Static Hand Gesture Recognition using Convolutional Neural Network with Data Augmentation

Md. Zahirul Islam
Department of Computer Science and Engineering
University of Chittagong
Chittagong, Bangladesh
zahirulislam261@gmail.com

Mohammad Shahadat Hossain
Department of Computer Science and Engineering
University of Chittagong
Chittagong, Bangladesh
hossain_ms@cu.ac.bd

Raihan Ul Islam
Department of Computer Science, Electrical and Space Engineering
Lule University of Technology
Skellefte, Sweden
raihan.ul.islam@ltu.se

Karl Andersson
Department of Computer Science, Electrical and Space Engineering
Lule University of Technology
Skellefte, Sweden
karl.andersson@ltu.se

***Abstract*—Computer is a part and parcel in our day to day life and used in various fields. The interaction of human and computer is accomplished by traditional input devices like mouse, keyboard etc. Hand gestures can be a useful medium of human-computer interaction and can make the interaction easier. Gestures vary in orientation and shape from person to person. So, non-linearity exists in this problem. Recent research has proved the supremacy of Convolutional Neural Network (CNN) for image representation and classification. Since, CNN can learn complex and non-linear relationships among images, in this paper, a static hand gesture recognition method using CNN was proposed. Data augmentation like re-scaling, zooming, shearing, rotation, width and height shifting was applied to the dataset. The model was trained on 8000 images and tested on 1600 images which were divided into 10 classes. The model with augmented data achieved accuracy 97.12% which is nearly 4% higher than the model without augmentation (92.87%).**

***Index Terms*—Convolutional Neural Network, Static hand gestures recognition, Data augmentation.**

I. Introduction

Bobick and Wilson have given a definition of gesture. According to them, the motion of the body that is intended to communicate with other agents can be defined as gesture [1]. The sender and receiver must have same sort of information for a successful gesture. Gestures may be classified as dynamic or static. A dynamic gesture intends to vary over a period of time and static gesture tends to remain almost unchanged over time. This project emphasizes to recognize static gestures. The automated recognition of hand gestures may have applications in various area such as in design, robotics, virtual reality and most importantly in sign language.

The key problem is how to make a computer able to understand the hand gestures. Hand gestures vary in orientation of fingers and shape of hands. So, non-linearity is one of the characteristics of hand gestures that has to be dealt. It can be done using the metadata and content information carried by the images. The meta information of hand gesture images can be used to recognize the gestures. The process is the integration of two tasks: feature extraction and classification. Prior to the recognition of any gesture, the features of an image must be extracted. After extracting those features, any classification method should be applied. So, the main problem is how to extract those features and imply those features for classification.

Huge features are mandatory for classification and recognition. Conventional models for pattern recognition cannot process natural data in raw form [2]. Therefore, huge efforts are required to extract features from raw data and they are not automated. CNN, a class of deep learning neural network, can extract features on the fly [3] and fully connected layers can be used for classification. CNN combines these two steps to reduce the memory requirements and computational complexity and gives a better performance. It can also understand the complex and non-linear relationships among the images. Therefore, CNN based approach will be used to solve the problem.

Therefore, this paper emphasizes on recognition of static hand gestures by building a model using CNN that can analyze large amount of image data and recognize static hand gestures. The other objectives are to investigate the existing methods of gesture recognition and analyzing the effect of data augmentation in deep learning.

II. Related Works

Various works have been done on hand gesture recognition and some notable research in this topic are mentioned.

A hand gesture recognition system using an artificial neural network (ANN) based on shape fitting technique has been developed [4]. In this system, a color segmentation technique on YbrCr color space was used after filtering to detect hand. Then the shape of the hand was approached by the hand morphology. The shape of hands and finger orientation features were extracted and passed to an ANN. They achieved 94.05% accuracy by using this method.

Gesture detection by using an ANN has also been developed [5]. In this system, images were segmented based on skin colors. The selected features for ANN were changes of pixel through cross sections, boundary and the scalar description like aspect ratio and edge ratio. After establishing those feature vectors, they were fed to the ANN for training. The accuracy was around 98%.

A statistical method based on haar-like features to detect gestures was proposed [6]. In this system, AdaBoost algorithm was used to learn the model. The whole work was divided into two levels. In higher level, a stochastic context-free grammar was used to detect gestures. In lower level, postures were detected. A terminal string was generated for each input according to the grammar. The probability associated with each rule was calculated and the rule with highest probability for the given string was selected as the rule. The gesture associated with this rule was returned as the gesture of the input.

However, feature extraction by manual process has some drawbacks. The extraction process is tedious and every possible feature may not be extracted. Moreover, the extraction may become human-biased.

Then comes the automated feature engineering which is not tedious and not human biased. Also, almost all the features can be captured using automated feature engineering. Useful features from structured data can be extracted by CNN. So, a move to automated feature engineering was made and deep learning i.e. CNN started to emerge.

An algorithm to recognize hand gestures using a 3D CNN was proposed [7]. In this system, the basis of the recognition was challenging depth and intensity of the images. They also used data augmentation technique and achieved accuracy about 77.5% on VIVA challenge dataset [7].

Another system to detect gesture using CNN which is robust under five invariants scale, rotation, translation, illumination, noise and background was proposed [8]. Sign Language of Peru (LSP) were used as dataset. They achieved 96.20% accuracy on the LSP Dataset.

III. Experimental Methodology

This section provides the description of the dataset and CNN configuration that were used. The flowchart of methodology is shown on Figure 1. The approach is the combination of data collection, pre-processing, configuring the CNN and building the model.

A. Input Data and Training Data

Images needed to train and validate the model were collected using a webcam. The gestures were performed by 10 persons in front of the webcam. It is assumed that the input images exactly include one hand, gestures were made with right hand, the palm facing the camera and the hand were roughly vertical. The recognition process will be less complex and more efficient if the background is less complex and the contrast is high on the hand. So, it is assumed that the background of the images was less complex and uniform.

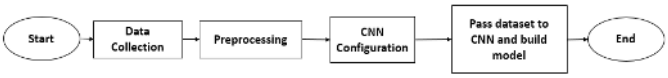


Fig. 1. System Framework

B. Pre-Processing

A minimal pre-processing was applied over the dataset to reduce the computational complication and achieve better efficiency. Firstly, the background of the images was removed using the method of background subtraction proposed by Z. ZivKovic [9] [10]. The background subtraction is mainly based on K-gaussian distribution which selects appropriate gaussian distribution for each pixel and provides a better adaptability on varying scenes due to illumination changes. After subtracting background, only the image of hand remains.

Then the images were converted to grayscale image. Since grayscale images contain only one color channel it will be easier for CNN to learn [11]. Then a morphological erosion was applied [12]. After that, median filter were applied to reduce the noise. In signal processing, it is often desirable to reduce noises [13]. Figure 2 visualizes the pre-processing steps. The images were then resized to size 50x50 for feeding to CNN.

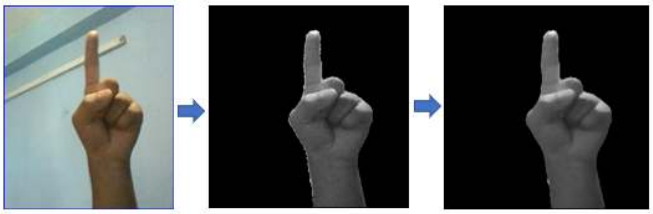


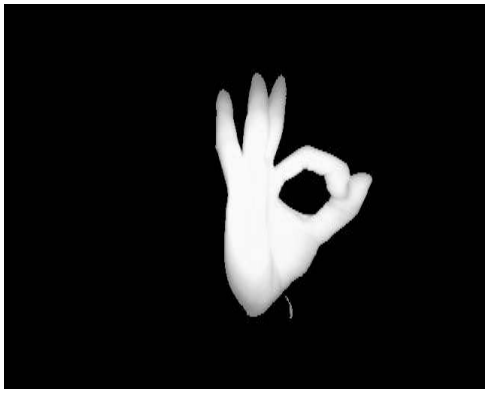
Fig. 2. Steps of Preprocessing

In addition to our self-developed dataset, another dataset named "Hand Gesture Recognition Database" [14] was also used in this experiment. By selecting largest object, hand in this case, other objects from these images were removed. Figure 3 shows the effect of selecting largest object.

C. Dataset

We selected 10 static gestures (Index, Peace, Three, Palm Opened, Palm Closed, OK, Thumbs, Fist, Swing, Smile) to recognize. Each class has 800 images for training and 160 images for testing purpose. So total number of images is 8000

(a)



(b)

Fig. 3. (a) Original Image and (b) Image after Selecting Largest Object

for training and 1600 for testing. Sample of finalized dataset is provided on Figure 4.

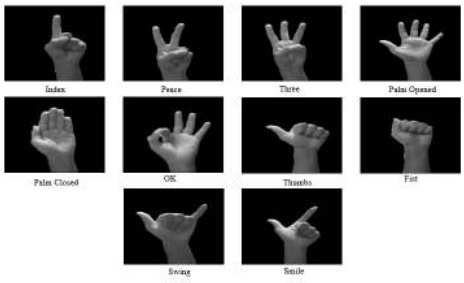


Fig. 4. Sample Images from self-developed Dataset

"Hand Gesture Recognition Database" [14] also contains 10 classes (Palm, I, Fist, Fist_Moved, Thumb, Index, OK, Palm_Moved, C, Down), each class having 2000 images. A snapshot from the database is provided in Figure 5.

D. CNN Configuration

The CNN that that has been considered in this research to recognize hand gesture is composed of two convolution layers, two max pooling layers, two fully connected layers and output

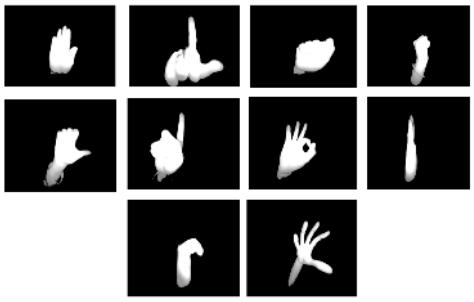


Fig. 5. Sample Images from Hand Gesture Recognition Database

layer. There are three dropout performance in the network to prevent over-fitting [15].

The first convolution layer has 64 different filters with the kernel size 3x3. The activation function used in this layer is Rectified Linear Unit (ReLU). ReLU was applied to introduce non-linearity [16] and it has been proved that ReLU performs better than other activation functions such as tanh or sigmoid. As it is input layer, we have to specify the input size. The stride is set to default. The input shape is 50x50x1 which means that gray-scale image of size 50x50 should be provided to this network. This layer produces the feature maps and passes them to the next layer.

Then the CNN has a max pooling layer with pool size 2x2 which takes the maximum value from a window of size 2x2. The spatial size of the representation is reduced progressively as the pooling layer takes only the maximum value and discards the rest. This layer helps the network to understand the images better because it only selects more important features.

The next layer is another convolution layer and it has 64 different filters with the kernel size 3x3 and default stride. Again, ReLU was used as the activation function in this layer. This layer is followed by another max pooling layer which has a pooling size 2x2. In this layer, first dropout was added which randomly discards 25% of the total neurons to prevent the model from over-fitting. Output from this layer is passed to the flatten layer.

Output from the previous layers are received by the flattening layer and they are flattened to a vector from two-dimensional matrix. This layer allows the fully connected layers to process the data achieved till now.

The next layer is first fully connected layer which has 256 nodes and ReLU was used as the activation function. The layer is followed by a dropout layer which excludes 25% of the neurons to prevent overfitting.

The second fully connected layer again has 256 nodes to receive the vector produced by first fully connected layer and uses ReLU as activation layer. The layer is followed by a dropout layer to exclude 25% of the neurons to prevent over-fitting.

The output layer has 10 nodes corresponding to each classes of the hand gestures. This layer uses SoftMax function [17]

as activation function which outputs a probabilistic value for each of the classes.

The model is then compiled with Stochastic Gradient Descent (SGD) [18] function with a learning rate 0.001. To evaluate loss, categorical cross-entropy function [17] was used since the model is compiled for more than two classes. Finally, the metrics of loss and accuracy were specified to keep track on the evaluation process.

This configuration was chosen after trying various combination of nodes and layers.

Summary of the CNN configuration is provided on Table I.

TABLE I
CNN CONFIGURATION

Model Content	Details
First Convolution Layer	64 filters of size 3x3, ReLU, input size 50x50
First Max Pooling Layer	Pooling Size 2x2
Second Convolution Layer	64 filters of size 3x3, ReLU
Second Max Pooling layer	Pooling size 2x2
Dropout Layer	Excludes 25% neurons randomly
First Fully connected Layer	256 nodes, ReLU
Dropout Layer	Excludes 25% neurons randomly
Second Fully Connected Layer	256 nodes, ReLU
Dropout Layer	Excludes 25% neurons randomly
Output Layer	10 nodes for 10 classes, SoftMax
Optimization Function	Stochastic Gradient Descent (SGD)
Learning Rate	0.001
Metrics	Loss, Accuracy

E. System Implementation

To implement the system, python was used as the programming language and a python IDE Spyder was used to write and run code. The library Keras was used for building the CNN classifier. The library PIL was used for image preprocessing. Sklearn was used to calculate the confusion matrix. Matplotlib was used to visualize model accuracy and loss values and confusion matrix. NumPy was used for array operations.

The training process on dataset is composed of two phases.

1) Training with Base Dataset: In this phase, the model was trained using the base dataset achieved after pre-processing.

2) Training with Expanded Dataset: In this phase, the dataset was augmented. Data augmentation is a technique to increase the number of data by applying zoom, shear, rotation, flip etc [19]. This process not only increases the data but also brings variation in dataset which is essential for CNN to learn sophisticated differences of images. Figure 6 shows the effect of augmentation. A random image was selected to provide the demonstration.

IV. EXPERIMENTAL RESULT

This section describes the results obtained form the experiment using the CNN configuration according to Table I. The experimental result shows that the model which was augmented with temporary data achieved 97.12% accuracy which is about 4% higher than the model without any augmented data.

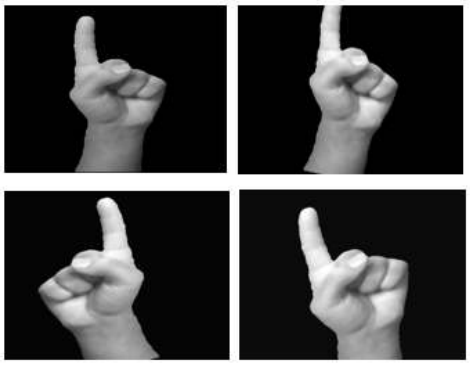
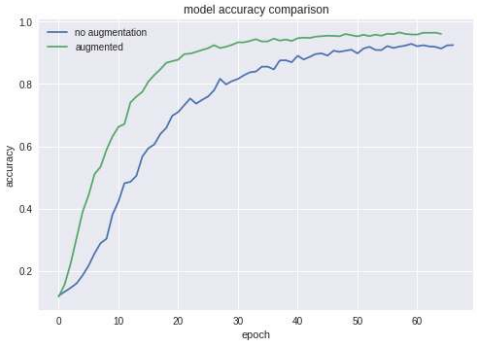
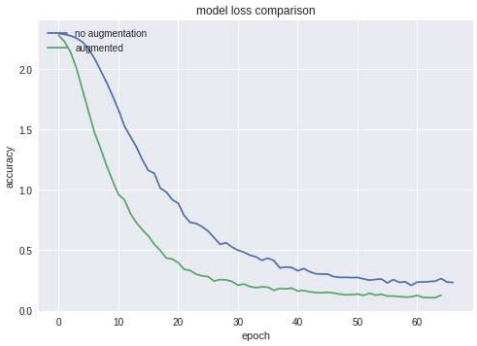


Fig. 6. Effects of Data Augmentation



(a)



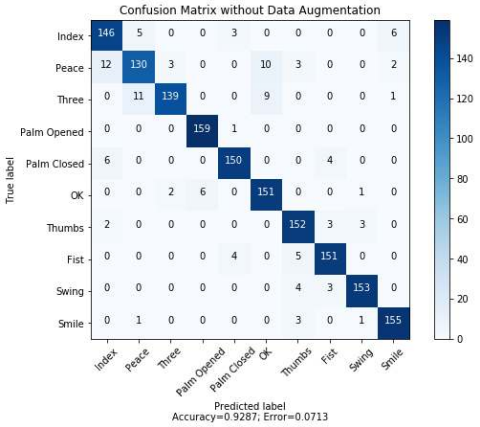
(b)

Fig. 7. Compared (a) Accuracy and (b) Loss of the Models

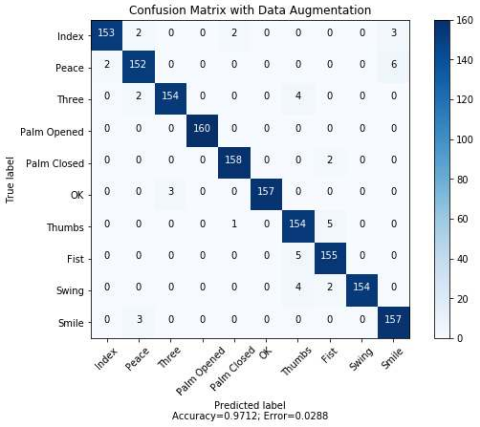
The accuracy and loss comparison graphs are shown on Figure 7. The graphs show that non-augmented model was early-stopped after 65 epochs and augmented model was early-stopped after 67 epochs. The accuracy of augmented model was higher than non-augmented model at each epoch. It also shows that the progress of accuracy on augmented model was faster than non-augmented model. The loss of augmented model was also less than non-augmented model.

The confusion matrices of both models are provided in

Figure 8. The diagonal values dictate the number of tuples that were correctly classified by the models and off-diagonal values represent the number of misclassified tuples. The higher the diagonal value, the better the performance. These matrices tell us that non-augmented model could not perform better on three classes (Index, Peace and Three) and the augmented model showed a better performance on these classes since the model was provided with increased amount of data. Thats why the performance was better in the case of augmented model.



(a)



(b)

Fig. 8. Confusion Matrix for (a) Non-augmented Model and (b) Augmented Models

Other performance measures provided on Table II prove this statement. According to Figure 9, the training and testing accuracy were close to each other in every epoch which indicates that the model was not over-fitted due to data augmentation.

The same experiment was performed again using train-test split 65-35. It was observed that accuracy for 65-35 split is 96.57%, which indicates the adaptability of the model for larger test set.

TABLE II
CLASSIFICATION RESULT

Performance Measure	CNN without augmentation	CNN with augmentation
Precision	0.9291	0.9718
Recall	0.9287	0.9713
F-Measure	0.9289	0.9715
Accuracy	92.87%	97.12%

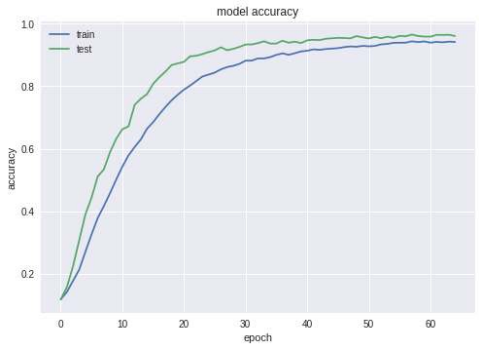


Fig. 9. Training vs Testing Accuracy of Augmented Model

The same dataset was passed as input to Support Vector Machine (SVM) and K Nearest Neighbors (KNN) models. These models achieved accuracy of 72% and 75% respectively. One of the reasons of poor performance showed by SVM and KNN is the adaptability issue with non-linear dataset. Since, the datset was provided in a raw format, their accuracy were lower than CNN.

By applying CNN to the "Hand Gesture Recognition Database" [14] with a splitting ratio 70:30, the achieved accuracy was 98.95% which proves the adaptability of the proposed method for other dataset. The confusion matrix achieved from the dataset [14] is shown on Figure 10.

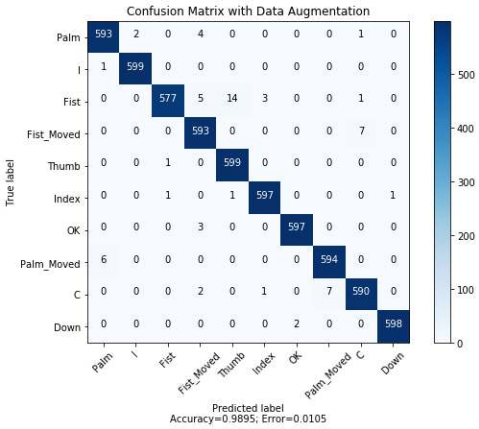


Fig. 10. Confusion Matrix of Hand Gesture Recognition Database

V. Conclusion and Future Works

This research explores the opportunity and challenges in recognition of hand gestures. It also analyzes the effect of data augmentation in deep learning. We can say after this research that CNN is a data driven methodology and data augmentation has a huge effect in deep learning. Although the system can recognize gestures successfully, some extension is still possible. For example, by applying knowledge driven methodology such as Belief Rule Base (BRB), which is widely used where uncertainty becomes an issue [20] [21] [22] [23] [24]. Hence, the recognition of gesture can be accomplished more accurately. More gestures can be added to list of recognition. It was assumed that the background should be less complex. Therefore, recognition of gestures in complex background can be another extension. Recognition of gestures made with both hands is not possible by this system. Therefore, another future work can be the recognition of gestures made with both hands.

Acknowledgment

This study was funded by the Swedish Research Council under grant 2014-4251.

References

[1] A. D. Wilson and A. F. Bobick, "Learning visual behavior for gesture analysis," in *Proceedings of International Symposium on Computer Vision-ISCV*. IEEE, 1995, pp. 229–234.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[3] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, contour and grouping in computer vision*. Springer, 1999, pp. 319–345.

[4] E. Stergiopoulou and N. Papamarkos, "Hand gesture recognition using a neural network shape fitting technique," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 8, pp. 1141–1158, 2009.

[5] T.-N. Nguyen, H.-H. Huynh, and J. Meunier, "Static hand gesture recognition using artificial neural network," *Journal of Image and Graphics*, vol. 1, no. 1, pp. 34–38, 2013.

[6] Q. Chen, N. D. Georganas, and E. M. Petriu, "Hand gesture recognition using haar-like features and a stochastic context-free grammar," *IEEE transactions on instrumentation and measurement*, vol. 57, no. 8, pp. 1562–1571, 2008.

[7] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Hand gesture recognition with 3d convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2015, pp. 1–7.

[8] C. J. L. Flores, A. G. Cutipa, and R. L. Enciso, "Application of convolutional neural networks for static hand gestures recognition under different invariant features," in *2017 IEEE XXIV International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*. IEEE, 2017, pp. 1–4.

[9] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *null*. IEEE, 2004, pp. 28–31.

[10] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern recognition letters*, vol. 27, no. 7, pp. 773–780, 2006.

[11] M. Grundland and N. A. Dodgson, "Decolorize: Fast, contrast enhancing, color to grayscale conversion," *Pattern Recognition*, vol. 40, no. 11, pp. 2891–2896, 2007.

[12] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology," *IEEE transactions on pattern analysis and machine intelligence*, no. 4, pp. 532–550, 1987.

[13] Y. Zhu and C. Huang, "An improved median filtering algorithm for image noise reduction," *Physics Procedia*, vol. 25, pp. 609–616, 2012.

[14] T. Mantecón, C. R. del Blanco, F. Jaureguizar, and N. García, "Hand gesture recognition using infrared imagery provided by leap motion controller," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2016, pp. 47–57.

[15] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[16] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[17] R. A. Dunne and N. A. Campbell, "On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function," in *Proc. 8th Aust. Conf. on the Neural Networks, Melbourne*, vol. 181. Citeseer, 1997, p. 185.

[18] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[19] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," *arXiv preprint arXiv:1712.04621*, 2017.

[20] R. U. Islam, M. S. Hossain, and K. Andersson, "A novel anomaly detection algorithm for sensor data under uncertainty," *Soft Computing*, vol. 22, no. 5, pp. 1623–1639, 2018.

[21] M. S. Hossain, S. Rahaman, A.-L. Kor, K. Andersson, and C. Pattinson, "A belief rule based expert system for datacenter pue prediction under uncertainty," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 2, pp. 140–153, 2017.

[22] M. S. Hossain, F. Ahmed, K. Andersson *et al.*, "A belief rule based expert system to assess tuberculosis under uncertainty," *Journal of medical systems*, vol. 41, no. 3, p. 43, 2017.

[23] M. S. Hossain, P.-O. Zander, M. S. Kamal, and L. Chowdhury, "Belief-rule-based expert systems for evaluation of e-government: a case study," *Expert Systems*, vol. 32, no. 5, pp. 563–577, 2015.

[24] R. Ul Islam, K. Andersson, and M. S. Hossain, "A web based belief rule based expert system to predict flood," in *Proceedings of the 17th International conference on information integration and web-based applications & services*. ACM, 2015, p. 3.