Auto-Encoder Variants for Solving Handwritten Digits Classification Problem

Muhammad Aamir¹, Nazri Mohd Nawi², Hairulnizam Bin Mahdin¹, Rashid Naseem³, and Muhammad Zulqarnain¹
¹Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Batu Pahat, Malaysia
²Soft Computing and Data Mining Center, Universiti Tun Hussein Onn Malaysia, Batu Pahat, Malaysia
³Department of IT and Computer Science, Pak-Austria Fachhochschule Institute of Applied Sciences and Technology, Haripur, Pakistan

Abstract

Auto-encoders (AEs) have been proposed for solving many problems in the domain of machine learning and deep learning since the last few decades. Due to their satisfactory performance, their multiple variations have also recently appeared. First, we introduce the conventional AE model and its different variant for learning abstract features from data by using a contrastive divergence algorithm. Second, we present the major differences among the following three popular AE variants: sparse AE (SAE), denoising AE (DAE), and contractive AE (CAE). Third, the main contribution of this study is performing the comparative study of the aforementioned three AE variants on the basis of their mathematical modeling and experiments. All the variants of the standard AE are evaluated on the basis of the MNIST benchmark handwritten digit dataset for classification problem. The observed output reveals the benefit of using the AE model and its variants. From the experiments, it is concluded that CAE achieved better classification accuracy than those of SAE and DAE.

Keywords: Sparse auto-encoder (SAE), Denoising auto-encoder (DAE), Contractive auto-encoder (CAE), MNIST, Classification

1. Introduction

Auto-encoders (AEs) are unsupervised neural networks that apply the back-propagation behavior by setting up the high-dimensional input feature set to a low-dimensional output feature set and then recovering the original feature set from the output. The reduction procedure of high-dimensional data to low-dimensional data is known as encoding, while the reconstruction of the original data from the low-dimensional data is called decoding. AE was proposed to improve the reconstruction reliability of low-dimensional feature sets. Notably, feature-set selection is the most important step in solving big-data problems and processing complex data with high-dimensional attributes [1]. In addition, feature-selection is a valuable method for performing classification and prediction [2] [3]. It is a satisfactory procedure to split the useful and effective features from ineffective and useless ones within the feature representation space. However, if irrelevant raw features are provided as input, feature selection might fail [4] [5]. Using AEs is one of the most efficient approaches for feature reduction, as it results in satisfactory classification. It traditionally works in two phases,
namely, encoding and decoding. The process of converting the input features to a new representation is called encoding, whereas that of converting this new representation back to the original inputs is called decoding. The primary aim of AEs is to identify more useful and informative features from a large amount of data without applying any feature-reduction technique [6][8].

To improve the efficiency of the standard AE, [9] proposed an AE-based learning approach with two-stage architecture. They stacked the AE by considering both intra- and inter-modal semantics and named it sparse AE (SAE). Similarly, [10] proposed a feature-learning approach by stacking the contractive AE (CAE) and naming it sCAE. Their research work focused on extracting the temporal feature difference of superpixel and noise suppression. To benefit from both the characteristics of AE and properties of conventional algorithms, [11][12] merged the CAE with the convolutional neural network to form a deep architecture named DSCNN, to solve the problem of both the presence of speckle noise and absence of effective features in single-polarized SAR (synthetic aperture radar) data.

Notable, some popular AE variants are SAEs [13], denoising AEs (DAEs) [14], and CAEs [15]. In this study, we first introduce AE and then present its three popular variants, namely, SAEs, DAEs, and CAEs. We conducted some experiments to empirically compare their performances and properties with one another. A comprehensive analysis based on the experimental results is also given. Conclusion is presented in the last section.

2. Auto-Encoder

AEs are also called auto-associators. In the domain of machine learning, AEs comprise a complicated neural network with three layers and have been investigated by many researchers since the 1990s [16]. The entire processing inside an AE proceeds in two phases, namely encoding and decoding, as depicted in Figure [1].

2.1 Encoding

The process of mapping the input feature set to transform it to give its intermediate representation to the hidden layer is called encoding, which is given as follows:

\[ y = f(x) = s_e(w_x + b_h), \]  

(1)

where \( f(x) \) denotes the outputs of the input layer that is given as inputs to the hidden layer, \( w \) the weights given to each input, and \( b \) the biasness value associated with the input feature set.

2.2 Decoding

The process of mapping the output of the hidden layer back into the input feature set is called decoding, which is given as follows:

\[ r = g(y) = s_d(w'_y + b'_r), \]  

(2)

where \( g(y) \) denotes the output of the hidden layer, \( w' \) the weights given to the inputs of the hidden layer, and \( b' \) the biasness of the inputs given to the hidden layer.

The terms \( s_e \) and \( s_d \) denote the encoding- and decoding-activation functions, respectively, and are given by Eq. (3) for nonlinear representation (sigmoid function) and by Eq. (4) for linear representation (hyperbolic tangent function). One has the following:

\[ \text{sigmoid}(x) = \frac{1}{(1 + e^{-x})}, \]  

(3)

\[ \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}. \]  

(4)

The primary aim of the reconstruction is to generate the outputs that are similar to the original inputs to the maximum possible extent, by reducing the reconstruction error. The reconstruction layer uses the following parameter set to reconstruct the original inputs:

\[ \theta = [W, h_h, b_r]. \]  

(5)

Let us assume we have the input feature set as \( D_i \), then the reconstruction error is minimized by minimizing the following cost function:

While \( D_i = [x_1, x_2, x_3, \ldots, x_n] \).

\[ J_{AE} (\theta) = \sum_{x \in D_i} R(x, r), \]  

(6)

where \( R \) denotes the reconstruction error. In the case of linear representation, it is the Euclidean distance, whereas in the case of nonlinear representation, it is the cross-entropy loss. To avoid overfitting and penalizing the large weights due to Eq. (6), the following is the simplest form of Eq. (6):

\[ J_{AE} - w_d(\theta) = \sum_{x \in D_i} R(x, r) + \frac{1}{2}(\lambda||W||)^2, \]  

(7)

where the relative importance of regularization is controlled via...
The main purpose of adding a sparse penalty term to the SAE is to limit the activation value at the average of error and number of code-words necessary for reconstruction. This leads a network that comprises a single layer as an encoder and decoder. DAEs are some of the simplest AE variants that contain a sparsity term, which leads a network that comprises a single layer to learn a code dictionary that reduces both the reconstruction error and number of code-words necessary for reconstruction. The main purpose of adding a sparse penalty term to the SAE cost function is to limit the activation value at the average of the hidden-layer neuron [17]. Extracting sparse features from raw data is the main aim of SAEs. To achieve the sparsity of objects presentation, two methods can be used. One method is to penalize the hidden unit bias, and the other one is to directly penalize the hidden unit’s activation output, which has been discussed by [18]. Notably, [19] used sparse multi-layered AE framework for performing the auto-detection of nuclei on a set of 537 marked histopathological breast-cancer images. The entire architecture comprised one input layer, two hidden layers, and one output layer. They stacked the SAE and used Softmax for performing classification. In addition, 3,468 nodes served as input to the input layer, 400 nodes in the first hidden layer, and 255 nodes in the second hidden layer. The output from the second hidden layer was an input to the final Softmax layer, which was mapped to two classes, either 1 or 0. Generally, if the output value of a neuron is 1, it is indicated that the neuron is active. However, if the output value of a neuron is 1, it is indicated that the neuron is inactive. The aim of implementing sparsity is to bind the unwanted activation and the updated weight coefficient decay $\lambda$. The complete structure and step-by-step working mechanism of the proposed model is depicted in Figure 1 and explained in Algorithm 1.

### 3. Sparse Auto-Encoder

SAEs are some of the simplest AE variants that contain a sparsity term, which leads a network that comprises a single layer to learn a code dictionary that reduces both the reconstruction error and number of code-words necessary for reconstruction. The main purpose of adding a sparse penalty term to the SAE cost function is to limit the activation value at the average of the hidden-layer neuron [17]. Extracting sparse features from raw data is the main aim of SAEs. To achieve the sparsity of objects presentation, two methods can be used. One method is to penalize the hidden unit bias, and the other one is to directly penalize the hidden unit’s activation output, which has been discussed by [18]. Notably, [19] used sparse multi-layered AE framework for performing the auto-detection of nuclei on a set of 537 marked histopathological breast-cancer images. The entire architecture comprised one input layer, two hidden layers, and one output layer. They stacked the SAE and used Softmax for performing classification. In addition, 3,468 nodes served as input to the input layer, 400 nodes in the first hidden layer, and 255 nodes in the second hidden layer. The output from the second hidden layer was an input to the final Softmax layer, which was mapped to two classes, either 1 or 0. Generally, if the output value of a neuron is 1, it is indicated that the neuron is active. However, if the output value of a neuron is 1, it is indicated that the neuron is inactive. The aim of implementing sparsity is to bind the unwanted activation and the updated weight coefficient decay $\lambda$. The complete structure and step-by-step working mechanism of the proposed model is depicted in Figure 1 and explained in Algorithm 1.

### 4. Denoising Auto-Encoder

DAE, which are the modest alteration of standard AEs, are not trained for reconstructing their input but to denoise an artificially corrupted version of their input [20]. Whereas an over-complete regular AE can easily learn a useless identity mapping, a DAE must extract more useful features to solve the considerably more difficult denoising problem. The main role of DAEs is to reconstruct noisy data from the original data. DAEs are used for the following two purposes: encoding the noisy data and recovering the original input data from the reconstructed output data. When data encoders are stacked in different layers, they form stacked DAEs [21][22]. Notably, [23] adopted a weighted reconstruction loss function to the conventional DAE for performing noise classification in a speech-enhancement system. They stacked several weighted DAEs to construct the model. In their experiments, they performed 50 steps with the number of input nodes ranging from 50 to 100. The un-noisy data comprised 8 languages, and a white noise with the signal-to-noise ratios (SNRs) of 6, 12, and 18 dB was selected from the NTT database to train the model. The model was trained on the data of length 1 hour, and it was tested on the data of
length 8 minutes. In the literature, DAEs have been the center of interest for many researchers in different contexts, as per [12, 13]. Based on AEs and Eqs. [6] and [7], the DAE is given by the following:

\[
J_{\text{DAE}}(\theta) = E_{q_0}(x, r)[||\text{decode}(\text{encode}(x)) - r||^2]
\]

\[
= E_{q_0}(x, r)[||W^T \text{sigmoid}(W_x + b) + c - r||^2].
\]

A training input \(x \in D_n\) is first corrupted by the additive Gaussian noise of covariance. The corrupted version \(r\) is encoded into a hidden representation \(x' \in R^{d_b}\) via an affine mapping, followed by the nonlinearity of \(h = \text{encode}(r) = \text{sigmoid}(w_r + b)\), where \(x \in R^{d_h}\), \(h \in (0, 1)^d\), \(W\) is a \(d_h \times d\) matrix with \(b \in R^{d_h}\). The hidden-layer representation \(h\) is decoded into reconstruction \(x'\) via affine mapping such that \(x' = \text{decode}(h) = W^T h + c\), where \(c \in R^d\). While \(\theta = w, b, c\) is optimized to minimize the squared reconstruction error \(||x' - x||\).

5. **Contractive Auto-Decoder**

CAEs, which are the popular and effective variants of AEs, are based on the unsupervised-learning approach for the production of valuable feature representations [24]. The trained models that learned feature representations using CAE are highly robust to minor noises and small changes within the training data. CAEs are considered the extended forms of the DAEs in which contractive penalty is added to the error function of reconstruction [25]. This penalty is, in turn, used to penalize the attribute sensitivity to the variations in the inputs. Based on learning a robust feature set, [26] proposed a CAE with an unconventional regularization yielding objective function. Based on Eqs. [6] and [7], the CAE is given by the following:

\[
J_{\text{CAE}}(\theta) = \sum_{x \in D_n} L(x, r) + \frac{1}{2}(\lambda ||f(x)||) \frac{2}{F},
\]

where \(f(x)\) denotes the Jacobian matrix of encoder \(f\) at \(x\). The mapping of the feature set to be contractive in the local domain of the training data is facilitated by adding the penalty of the Frobenius norm of the encoder Jacobian, for example, the intermediary representation of features that are robust to minor variations or noise in the input data.

6. **Experimental Setup**

We performed several experiments to evaluate and then compare the feature-learning abilities of the three AEs variants, namely, SAE, DAE, and CAE. To conduct fair experiments and comparison, all the AE variants were assumed to have the same architecture with one input layer, two hidden layers, and a Softmax regression classifier in the last layer for final classification. In this study, all the experiments were conducted using an Intel core i3 CPU with 4 GB of RAM and Windows 8.1 operating system. The compiler and language used for developing and testing these algorithms was Python2.7. For fast implementation of the comparative approaches, an efficient numeric computational open-source library TensorFlow [27] was used. In our experiments, 12,000 images were considered for evaluation and comparison with some comparative models. We used 5,000 random images for validating each MNIST variation dataset. All the images comprised 0-9 digits with the size of 28 x 28 pixels. A few example images, which show the shape deformation and variation in digits appearance, are depicted in Figure 2. This variation makes it a challenging dataset to be used to test the models. The variation datasets of MNIST are standard (basic), random-noise background digits (bg-rand), and rotation and image background digits (bg-img-rot) [28]. The first MNIST basic (mnist-basic) comprises the standard MNIST dataset. All the images comprised 0-9 digits with the size of 28 x 28 pixels. In the last dataset, the variation of the background for every pixel is consistently produced in the fixed range of 0 to 255. In the last dataset, the variation factor of rotating digits and addition of random background is merged in a single dataset. In addition, six random images from each MNIST variation dataset are depicted in Figures 2, 3, and 4 respectively. All the comparative approaches are used in combination with features extractors, and the Softmax classifier is fitted in the last layer for performing final classification. In the experiments using each approach, we randomly selected sample images for half of the training and half of the testing for each run. Every instance of model running is independently repeated 10 times, and the mean value of the errors is reported in the results.
7. Results and Discussion

In experiments, the different AE variants leveraged a Softmax classifier to estimate the overall classification behavior of models based on MNIST benchmark variant datasets. Figures 5, 6, and 7 depict the results for SAE on three MNIST benchmark variant datasets. Similarly, Figures 8, 9, and 10 depict the results for DAE. Furthermore, Figures 11, 12, and 13 depict the output roc curve for CAE. Upon analyzing all the experiments, as depicted in these figures, a gradual decrease is seen in the overall classification accuracy of all the AE variants. The results of the AE variants are verified and validated in Table 1. Table 1 also shows that starting from MNIST basic subset until MNIST random background digits. This decrease is responsible for the gradual increase in the complexity and noisiness in the datasets. Other than this complexity, the CAE outperformed both SAE and DAE in terms of the class level accuracy.

8. Conclusions

We first introduced the conventional AE model and its most popular three variants, namely, SAE, DAE, and CAE. Notably, the AE is a three-layered neural-network model with one input layer having $n$ number of units that are connected to $m$ number of hidden layer units, such that $m < n < 0$. The hidden layer is then connected to the final layer that comprises $nc$ number of units, such that $n = nc$. All the three AE variants possess
Table 1. Performance evaluation of SAE, DAE, and CAE based on MNIST benchmark datasets

| Datasets          | Approach      | Execution time | Training error | Test error | Execution time | Training error | Test error | Execution time | Training error | Test error |
|-------------------|---------------|----------------|----------------|------------|----------------|----------------|------------|----------------|----------------|------------|
|                   | SAE + Softmax | 24m 25s        | 5.59           | 13.54      | 31m 25s       | 8.92           | 21.54      | 40m 25s       | 11.87          | 26.58      |
|                   | DAE + Softmax | 31m 20s        | 7.73           | 15.78      | 37m 20s       | 10.38          | 21.78      | 45m 20s       | 17.73          | 28.72      |
|                   | CAE + Softmax | 28m 55s        | 3.35           | 11.65      | 32m 55s       | 7.58           | 20.65      | 37m 55s       | 11.05          | 24.78      |

Figure 9. ROC for the DAE based on random noise background digits (bg-rand).

Figure 10. ROC for the DAE based on rotation and image background digits (bg-img-rot).

Specific properties, which generate the interests of researchers to use them for solving problems in the domain of modern machine learning. The main purpose of adding the sparse penalty term to the SAE cost function was to limit the activation value at the average of the hidden-layer neuron. The key purpose of DAE was to introduce the ability to reconstruct noisy data from the original data. The CAE-based training model could learn the feature representations that were highly robust to minor noises and small changes in the training data. The classification results of the comparative AE variants based on the MNIST dataset showed that all the AE models performed satisfactorily. Notably, SAE had faster execution time than those of DAE.
and CAE. The overall experimental results revealed that CAE outperformed SAE and DAE in terms of both training and testing errors.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgements

The authors would like to thank the Ministry of Higher Education, Malaysia, and Universiti Tun Hussein Onn Malaysia for financially supporting this Research under Trans-disciplinary Research Grant vote no. T003.

References

[1] B. Kim and J. Lee, “A deep-learning based model for emotional evaluation of video clips,” International Journal of Fuzzy Logic and Intelligent Systems, vol. 18, no. 4, pp. 245-253, 2018. https://doi.org/10.5391/IJFIS.2018.18.4.245

[2] F. Wahid, A. K. Z. Alsaeedi, and R. Ghazali, “Using improved firefly algorithm based on genetic algorithm crossover operator for solving optimization problems,” Journal of Intelligent & Fuzzy Systems, vol. 36, no. 2, pp. 1547-1562, 2019. https://doi.org/10.3233/JIFS-181936

[3] A. Palvanov and Y. I. Cho, “Comparisons of deep learning algorithms for MNIST in real-time environment,” International Journal of Fuzzy Logic and Intelligent Systems, vol. 18, no. 2, pp. 126-134, 2018. https://doi.org/10.5391/IJFIS.2018.18.2.126

[4] M. Aamir, N. M. Nawi, A. Shahzad, H. Mahdin, and M. Z. Rehman, “A new argumentative based reasoning framework with rough set for decision making,” in Proceedings of 2017 6th ICT International Student Project Conference (ICT-ISPC), Skudai, Malaysia, 2017, pp. 1-4. https://doi.org/10.1109/ICT-ISPC.2017.8075336

[5] N. M. Nawi, A. Khan, and M. Z. Rehman, “An accelerated particle swarm optimized back propagation algorithm,” Jurnal Teknologi, vol. 77, no. 28, pp. 49-53, 2015.

[6] J. Song, H. Zhang, X. Li, L. Gao, M. Wang, and R. Hong, “Self-supervised video hashing with hierarchical binary auto-encoder,” IEEE Transactions on Image Processing, vol. 27, no. 7, pp. 3210-3221, 2018. https://doi.org/10.1109/TIP.2018.2814344

[7] C. Aytekin, X. Ni, F. Cricri, and E. Aksu, “Clustering and unsupervised anomaly detection with L2 normalized deep auto-encoder representations,” in Proceedings of 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 2018, pp. 1-6. https://doi.org/10.1109/IJCNN.2018.8489068

[8] M. Zulqarnain, S. A. Ishak, R. Ghazali, N. M. Nawi, M. Aamir, and Y. M. M. Hassim, “An improved deep learning approach based on variant two-state gated recurrent unit and word embeddings for sentiment classification,” International Journal of Advanced Computer Science and Applications, vol. 11, no. 1, pp. 594-603, 2020. https://dx.doi.org/10.14569/IJACSA.2020.0110174

[9] Y. Liu, X. Feng, and Z. Zhou, “Multimodal video classification with stacked contractive autoencoders,” Signal Processing, vol. 120, pp. 761-766, 2016. https://doi.org/10.1016/j.sigpro.2015.01.001

[10] N. Lv, C. Chen, T. Qiu, and A. K. Sangaiah, “Deep learning and superpixel feature extraction based on contractive autoencoder for change detection in SAR images,” IEEE Transactions on Industrial Informatics, vol. 14, no. 12, pp. 5530-5538, 2018. https://doi.org/10.1109/TII.2018.2873492

[11] J. Geng, H. Wang, J. Fan, and X. Ma, “Deep supervised and contractive neural network for SAR image classification,” IEEE Transactions on Geoscience and
Remote Sensing, vol. 55, no. 4, pp. 2442-2459, 2017. https://doi.org/10.1109/TGRS.2016.2645226

[12] A. L. Caterini, A. Doucet, and D. Sejdinovic, “Hamiltonian variational auto-encoder,” Advances in Neural Information Processing Systems, vol. 31, pp. 8167-8177, 2018.

[13] A. Ng, “Sparse autoencoder,” CS294A Lecture notes, 2011, Available https://web.stanford.edu/class/cs294a/sparseAutoencoder_2011new.pdf

[14] P. Vincent, “A connection between score matching and denoising autoencoders,” Neural Computation, vol. 23, no. 7, pp. 1661-1674, 2011. https://doi.org/10.1162/NECO_a_00142

[15] S. Rifai, G. Mesnil, P. Vincent, X. Muller, Y. Bengio, Y. Dauphin, and X. Glorot, “Higher order contractive autoencoder,” in Machine Learning and Knowledge Discovery in Databases. Heidelberg: Springer, 2011, pp. 645-660. https://doi.org/10.1007/978-3-642-23783-6_41

[16] Y. Wang, H. Yao, and S. Zhao, “Auto-encoder based dimensionality reduction,” Neurocomputing, vol. 184, pp. 232-242, 2016. https://doi.org/10.1016/j.neucom.2015.08.104

[17] L. Zhang, Y. Lu, B. Wang, F. Li, and Z. Zhang, “Sparse auto-encoder with smoothed L1 regularization,” Neural Processing Letters, vol. 47, pp. 829-839, 2018. https://doi.org/10.1007/s11063-017-9668-5

[18] W. Liu, T. Ma, D. Tao, and J. You, “HSAE: a hessian regularized sparse auto-encoders,” Neurocomputing, vol. 187, pp. 59-65, 2016. https://doi.org/10.1016/j.neucom.2015.07.119

[19] J. Xu, L. Xiang, Q. Liu, H. Gilmore, J. Wu, J. Tang, and A. Madabhushi, “Stacked sparse autoencoder (SSAE) for nuclei detection on breast cancer histopathology images,” IEEE Transactions on Medical Imaging, vol. 35, no. 1, pp. 119-130, 2016. https://doi.org/10.1109/TMI.2015.2458702

[20] N. Chauhan and B. J. Choi, “Denoising approaches using fuzzy logic and convolutional autoencoders for human brain MRI image,” International Journal of Fuzzy Logic and Intelligent Systems, vol. 19, no. 3, pp. 135-139, 2019. https://doi.org/10.5391/IJFIS.2019.19.3.135

[21] M. Chen, K. Q. Weinberger, Z. Xu, and F. Sha, “Marginalizing stacked linear denoising autoencoders,” The Journal of Machine Learning Research, vol. 16, no. 1, pp. 3849-3875, 2015.

[22] C. Tong, J. Li, C. Lang, F. Kong, J. Niu, and J. J. Rodrigues, “An efficient deep model for day-ahead electricity load forecasting with stacked denoising auto-encoders,” Journal of Parallel and Distributed Computing, vol. 117, pp. 267-273, 2018. https://doi.org/10.1016/j.jpdc.2017.06.007

[23] B. Xia and C. Bao, “Wiener filtering based speech enhancement with weighted denoising auto-encoder and noise classification,” Speech Communication, vol. 60, pp. 13-29, 2014. https://doi.org/10.1016/j.specom.2014.02.001

[24] S. Zhang, L. Yao, and X. Xu, “AutoSVD++: an efficient hybrid collaborative filtering model via contractive autoencoders,” in Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 2017, pp. 957-960. https://doi.org/10.1145/3077136.3080689

[25] Q. Yang and F. Sun, “Small sample learning with high order contractive auto-encoders and application in SAR images,” Science China Information Sciences, vol. 61, article no. 099101, 2018. https://doi.org/10.1007/s11432-017-9214-8

[26] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, “Contractive auto-encoders: explicit invariance during feature extraction,” in Proceedings of the 28th International Conference on Machine Learning (ICML), Bellevue, WA, 2011, pp. 833-840.

[27] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, et al., “TensorFlow: a system for large-scale machine learning,” in Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI), Savannah, GA, 2016, pp. 265-283.

[28] M. Aamir, F. Wahid, H. Mahdin, and N. M. Nawi, “An efficient normalized restricted Boltzmann machine for solving multiclass classification problems,” International Journal of Advanced Computer Science and Applications, vol. 10, no. 8, pp. 416-426, 2019.
Muhammad Aamir has recently received his PhD in Information Technology from University Tun Hussien Onn Malaysia. He did his Masters degree in Computer Science from City University of Science and Information Technology Pakistan. He had worked for two years in Xululabs LLC as data scientist. Currently he is working on research related to big data processing and data analysis. His fields of interest are Data Science, Deep Learning, and Computer Programming.

Nazri Mohd Nazwi is Professor at the Faculty of Computer Science and Information Technology, University Tun Hussein Onn Malaysia. He obtained his PhD in Computer Science (Data mining) from Swansea University United Kingdom, his MSc in Computer Science from the University of Technology Malaysia. He has published more than 78 indexed journals and conference proceedings. His research interests are in the field of data analysis, database system, optimization methods and data mining techniques using Artificial Neural Network.

Hairunizam Bin Mahdin is Associate Professor at the Faculty of Computer Science and Information Technology, University Tun Hussein Onn Malaysia. He obtained his PhD in Computer Science from Deakin University Australia, his MSc and BS in Computer Science from the University Putra Malaysia. His research interests are in the field of Data management, Database system, Internet of things and Big Data.

Rashid Naseem belongs to Landikotal, Khyber, KPK, Pakistan. He received the BCS degree in computer science from the University of Peshawar, Pakistan, in 2008 and the MPhil degree in computer science from the Quaid-i-Azam University, Pakistan, in 2011. He obtained PhD in Information Technology from the Universiti Tun Hussein Onn Malaysia in February 2017. He is currently Assistant Professor of Software Engineering at Pak-Austria Fachhochschule Institute of Applied Sciences and Technology, Mang Khanpur Road Haripur, Pakistan. His research interests include software modularization, architecture recovery, datamining and clustering techniques.

Muhammad Zulqarnain received his Bachelor and Master degree in Computer Science and Information Technology from The Islamia University of Bahawalpur (IUB), Pakistan. He received his M.Phil degree (Master of Philosophy) from National College of Business Administration and Economics, Lahore, Pakistan. He is currently pursuing Ph.D from University Tun Hussein Onn Malaysia. His research interest is Machine Learning and Deep learning for natural language processing and its application.