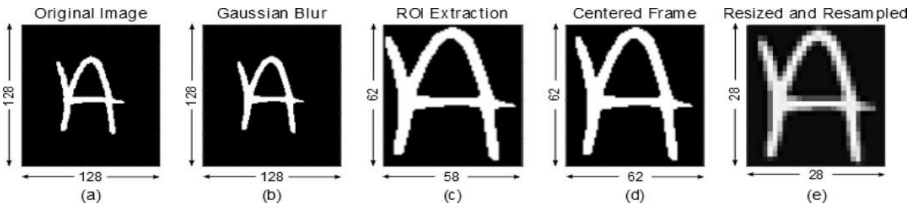

Fig. 1. Diagram of the conversion process used to convert the NIST dataset [11].

Fig. 1(a) shows the input image stored in a 128 × 128 pixels binary image. Fig. 1(b) represents Gaussian filter applied to the image to smoothen the edges. Fig. 1(c) the region around the actual digits extracted. In Fig. 1(d) digits placed in the center of a square image. Finally, the image is reshaped into 28 × 28 pixels using bi-cubic interpolation.

The EMNIST dataset is provided in six different splits. A brief outline of the dataset is given below:

- EMNIST ByClass: 62 unbalanced classes with a total of 814,255 characters.
- EMNIST ByMerge: 47 unbalanced classes with a total of 814,255 characters.
- EMNIST Balanced: 47 balanced classes with a total of 131,600 characters.
- EMNIST Letters: 26 balanced classes with a total of 145,600 characters.
- EMNIST Digits: 10 balanced classes with a total of 280,000 characters.
- EMNIST MNIST: 10 balanced classes with a total of 70,000 characters.

In this research, we used the EMNIST Digits as our data.

A. EMNIST Digits Dataset

The EMNIST Digits dataset provides balanced handwritten digit datasets directly compatible with the original MNIST dataset.

A short summary of the dataset is provided below:

- Train data: 240,000 characters.
- Test data: 40,000 characters.
- Total data: 280,000 characters.
- Classes: 10 balanced (0-9).

B. EMNIST Digits Dataset Preprocessing

The shape of the training dataset is (240k, 28, 28); 240k represents the number of images in the training dataset and (28, 28) represents the size of the image: 28 × 28 pixels. To be able to use the dataset in Keras API to work with the Convolution2D layers, we reshaped 3-dimensional arrays to 4-dimensional arrays (240k, 28, 28, 1), the last number is 1, which signifies that the images are greyscale. Same reshape process applies to the testing data as well. In addition, it is a good idea to normalize the input data to the range 0 and 1by dividing each pixel value to the maximum of 255. Finally, the output variable is an integer from 0 to 9. This is a multi-class

classification problem. As such, it is reliable practice to use the one-hot encoding of the class values, transforming the vector of class integers into a binary matrix.

III. Convolutional Neural Network

A CNN is a deep learning model that takes its inspiration from the human visual cortex model. CNNs are one of the most influential innovations in the computer vision field. They have outperformed conventional computer vision techniques and have delivered state-of-the-art results. Some real time applications of CNN are Image classification, face recognition, object detection, self-driving cars etc. [12], [13]. The founding father of Convolutional Neural Network is the well-known computer scientist working in Facebook Yann LeCun who was the first to use it to solve the handwritten digits problem using the MNIST Dataset [14]. CNNs are inspired by the biological visual cortex. The visual cortex contains small areas of cells that are sensitive to specific regions in the visual field. Hubel and Wiesel expanded this idea through successful experimentation in 1962 [15].

In this experiment, they pointed out that there are some individual neurons in the brain that fire only when they encounter a particular orientation like horizontal or vertical edges. In other words, some neurons are activated only when exposed to vertical edges and some when exposed to horizontal edges. Both Hubl and Wiesel discovered that all these neurons were arranged in a columnar fashion, due to which they were able to generate visual perception. The idea of specialized neurons within the virtual system having specific tasks is the inspiration behind CNNs.

A. CNN Architecture

For image classification, the main task is to take image as an input and infer an output class (a person, house, etc.) or a probability of certain class. Humans develop the recognition skill naturally and is one of the first learnt talent. However, these skills of pattern recognition and learning from experience are challenging for a computer. Computer sees an image as a matrix of pixel values. Based on the inputs, the computer should be capable of classifying the images provided to it and learn the unique features that make a person a person or that make a house a house [16].

Let us consider an image of size 32×32×3 (width, height, channels). In order to train an image classifier for images of this size, a fully connected neural network will require to learn 32×32×3 = 3072 weights for first hidden layer. For larger image size, the fully connected neural network will require more weights to learn. For example, for an image of size 340×340×3, number of weights for first hidden layer would be 346800, which will be computationally expensive. In addition, a fully connected neural network will have very large number of neurons for an image classification setup and hence these significant numbers of parameters can lead to overfitting.

CNN manages images differently, but still follows the general concept of a neural network. The basic building block of CNN is still a neuron with weights and biases and activation functions. There are several layers in CNN such as convolution, pooling and fully connected etc. Every neuron in a CNN receives the inputs, performs a simple mathematical operation and pass on the result to the next layer.

B. Layers in CNN

Each layer in a CNN architecture does a unique operation; however, there can be several layers of convolution and pooling layers.

1) Input layer

In CNN, input layer always consists of three-dimensional images (e.g. 300×300×3).

2) Convolutional layer

Convolution layer is designed to extract patterns (features) from the image. In the Convolution layer, element-wise multiplications between a region of the input image and a weight array called, a filter (kernel) is computed. The filter will slide across the whole image repeating the same dot product calculation. The output of convolution layer results in a feature map.

A filter is small spatially along the width and height but its depth is the same as that of the input image. For an image of depth 3, filter size of 3×3×3 is chosen to convolve with this image.

The process is repeated after sliding the filter across the image with a stride size and the output is a two- dimensional array, which is called a feature map. Three parameters namely number of filters; filter size, padding and stride control, determine the size of this feature map. These parameters need to be specified before performing the convolution.

Stride controls how the filter convolves across the image. When the stride is one, the filter moves across in single step, when stride is 2, filter moves across using two steps. The higher the stride, the less is the dimensions of the output feature map.

Zero padding means filling the image around the corners with zero when filter is around the edges of the image. Based on the above parameters, the dimensions of the output of the Convolution layer is given by equations (1) and (2):

$$\text{Output Feature Width} = \frac{2P + W - F}{S} + 1 \quad (1)$$

$$\text{Output Feature Height} = \frac{2P + H - F}{S} + 1 \quad (2)$$

where F represents the Filter size, S represents the stride W represents the width and H represents the height of the image and P represents the number of zero padding around an image. The number of channels of the feature map output is the same as the number of filters used through the convolution operation.

For example, for input image of size 9×9, W=9, H=9, Filter size F of 3×3, stride S equal to 1, and number of filters equal to 1 and padding P equal to zero, the output feature size is

Feature Map Size = 1 + (9 -3 +2×0) /1 =7.

Therefore, the output image will be 7×7.

3) Nonlinear layer (activation function)

The output of the feature Map is presented to nonlinear activation function ReLU. There are several types of nonlinear functions such as sigmoid, tanh, and ReLU. ReLU stands for Rectified Linear Unit is the choice for CNN because of its computational effectiveness and good

performance in terms of accuracy. ReLU's output is given in equation (3).

$$\text{ReLU} = \text{Max}\,(0, \text{input}) \tag{3}$$

ReLU activation function is applied per pixel and all negative values in feature map are replaced by zero.

4) Pooling layer

Pooling layer is also known as down sampling layer, which takes large images as its input and shrink them down while retaining most important information in those images. There are three types of pooling that are commonly used namely average, L2-norm and maxpool. Maxpool is the most widely used nonlinear function. Maxpool layer applies a filter of size $n \times n$ on image and takes the maximum value from the filter at each step. The stride with which filter steps across the input image is the same as that of filter size. Number of parameters (weights) can be reduced through the process of pooling, hence computation cost can be reduced, and overfitting can be avoided.

5) Fully connected layer

The output of the final pooling layer is provided to the fully connected network. A Fully Connected layer (FCL) is a regular artificial neural network layer, which is the last layer of CNN architecture. Activation function used by FCL in its output layer is softmax, which generates an output in range of 0 and 1. The generated output of softmax activation function gives the probability of any input belonging to a particular class i.e. categorical probability distribution. It calculates the probabilities of each target and normalize it with probabilities of all possible target classes. The softmax function is shown in Equation (4), where *z* represents the vector of the inputs presented to the output layer and j= {1, 2, 3,.. *K*} represents the output class.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{4}$$

6) CNN Training using backpropagation

As discussed earlier, in a CNN architecture, convolutional and pooling layers extract features like curves and edges from the input image, whereas FCL acts as a classifier at the end, which generates the probabilities for different classes. This is a forward propagation step. In training step, all parameters of a CNN are initialized randomly and it is trained using a backpropagation algorithm. Backpropagation algorithm keeps on adjusting the parameters (weights) of FCL in CNN until it converges and obtain a high classification accuracy. This classification accuracy can then be used as fitness value while tuning CNN architecture using CSA. The next section explains how CSA is applied to determine the architecture of a CNN for the EMNIST dataset. This section also describes the inner operations of the proposed algorithm.

IV. Clonal Selection Algorithm

The inspiration of natural biological systems is frequently used as a source to solve engineering problems in different domains [18]-[20]. CSA has enormous potential in various engineering applications. In recent past, researchers have proposed several different artificial immune models inspired by biological immune system to find global optima for various real-world applications.

CSA is inspired by biological immune system, one of the most complex biological systems. Basic building blocks of biological immune system are known as lymphoid organs [21]. These lymphoid organs are made up of lymphocytes which are a type of white blood cells having receptors to identify pathogens. There are two major types of lymphocytes, which are B-lymphocytes and T-lymphocytes. These B and T lymphocytes are also known as B and T cells respectively. Both these cells can identify certain molecular patterns found on pathogens and reproduce themselves through process of cloning to fight these organisms. Blood cells that generate and maturate in bone marrow and act as immune cells within bone marrow are called B-cells. Whereas, the cells which are produced in the bone marrow but migrated to thymus and maturate there are called T-cells. In thymus, T-cells become immune-capable and learn to differentiate among invasive cells and cells of organism [22].

The human body's immune system can identify foreign cells that invade the human body, which can be harmful and can cause infections or diseases. These foreign cells are known as Antigens. Immune system of human body learns how to cancel the effects of antigens by first understanding and then opposing the behavior pattern of antigen. Cells that are produced by human immune system to fight antigens are called antibodies. Affinity is a degree of recognition between antigen and antibody. The higher the affinity, the better the recognition, and vice-versa. CSA uses affinity to select the best antibodies. The best antibodies are cloned [23]. To introduce diversity mutation operation is applied on antibodies with an intent to improve their affinity so that they can destroy the antigen. The antibody with best affinity is picked and becomes the antigen for the next iteration. Antibodies are created in response to the new antigen, the best few are cloned and mutated, and again the best antibody is picked as antigen for the next iteration of the algorithm. The process is repeated for a fixed number of iterations or until the criterion for problem is achieved. The final antibody is the solution and provides the structure of CNN in terms of hyperparameters.

TABLE I: Hyperparameters Range of CNN

Hyperparameter	**Range**
Number of Epoch	0-127
Batch Size	0-256
Number of convolution layers	0-8
Number of filters at each convolution layer	0-64
Filter size of each convolution layer	0-8
Activation function used at each convolution layer	ReLU, Tanh, linear, Sigmoid
Maxpool layer after each convolution layer	True, False
Pool size of each maxpool layer	0-8
Number of feed-forward hidden layers	0-8
Number of feed-forward hidden neurons at each layer	0-64
Activation function used at each feed-forward layer	Tanh, ReLU, Sigmoid, Softmax
Optimizer	RMSprop, SGD, Adam, Adadelta

Before training a CNN, it is necessary to determine the optimal architecture of network for a given dataset. It is generally achieved through hyperparameter tuning. In this work, CSA is used to select best set of hyperparameters for CNN. Table I provides the ranges of hyperparameters associated with CNN architecture.

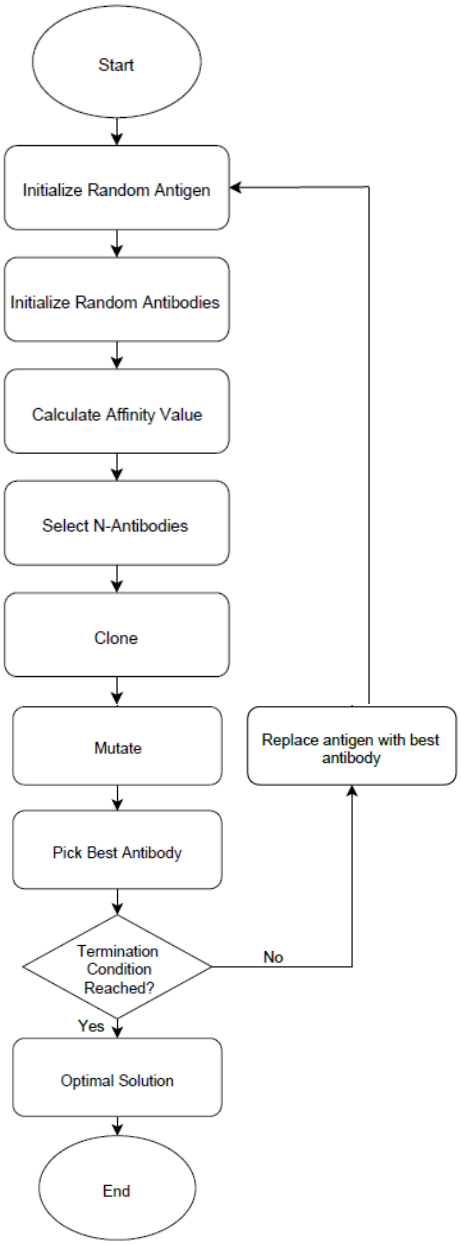

Fig. 2. Flowchart describing steps of the CSA.

TABLE II: MAPPING OF IMMUNE SYSTEM TERMINOLOGY TO CONVOLUTIONAL NEURAL NETWORK

Immune System	CNN Model
Antigen	Initial Solution (a set of hyperparameters)
Antibody	Candidate Solution
Gene	Hyperparameter
Affinity	Fitness value of each antibody to antigen
Cloning	Process of creating multiple copies of antibody
Mutation	Process of changing one or more Genes of Antibody
Population	Total number of Antibodies
Generation	Number of Iterations

In order to develop a computational model for the CNN based on CSA, the terminologies of the immune system need to be mapped to the architecture of the CNN. Table II shows the mapping of CSA terminology to the CNN. Fig. 2 shows the workflow diagram of the CNN hyperparameters tuning for the EMNIST data classification using CSA. It consists of the following steps [24].

A. Antigen Initialization

Antigen is s a toxin or foreign body that attacks the human body in the form of virus, bacteria, or fungus [25]. In the context of a convolutional neural network, an antigen is a potential solution that is represented as vector of all the hyperparameters that represent the architecture of CNN. The format of antibodies is the same as that of antigen. In each generation, antibodies that have higher fitness are selected. Here the fitness score refers to the image classification accuracy represented by the architecture of the CNN based on the hyperparameters described in the antibody or antigen. Firstly, a random antigen is created which represents the values of the hyperparameters of CNN. There are 12 genes in the antigen as the number of hyperparameters values, which need to be tuned using CSA. Each gene contains a hyperparameter value. Fig. 3 represents a random CNN architecture obtained from an antigen. The antigen is the ensemble of the value of hyperparameters, whose values lie in between the range as in Table I.

B. Antibody Initialization

In response to the initial antigen, several antibodies with the same format are created to fight the antigen. In this work, 10 antibodies were generated in response to one antigen.

C. Affinity Calculation

The affinity of the generated antibodies is measured according to the objective function of the problem that is being optimized. The objective function used in this work for the convolutional neural network is the classification accuracy and that is represented as ratio of correct predictions made to the total number of samples as described in equation 5. In equation 5, n represents the number of examples correctly classfied and N represents total number of examples in training set. The higher the classification accuracy, higher the fitness of the antibody. In other words, the CSA attempts to find a set of hyperparameters that maximizes the classificcation accuracy.

$$\text{Classification Accuracy} = \frac{n}{N} \tag{5}$$

D. Selection of Antibodies

Four antibodies that have the highest fitness among the 10 antibodies are selected.

E. Clone the Antibodies

In the cloning process, multiple copies of the antibodies are created. the number of clones is fixed to 10 and is the same for all the selected antibodies. Hence, for every iteration 10 clones are generated for each of the 4 selected antibodies.

F. Mutation

In the mutation process, antibodies are randomly altered to introduce diversity in the population. Mutation operation enables to achieve global optimization and hence helps to escape from local optimization. In immune system based

algorithms, mutation is performed on antibodies with an intent to enhance their ability to attack the foreign antigen better. There are two different types of mutation methods which are comonly used. First method is bit mutation, which replaces genes in antibody through fliping random bits in a binay array. Whereas second method is more complex, as it replaces genes in antibody with random values drawn from a Gaussian or Uniform distribution. In this work, the latter type is implemented.

G. Stopping Criteria

The algorithm stops running when the termination condition defined by the user is reached. In this work, termination condition is based on the classification accuracy of the algorithm. The algorithm is terminated if the desired classification accuracy is achieved. The hyperparameters evolved by the CSA algorithm determine the optimal architecture of the CNN.

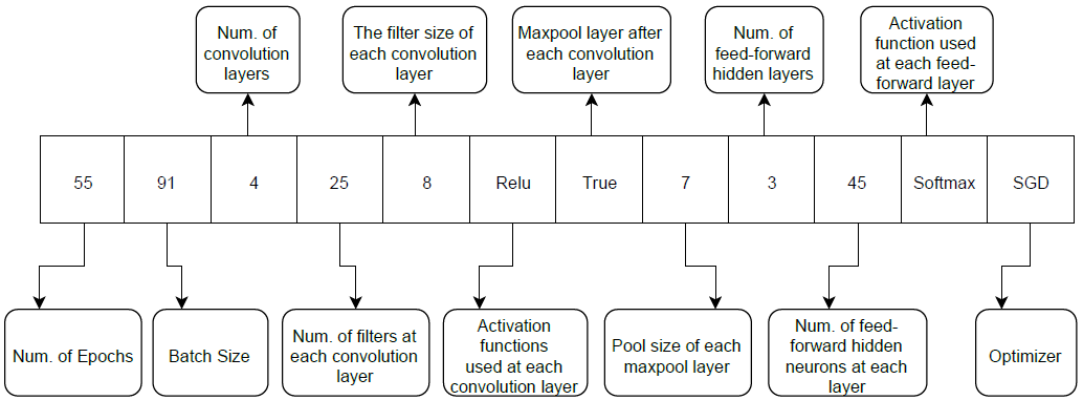

Fig. 3: A sample antigen of CNN.

V. Experimental Setup and Results

The CSA based CNN architecture tuner was implemented in Python 3.7 using a high-level neural networks API called

Keras running on top of TensorFlow. The proposed method was tested on the EMNIST Digits dataset with 240k images as its training set and 40k images as its testing set.

The experiment was carried out in a Dell desktop workstation powered by a processor-Intel(R) Core(TM) i7-7800X CPU @ 3.50GHz and 32 GB DDR4 RAM with Nvidia Quadro P4000 GPU with 8 GB of GDDR5 memory. The approximate processing time for CSA to generate a successful CNN architecture was 3.63 hours.

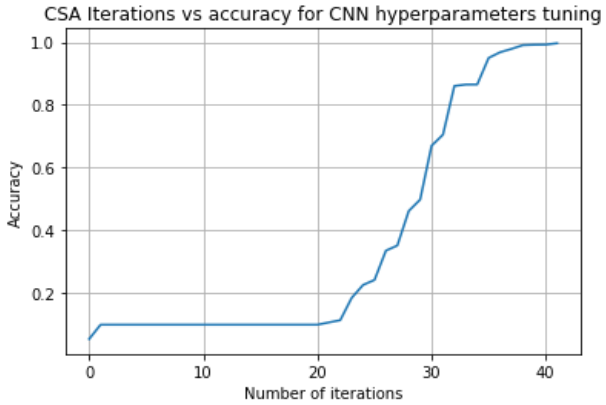

Fig. 4. Convergence curve for the hyperparameters tuning of CNN using clonal search algorithm.

The task of CSA tuner is to generate a CNN architecture, which gives the best classification accuracy for a given dataset. Fig. 4 shows the convergence curve of CSA optimization. It can be observed that accuracy is increasing with every iteration. The CNN architecture optimized through CSA gives a classification accuracy of 99.49%. Table III provides the final tuned hyperparameters attained using CSA.

TABLE III: Final Hyperparameters of the CNN Architecture Selected by CSA

Hyperparameter	Range
Number of Epoch	26
Batch Size	161
Number of convolution layers	6
Number of filters at each convolution layer	32
The filter size of each convolution layer	7×7
Activation functions used at each convolution layer	Relu
Maxpool layer after each convolution layer	True
Pool size of each maxpool layer	8×8
Number of feed-forward hidden layers	2
Number of feed-forward hidden neurons at each layer	6
Activation function used at each feed-forward layer	Softmax
Optimizer	Adam

VI. Conclusion

This paper developed a Clonal Selection Algorithm for evolving the optimal set of hyperparameters for CNN architecture that will best classify the images from EMNIST-Digits dataset. The accuracy achieved by the proposed method is 99.49%. Our proposed method is also adaptable to every uniform dataset. It provides an alternative approach to manually determining the best architecture of CNN by trial and error and that can be very time consuming and may not be optimal.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Author Devinder Kaur laid the framework of the problem and defined the process of mapping the CSA algorithm to tune the hyperparameters to develop optimal architecture of CNN for accurate classification of EMINST dataset. Author

Ali implemented the code of the algorithm in python. Both authors contributed to writing of the paper.

REFERENCES

[1] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine Learning*, vol. 3, no. 2, pp. 95–99, 1988.

[2] A. A. Bataineh, "A comparative analysis of nonlinear machine learning algorithms for breast cancer detection," *International Journal of Machine Learning and Computing* vol. 9, no. 3, pp. 248-254, 2019.

[3] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[4] J. J. Hopfield, "Artificial neural networks," *IEEE Circuits and Devices Magazine*, vol. 4, no. 5, pp. 3–10, 1988.

[5] B. M. Keneni, D. Kaur, A. Al Bataineh, V. K. Devabhaktuni, A. Y. Javaid, J. D. Zaientz, and R. P. Marinier, "Evolving rule-based explainable artificial intelligence for unmanned aerial vehicles," *IEEE Access*, vol. 7, pp. 17 001–17 016, 2019.

[6] A. Al Bataineh and D. Kaur, "A comparative study of different curve fitting algorithms in artificial neural network using housing dataset," in *Proc. NAECON 2018-IEEE National Aerospace and Electronics Conference*, 2018, pp. 174–178.

[7] A. Al Bataineh, D. Kaur, and A. Jarrah, "Enhancing the parallelization of backpropagation neural network algorithm for implementation on fpga platform," in *Proc. NAECON 2018-IEEE National Aerospace and Electronics Conference*, 2018, pp. 192–196.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," *arXiv preprint arXiv: 1611.02167*, 2016.

[10] P. J. Grother. (2016). NIST Special Database 19. NIST Handprinted Forms and Characters Database. [Online]. Available: https://www.nist.gov/publications/nist-special-database-19-nist-handprinted-forms-and-characters-database

[11] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: An extension of MNIST to handwritten letters," *arXiv preprint arXiv: 1702.05373*, 2017.

[12] MathWorks, Introducing Deep Learning with MATLAB. [Online]. Available: https://www.mathworks.com/content/dam/mathworks/tag-team/Objects/d/80879v00_Deep_Learning_ebook.pdf

[13] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of physiology*, vol. 160, no. 1, pp. 106–154, 1962.

[14] A. Karpathy, "Cs231n convolutional neural networks for visual recognition," *Neural Networks*, vol. 1, 2016.

[15] K. Ujjwal, "An intuitive explanation of convolutional neural networks," *The Data Science Blog*, 2016.

[16] Y. L. Cun, L. D. Jackel, B. Boser, J. S. Denker, H. P. Graf, I. Guyon, D. Henderson, R. E. Howard, and W. Hubbard, "Handwritten digit recognition: Applications of neural network chips and automatic learning," *IEEE Communications Magazine*, vol. 27, no. 11, pp. 41–46, 1989.

[17] A. Bhandare and D. Kaur, "Designing convolutional neural network architecture using genetic algorithms," in *Proc. on the International Conference on Artificial Intelligence*, 2018, pp.150–156.

[18] Y. Atay and H. Kodaz, "Optimization of job shop scheduling problems using modified clonal selection algorithm," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 22, no. 6, pp. 1528–1539, 2014.

[19] A. A. Bataineh, A. Jarrah, and D. Kaur, "High-speed FPGA-based of the particle swarm optimization using HLS tool," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 10, no. 5, 2019.

[20] A. Mairaj, A. A. Bataineh, D. Kaur, and A. Y. Javaid, "*Identifying the Optimal Solutions of Bohachevsky Test Function Using Swarming Algorithms*," in *Proc. 21st International Conference on Artificial Intelligence*, Las Vegas, USA, July 29-August 1, 2019.

[21] L. N. de Castro and F. J. V. Zuben, "Immune and neural network models: Theoretical and empirical comparisons,"*International Journal of Computational Intelligence and Applications*, vol. 1, no. 3, pp. 239–257, 2001.

[22] I. Muthreja and D. Kaur, "A comparative analysis of immune system inspired algorithms for traveling salesman problem," in *Proc. on the International Conference on Artificial Intelligence*, 2018, pp. 164–170.

[23] J. Brownlee, *Clever Algorithms: Nature-Inspired Programming Recipes*, Jason Brownlee, 2011.

[24] A. A. Bataineh and D. Kaur, "Immuno-computing-based neural learning for data classification," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 6, 2019.

[25] L. N. de Castro and F. J. Von Zuben, "Learning and optimization using the clonal selection principle," *IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems*, vol. 5, pp 291-317, 2002.

Ali Al Bataineh received a B.S. degree in computer engineering from Yarmouk University, Jordan, in 2010, and the M.S. degree in computer engineering from the University of Bridgeport, Bridgeport, CT, USA, in 2016. He is currently pursuing a Ph.D. degree with the Department of Electrical Engineering and Computer Science, The University of Toledo, Toledo, OH, USA. His current research interests include the areas of artificial intelligence, machine learning, computer vision, metaheuristic optimization, fuzzy logic, and FPGAs.

Devinder Kaur received the B.Sc. and M.Sc. degrees (Hons.) in physics, majoring in electronics, from Panjab University, in 1969 and 1970, respectively, the M.Sc. degree in medical physics from the University of Aberdeen, U.K., in 1976, under the Commonwealth Scholarship Award, and the M.Sc. and Ph.D. degrees in computer engineering from Wayne State University, USA, in 1985 and 1989, respectively. She was a scientist with the Central Scientific Instruments Organization, a national laboratory, under the Ministry of Science and Technology, Chandigarh, India, from 1971 to 1981. In 1989, she joined the University of Toledo, as a faculty member, where she is currently a full professor with the Department of EECS. She visited Nippon Institute of Technology, Japan, as a fulbright senior specialist in 2003. She has published over 100 papers in refereed journals and proceedings of the international conferences. She has worked on projects funded by NSF, AFRL, Daimler Chrysler, and ROMAN Engineering. Her research interests include developing intelligent applications based on hybrid computational models using biologically inspired computing and fuzzy systems. She received the University Medal for obtaining the first rank. She was a recipient of the IIT Delhi Fellowship from 1970 to 1971 and the Fulbright Senior Specialist Award, in 2004.