SAMPUL

IEEE Access

Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

AN IEEE JOURNAL

Volume 8
2020

A multidisciplinary, applications-oriented, all-electronic archival journal continuously presenting the results of original research or development across all of IEEE's fields of interest. Supported by author publication fees, its hallmarks are a rapid peer review and publication process with open access to all readers.

(IAECCG)
(ISSN 2169-3536)

IEEE

DAFTAR ISI

IEEE Access

Missing-Insensitive Short-Term Load Forecasting Leveraging Autoencoder and LSTM

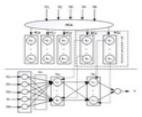
Kyungnam Park; Jaeik Jeong; Dongjoo Kim; Hongseok Kim

Publication Year: 2020 , Page(s): 206039 - 206048

Cited by: Papers (10)

Abstract HTML

A Novel Approach in Determining Neural Networks Architecture to Classify Data With Large Number of Attributes



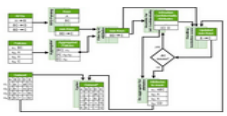
Muhammad Ibnu Choldun Rachmatullah; Judhi Santoso; Kridanto Surendro

Publication Year: 2020 , Page(s): 204728 - 204743

Cited by: Papers (4)

Abstract HTML

GDPR Compliant Information Confidentiality Preservation in Big Data Processing



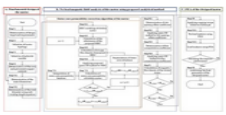
Loredana Caruccio; Domenico Desiato; Giuseppe Polese; Genoveffa Tortora

Publication Year: 2020 , Page(s): 205034 - 205050

Cited by: Papers (9)

Abstract HTML

Analysis and Design of an Axial Flux Permanent Magnet Motor for in-Wheel System Using a Novel Analytical Method Combined With a Numerical Method



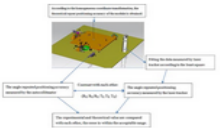
Byung-Oh Tak; Jong-Suk Ro

Publication Year: 2020 , Page(s): 203994 - 204011

Cited by: Papers (6)

Abstract HTML

Research and Experiment of Repairable Space Telescope Interface System



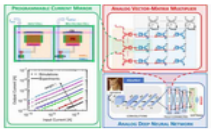
Zhen Shi; Weiguo Zhao; Libao Yang; Yang Xun; Qingya Li; Yaoyu Zhang

Publication Year: 2020 , Page(s): 225097 - 225108

Cited by: Papers (3)

Abstract HTML

Analog Vector-Matrix Multiplier Based on Programmable Current Mirrors for Neural Network Integrated Circuits



Maksym Paliy; Sebastiano Strangio; Piero Ruiu; Tommaso Rizzo; Giuseppe Iannaccone

Publication Year: 2020 , Page(s): 203525 - 203537

Cited by: Papers (14)

Abstract HTML

A Metadata-Driven Approach for Testing Self-Organizing Multiagent Systems

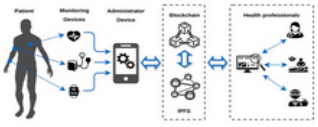
Nathalia Nascimento; Paulo Alencar; Carlos Lucena; Donald Cowan

Publication Year: 2020 , Page(s): 204256 - 204267

Cited by: Papers (4)

Abstract HTML

Preserving Privacy in Mobile Health Systems Using Non-Interactive Zero-Knowledge Proof and Blockchain



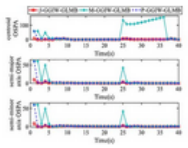
Antonio Emerson Barros Tomaz; José Cláudio Do Nascimento; Abdelhakim Senhaji Hafid; José Neuman De Souza

Publication Year: 2020 , Page(s): 204441 - 204458

Cited by: Papers (23)

Abstract HTML

A Generalized Labelled Multi-Bernoulli Filter for Extended Targets With Unknown Clutter Rate and Detection Profile



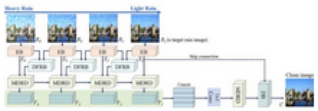
Cuiyun Li; Zehao Fan; Renzheng Shi

Publication Year: 2020 , Page(s): 213772 - 213782

Cited by: Papers (2)

Abstract HTML

Progressive Rain Removal via a Recurrent Convolutional Network for Real Rain Videos



Kyu-Ho Lee; Eunji Ryu; Jong-Ok Kim

Publication Year: 2020 , Page(s): 203134 - 203145

Cited by: Papers (3)

Abstract HTML

DEWAN REDAKSI

IEEE *Access* Editor-in-Chief

Derek Abbott

University of Adelaide, Australia

Professor with the School of Electrical and Electronic Engineering

About Professor Abbott *IEEE Fellow*

Derek Abbott (M'85–SM'99–F'05) was born in South Kensington, London, U.K. He completed his B.Sc. (Hons.) degree in physics (1982) from Loughborough University, Leicestershire, U.K. and the Ph.D. degree in electrical and electronic engineering (1995) from The University of Adelaide, Adelaide, SA, Australia.

From 1978 to 1986, he was a Research Engineer with the GEC Hirst Research Centre, London, U.K. From 1986 to 1987, he was a VLSI Design Engineer with Austek Microsystems, Australia. Since 1987, he has been with The University of Adelaide, where he is currently a full Professor with the School of Electrical and Electronic Engineering. His research interests include multidisciplinary physics and electronic engineering applied to complex systems, networks, game theory, energy policy, stochastics, and biophotonics. He co-edited Quantum Aspects of Life (Imperial College Press, 2008), and co-authored Stochastic Resonance (Cambridge Univ. Press, 2008) and Terahertz Imaging for Biomedical Applications (Springer-Verlag, 2012).

He has served as Guest Editor for IEEE J. Solid-State Circuits (1999) and Associate Editor for IEEE Photonics (2009–2014). He has served on the Editorial Board of Proceedings of the IEEE (2009–2014), the Editorial Board of IEEE *Access* (2015–Present), and he currently serves on the IEEE Publications Publication Services and Products Board (PSPB).

Prof. Abbott has received a number of awards, including an Australian Research Council Future Fellowship (2012), the David Dewhurst Medal (2015) for biomedical engineering, the Barry Inglis Medal (2018) for measurement science, and the M. A. Sargent Medal (2019) for eminence in engineering.

IEEE *Access* Editorial Board

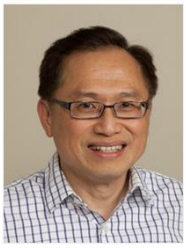
FIND ALPHABETICALLY

Sonia Aissa

Istitut National de la Recherche Scientifique (INRS), Montreal, Canada

Anuradha Annaswamy

MIT, Cambridge, Massachusetts, USA



Nirwan Ansari

New Jersey Institute of Technology, Newark, New Jersey, USA

Jun Cai

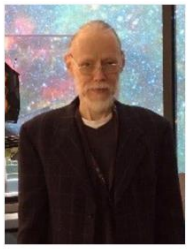
Concordia University, Canada, Montreal, Quebec, Canada

Kun-Shan Chen

Guilin University of Technology, Guangxi, China

J.-C. Chiao

Southern Methodist University, Dallas, TX, USA



John W. Evans

NASA, Washington, DC, USA

Shaikh Anowarul Fattah

Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh

Maya Gokhale

Lawrence Livermore National Laboratory, Livermore, California, USA

Chih-Lin I

China Mobile Research Institute, Beijing, China

Giuseppe Iannaccone

University of Pisa, Pisa, Italy

Yumi Iwashita

NASA-JPL California Institute of Technology, Pasadena, CA, USA

Bruce Jacob

Florida Polytechnic University, Lakeland, Florida USA

Abbas Jamalipour

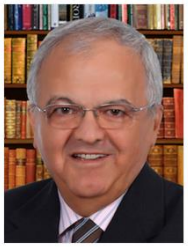
University of Sydney, Sydney, Australia

Bahram Javidi

University of Connecticut, Storrs, CT USA

Weihua Jiang

Nagaoka University of Technology,
Nagaoka, Niigata, Japan



Okyay Kaynak

Bogazici University,
Istanbul, Turkey

Germano Lambert-Torres

Gnarus Institute,
Itajuba, Brazil

Joy Laskar

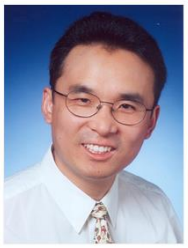
Maja Systems,
Milpitas, California, USA

Victor Leung

The University of British Columbia,
Vancouver, British Columbia, Canada

Shengtao Li

Xi'an Jiaotong University,
China



Qilian Liang

University of Texas at Arlington,
Arlington, Texas, USA

Leda Lunardi

North Carolina State University,
Raleigh, NC, USA

Michal Mrozowski

Gdansk University of Technology,
Gdansk, Poland

Michele Nappi

University of Salerno,
Fisciano, Italy

Dalma Novak

Octane Wireless,
Hanover, Maryland, USA

Hugo Proença

University of Beira Interior,
Covilhã, Portugal

Yi Qian

University of Nebraska-Lincoln,
Lincoln, Nebraska, USA

Zhihua Qu

University of Central Florida,
Orlando, Florida, USA

Susanto Rahardja

Singapore Institute of Technology/Infocomm Technology Cluster,
Singapore, Singapore

Mehrdad Saif

University of Windsor,
Windsor, Ontario, Canada

Ali Serpengüzel

Koç University,
Istanbul, Turkey

Peter Siegel

California Institute of Technology (CalTech),
Pasadena, California, USA

Ashitey Trebi-Ollennu

NASA-JPL California Institute of Technology,
Pasadena, California, USA

Daniela Tuninetti

University of Illinois Chicago,
Chicago IL, USA

Ge Wang

Rensselaer Polytechnic Institute,
Troy, New York, USA

Andreas Weisshaar

Oregon State University,
Corvallis, Oregon, USA

Enrico Zio

Mines ParisTech (France), Politecnico di Milano (Italy),
Paris, France, Milano, Italy

ARTIKEL

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 21, 2020, accepted October 29, 2020, date of publication November 9, 2020,
date of current version November 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3036853

A Novel Approach in Determining Neural Networks Architecture to Classify Data With Large Number of Attributes

MUHAMMAD IBNU CHOLDUN RACHMATULLAH, JUDHI SANTOSO, (Member, IEEE), AND KRIDANTO SURENDRO

School of Electrical Engineering and Informatics, Institut Teknologi Bandung, Bandung 40132, Indonesia

Corresponding author: Muhammad Ibnu Choldun Rachmatullah (ibnucholdun@poltekpos.ac.id)

This work was supported in part by Institut Teknologi Bandung and Ministry of Research and Technology/National Research and Innovation Agency Republic of Indonesia.

ABSTRACT One of the challenges in the successful implementation of deep neural networks (DNN) lies on the determination of its architecture, in terms of the number of hidden layers and neurons for each hidden layer. In this research, a new approach is proposed to determine the neural networks architecture especially in the form of Multi-layer Perceptron (MLP) which will later be used as a machine learning method to classify data with large number of attribute. The new approach is proposed since the previous approaches are no longer applicable as general guidelines to determine the architecture of neural networks. Thus, the proposed approach aims to determine the number of hidden layers by using principal component analysis (PCA), while the number of neurons for each hidden layer is determined by using K-Means clustering. The determined neural network architecture is utilized to classify data with large number of attribute, such as the Gas Sensor Array Drift dataset which has 128 input attributes and six output classes and the Parkinson's Disease Classification dataset which has 754 output attributes and two output classes. The results indicate that the best-performing architecture for the first dataset is the one that uses one hidden layer, with a PCA cumulative variance of 69.7%, while for the second dataset is the one that uses three hidden layers, with a PCA cumulative variance of 38.9%. Increasing the number of hidden layers does not always improve the performance of neural networks. Therefore, it is essential to determine the number of hidden layers and neurons that are appropriate to achieve good performance in neural networks. The use of PCA and K-Means clustering is expected to provide guidelines in determining neural networks architectures with good performance.

INDEX TERMS Hidden layer, hidden neurons, K-Means clustering, large attribute data, Multi-Layer Perceptron, neural networks, principal component analysis.

I. INTRODUCTION

As one of the techniques in artificial intelligence, artificial neural networks have been widely applied in various fields, such as in: medicine, engineering, and finance, due to its ability to approach arbitrary functions. Over the past 30 years, published results indicating that artificial neural networks (recognized as feedforward networks with one hidden layer) can approach all functions arbitrarily [1]–[7]. One important aspect for designing neural networks is due to its architecture having generalization capabilities [8], [9]. In some cases by using a fully connected network, and given a set of predetermined inputs and outputs, the architecture is determined by the number of layers and neurons for each hidden layer. Former literature solely discusses on how to determine the number of hidden neurons (assuming by using one hidden layer), but does rarely discuss the way to determine the optimal number of hidden layers. This fact is due to the assumption that a network with only one hidden layer is sufficient to universally approve almost all functions [3]–[7]. However, several studies probed that the application of two hidden layers provides better performance compare to one hidden layer in some cases [3]–[7].

Since the increasing ability of computers, the application of neural networks with more than one hidden layer has become one of the attractions for researchers, especially since the use

The associate editor coordinating the review of this manuscript and approving it for publication was Yongping Pan.

of deep neural networks (DNN) to solve problems in various fields. The utilization of DNN is defined as a technique applying neural networks by using numerous hidden layers between the input and output layers [3], [10]; or in other words it is considered as machine learning with DNN. One of the challenges in the successful implementation of deep neural networks lies on the determination of the network architecture, which is closely related to the number of hidden layers and hidden neurons. Determination of hidden layer and hidden neurons plays a significant role which affects the performance of deep neural networks [8], [9]. However, determination of these two amounts manually (usually through the 'trial and error' approach) to get a fairly optimal architecture, is a time consuming process.

Deep neural networks have achieved great success in practice with a large influence on the machine learning literature and artificial intelligence [11], [12]. Although it has achieved success on a practical level, the theoretical properties of architectural determination of neural networks are still investigated. Sample might include methods to find the number of hidden layers and hidden neurons in deep neural networks which are not based on the established theories [8], [12], [13]. Neural networks should be simply designed as they relate to computational time [14]. Therefore, it is necessary to find a practical method to determine the neural networks architecture.

The approach in this paper proposes the determination of neural networks architecture, which is to determine the number of hidden layers and neurons for each hidden layer. A predetermined architecture, in particular, is applied to classify data which have a large number of attributes. The increasing number of attributes in data set will lead to a more infrequent data space; a smaller distance between objects, and the reduced performance of distance-based machine learning [15]. Countless attribute data used in this research are those with more than one hundred attributes. Subsequently, this paper will only discuss the determination of neural networks architecture for classification purposes, especially for classification of data with large number of attribute. Data with ''Large Number of Attributes'' are data that have a number of input attributes between one hundred to one thousand. We used this term with the intention to differentiate it from ''high-dimensional data'' which are data whose the number of input attribute(p) outnumbers the amount of data (N), $p \gg N$ [16].

The architecture of the neural networks referred to this paper is the Multi-Layer Perceptron (MLP), which is a neural network where each neuron in the previous layer will be connected to all the neurons in the next layer. MLP architecture was chosen as the most fundamental architecture for deep learning [17]. Consequently, the approach proposed in determining MLP architecture is expected to be a guide in determining the number of hidden layers and neurons in other types of neural networks architecture.

This paper is organized as follows. Section II explains the related works. Further, section III discusses the proposed new approaches to determine the architecture of neural networks. Section IV presents the stages of experimentation. Subsequently, section V explains the results of experiments and discussions. Finally, section VI presents conclusions and future works.

II. RELATED WORKS

Prior researchers have attempted and proposed various methodologies to count the number of hidden neurons, by comparing the use of one and two hidden layers to determine the number of hidden layers and neurons in each layer. At the same time, the number of hidden layers and neurons is investigated after the use of deep neural networks. In general, research related to determine the number of hidden layers and neurons can be divided into: research that aims to only determine the number of neurons, assuming that the network has only one hidden layer; research that compares the use of one with two hidden layers; and research that determine both the number of hidden layers and neurons for each hidden layer. In the early stages of using neural networks in machine learning, the majority of researchers only focused on determining the number of neurons by assuming that neural networks have only one or two hidden layers. Several studies to count the number of hidden neurons in a neural network will be described chronologically. Arai(1993) proposed a method to determine the number of hidden neurons, where I-1 to I / 3 hidden neurons were sufficient for the purpose of binary classification [18]. Master (1993) argued that there are no theoretical reasons in using more than two hidden layers in a neural network. Likewise, no practical reasons either for it. Whereas for determining the number of hidden neurons using the geometric pyramid rule, it is stated that the number of neurons for each hidden layer will form the shape of a pyramid, where the number of neurons keeps decreasing going from input to output [19]. Tamura and Tateishi (1997) developed a method based on Akaike Information Criteria. The number of neurons in a three-layer neural network is N-1 and a four-layer neural network is N / 2 + 3 where N is the number of inputs minus the number of outputs [20]. Further, Islam and Murase (2001) proposed a large numbers of hidden neurons in the network with one hidden layer to improve performance, but the ability to generalize networks can be degraded when the number of neurons is too large resulting in false connections [21]. Zhang *et al.* (2003) applied a set of covering algorithm (SCA) in a three-layer neural network based on the number of inputs. Theoretically, the number of hidden neurons is estimated by random search [22]. Doukim *et al.* (2010) proposed a technique to discover the number of neurons in MLP neural networks by using a search technique applied in skin detection, consisting of binary and sequential search. This implementation was trained by 30 networks to discover the lowest average square error. In addition, sequential search is performed to find out the best number of hidden neurons [23]. Similarly, prior study conducted by Sheela and Deepa (2013) proposed a method to calculate the optimal number of neurons by making a list of 101 convergent

functions, and simulating them one by one to get the optimal number of neurons [24]. Whereas, prior study also calculated the optimal number of neurons by making a list of 151 convergent functions, and simulating them one by one [2].

Research comparing the use of one or two hidden layers was conducted on univariate or multivariate functions [4]–[7], [25]. Nakama(2011) compared the performance of using one and two hidden layers for linear and quadratic functions, presenting that the use of one hidden layer was faster to achieve convergence [25]. Thomas *et al.*(2016 and 2017) obtained different results, which were the use of two hidden layers with a better performance to predictive functions [4], [5]. Guliyev and Ismailov (2018) concluded that the use of one hidden layer is less favoured to approach the multivariate function; thus, the use of two hidden layers provides better performance [7], [6].

To determine the number of hidden layers and neurons, previous study broadly divided the approach into automatic and manual determination [26], [27]. Automatic determination relates to the application of models and without models. Manual determination relates to certain formulas which are performed by trial and error. Automatic determination without a model consists of grid search (GS) and random search (RS). GS is generally used to optimize DNN parameters only if the numbers are too low by selecting a range of values to explore. Further, DNN is trained for each specification along with other hyper-parameters. Random search (RS) becomes an alternative to GS, which is faster at achieving convergence. RS is not adaptive, meaning that it is not dynamically updated during the experiment (solutions do not affect the search). There are also hybrid algorithms that combine RS with other techniques to improve performance, such as manual updates provided by experts [28]–[30]. One model-based algorithm applies particle swarm optimization (PSO), which is an optimization technique by continuously counting potential solutions with a quality reference [3], [29]. This algorithm optimizes the problem by moving particles / potential solutions in the problem space with certain functions for the position and speed of the particles. The movement of particles is influenced by the best solutions of these particles, generally obtained from other particles. This set of particles is called a swarm, and eventually this swarm will move towards the best solution. One method that can be practically applied is data mining techniques, including clustering and regression [31], [32]. However, in this study, it is less explained how the relationship between the number of clusters and feature extraction are utilized in the machine learning process. Thus, subsequent researchers experience difficulty to replicate this method. Although some researchers have proposed a method to count the number of hidden layers and neurons at the same time, the method has been infrequently applied by other researchers due to its difficult application which does not match the type of dataset. Thus, several researchers prefer to apply the trial and error method [9].

III. PROPOSED APPROACH FOR NEURAL NETWORKS ARCHITECTURE DETERMINATION

Previous research has proposed an approach to determine the neural networks architecture which includes approaches such as: trial and error, rule of thumb, or other certain techniques. However, there has not been a general approach referred as a guideline by other researchers to determine the neural networks architecture for other datasets or cases since the approaches proposed by previous researchers are not generally applicable. The following will explain the reason in applying principal component analysis and clustering to determine neural networks architecture.

A. RATIONALE

The next section will explain the steps that form the rationale to determine neural networks architecture as proposed by researchers. The proposed approach would utilize principal component analysis to determine the number of hidden layers in a neural network. Based on the cumulative variance of the components, the number of components is able to be discovered, serving as the basis to determine the number of hidden layers. From each component used, clustering will be carried out to obtain the optimal number of clusters as the basis to determine the number of neurons in each hidden layer.

1) PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal Component Analysis conveys a multivariate analysis that transforms the original correlated variables into new incorrelated variables by reducing a number of these variables for smaller dimensions, but this analysis can account the diversity of the original variables. The number of principal components that are formed (PCi), where i is worth 1 to p (p = number of original variables) is equal to the number of original variables as in Figure 1 (number of original variables = 5). The principal component includes a linear combination of weighted origin variables and does not correlate with other principal components. Reduction (simplification) of dimensions is performed by the cumulative percentage criteria of variance (a %) of data explained by the first (q) principal components; or based on the eigenvalues of the principal components (Figure 1 top section marked with dashed lines). Based on Figure 1, two representative components are taken, symbolized by a box notation.

Furthermore, each data on the original variable is projected into the selected principal components (PCi). The first principal component is the component that has the highest variance, the second principal component is the component that has the second highest variance, and so on until the last component that has the lowest variance. Then, PCi will be notated as Yi. Each of these components is a linear combination of the original variables and weights.

$$Y_i = \sum_{j=1}^{p} w_{ij} x_j \tag{1}$$

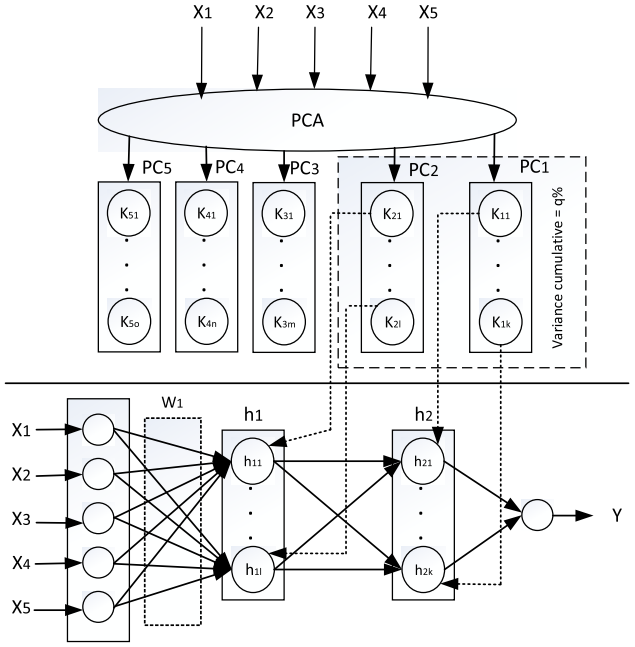


FIGURE 1. The Rationale of Proposed Neural Network Architecture Determination.

where Y_i is the i^{th} principal component (the i value is from one to the number of input attributes = p), w_{ij} is the weight for the j^{th} attribute in the i^{th} component, x_j is the j^{th} attribute. The number of principal components(Y_i) equals to the number of the original attributes. For example, if a dataset consisting of 5 input attributes is given, the PCA process will produce five principal components.

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} a = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix} x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

If y is the principal component, a is the weight matrix, and x is the original attribute, then the principal component expressed in Dirac notation is:

$$y = a|\, x\rangle \tag{2}$$

After the transformation of the original data into the equation of each component, the data of the transformation will have values between a certain range (minimum and maximum). Data with a certain range is grouped into several clusters or classes with certain techniques, such as by K-Means method. To obtain an optimal number of clusters, the Elbow criterion is then applied. Thus, clustering will be performed on data that have been transformed into a matching principal component equation so as each of these components is divided into a number of clusters, where each cluster will has their own range. Clustering results for each component are indicated by the small circle notation that in each component in Figure 1. For example the clustering of the first component, the optimal number of clusters is k clusters, the results of the clustering can be seen in Table 1. For example, the first cluster has a

TABLE 1. Example of clustering results and value ranges.

The i^{th} cluster	Value range
1	$c_1..c_2$
2	$c_3..c_4$
3	$c_5..c_6$
.	.
.	.
.	.
k-1	$c_{2k-3}..c_{2k-2}$
k	$c_{2k-1}..c_{2k}$

value range between c_1 to c_2, the second cluster has a range between c_3 to c_4, and so on. Where $c_1 < c_2 < c_3 \ldots < c_{2k-2} < c_{2k-1} < c_2 k$.

2) MULTILAYER NEURAL NETWORKS

The specific architecture of a multilayer neural network is referred as a feed-forward network, because layers consecutively feed each other in the forward direction from input to output. The default architecture of a network assumes that all nodes (neurons) in one layer are connected to each neuron of the next layer. If the neural network contains of $p_1 \ldots p_n$ neurons in each layer k; then the output is represented as a vector (column), denoted by $h_1 \ldots h_k$ with dimensions of $p_1 \ldots p_n$. Therefore, the number of neurons in each layer is called the dimension of the layer. Each neuron represents all values of the input with a certain complexity [33], [34]. As an example, on the bottom of Figure 1, it is a neural networks with 5 input attributes, 2 hidden layers, and one output layer.

According to Aggarwal (2018), hidden layers in neural networks that are closer to output, calculate more complex patterns by using functions in the previous layers. Thus, the complexity of a layer is a concatenation or accumulation of complexity from the previous layers [35] as shown in Figure 2. Layer i is hidden layer i^{th} on the neural network, where i between 1 to k, and X is the input. So, complexity(layer k) > complexity(layer k-1) > ... > complexity (layer 1).

For example, if there is a neural network consisting of k hidden layers, then it has several k+1 computational layer (including the output layer). The value (k+1) of the corresponding matrix between successive layers is denoted by $W_1 \ldots W_{k+1}$. For example, $\bar{x}$ is a column vector with dimension d that corresponds to the corresponding input attribute, $\bar{h}_1 \ldots \bar{h}_k$ means it is a column vector related to the hidden layer, and $\bar{o}$ is a column vector with dimension m representing the output. The recurrence equation for multilayer neural network is as follows:

$$\bar{h}_1 = F_1\left(W_1^T \bar{x}\right) = W_1^T \bar{x} \tag{3}$$

$$\bar{h}_{p+1} = F_{p+1}\left(W_{p+1}^T \bar{h}_p\right) = W_{p+1}^T \bar{h}_p \quad \forall p \in \{1 \ldots k-1\} \tag{4}$$

$$\begin{aligned} \bar{h}_k &= F_k\left(W_k^T \bar{h}_{k-1}\right) \\ &= W_k^T \bar{h}_{k-1} = (W_1 W_2 \ldots W_k)^T \bar{x} = W_{xk}^T \bar{x} \end{aligned} \tag{5}$$

$$\bar{o} = F\left(W_{k+1}^T \bar{h}_k\right) = W_{k+1}^T \bar{h}_k \tag{6}$$

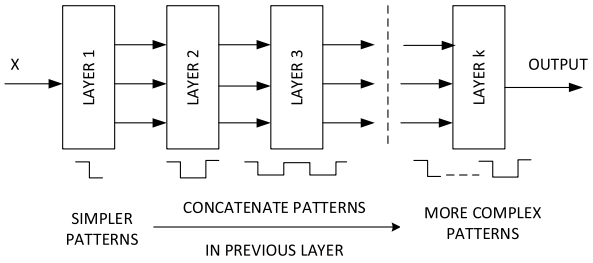


FIGURE 2. Hidden layer and pattern complexity [35].

If all the activation functions above F are identity functions [35].

3) MAPPING FROM PRINCIPAL COMPONENT ANALYSIS TO NEURAL NETWORKS

The greater number of hidden layers in neural networks represents the increasingly complex features [33]. Deep architecture in neural networks can detect features arranged at each layer. Lower layers (closer to the input layer) detect simpler features, whereas layers closer to the output layer will detect more complex features [34]. Increasingly complex features represent higher information content in PCA, represented in the principal components with high variance. In other words, the increasingly complex features of the hidden layer neural networks are aligned with the principal components with higher variance. Based on this rationale, the number of hidden layers in neural networks is in accordance with the number of components in principal component analysis. Therefore in this research, the number of hidden layers in neural networks will be determined based on the number of principal components formed through PCA.

$$\sum Variance\,(PC_i) \approx \sum Complexity(h_i) \tag{7}$$

PC_i is the principal component of PCA, h_i is hidden layer of neural networks

Each neuron at each layer on neural networks, despite producing different values, is assumed to have similar level of complexity; however, each neuron in similar layer will have different characteristic value. Each value has similar characteristics within similar group, while values that have different characteristics belong to different groups. This finding refers to the assumption that each principal component produced in PCA is aligned with the hidden layer on the neural network, where the grouping or clustering will be carried out on each principal component. The optimal number of clusters formed in each principal component is assumed to represent the number of different values in each neuron within similar hidden layer. Mapping from PCA and clustering into neural networks are depicted in Figure 1.

Based on equation 7 the cumulative variance in PCA is in line with the cumulative complexity in the neural network, and based on Figure 2, hidden layers that have higher complexity are those that are closer to the output, implying that principal components that have higher variance represent hidden layers that are closer to the output. Therefore, after determining the number of principal components to be used based on the cumulative variance, then sorting is performed as follows:

- The first principal component (that has the highest variance) represents hidden layers the closest to the output (i.e., the k^{th} hidden layer)
- The second principal component (that has the second highest variance) represents the $(k\text{-}1)^{th}$ hidden layer
- And so on successively, until the principal component that has the lowest variance represents the first hidden layer (closest to the input).

Based on equations 1 and 5, it can be said that equations 1 and 5 are compatible equations, which are a linear combination of weights and input attributes. For instance, a neural network that only utilizes one hidden layer, then

$$\bar{h}_1 = W_1^T \bar{x}$$

W_1 is the weight matrix that connects between input and neuron on the first hidden layer (in Figure 1, each weight is symbolized by a line connecting the input to the neuron). So, the linear combination results between input attributes and weight matrix lead to neuron h_{11}, h_{12},…h_{1I}, thus each of these neurons has a certain value. For instance, the first neuron on the first hidden layer:

$$h_{11} = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 \tag{8}$$

the second neuron for the first hidden layer,

$$h_{12} = w_6x_1 + w_7x_2 + w_8x_3 + w_9x_4 + w_{10}x_5 \tag{9}$$

and so on until the last neuron. The results of clustering the first component in Table 1 when connected with each neuron in the first hidden layer can be seen in Table 2.

TABLE 2. Relation between clustering results and hidden neurons.

The i^{th} cluster	Value range	Neuron
1	$c_1..c_2$	h_{11}
2	$c_3..c_4$	h_{12}
3	$c_5..c_6$	h_{13}
.	.	.
.	.	.
.	.	.
k-1	$c_{2k-3}..c_{2k-2}$	h_{1k-1}
k	$c_{2k-1}..c_{2k}$	h_{1k}

For instance, the linear combination results between W_1 weight matrix and each input attribute that has a range between $c_1..c_2$ are compatible with the first hidden neuron on the first hidden layer (h_{11}), the one with a range between $c_3..c_4$ are compatible with the second hidden neuron on the first hidden layer (h_{12}), and so on.

B. STAGES OF DETERMINING THE PROPOSED NEURAL NETWORKS ARCHITECTURE

Processing of provided available datasets consists of 3 stages by:

1. Analyzing the dataset using PCA, to obtain significant principal components.

2. Conducting clustering with the K-Means method for each principal component by varying the number of clusters.
3. Determining the optimal number of clusters for each component through Elbow criteria to obtain the optimal number of clusters for each principal component.

Algorithm of principal component analysis is performed through:

1. Calculating the average of all samples as follows:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{10}$$

2. Subtracting all data samples with the following averages:

$$D = \{d_1, d_2, \ldots, d_N\} = \sum_{i=1}^{N} x_i - \mu \tag{11}$$

3. Calculating the covariance matrix as follows:

$$\sum = \frac{1}{N-1} D\, x\, D^T \tag{12}$$

4. Calculating eigenvectors V and eigenvalues λ from the covariance matrix ($\sum$)
5. Sorting eigenvectors based on eigenvalues that correspond
6. Selecting eigenvectors with the largest eigenvalues W = $\{v_1,\ldots, v_k\}$. Eigenvectors (W) selected represent the PCA projection space.
7. Projecting all samples at a lower dimensional space than PCA (W) as follows:

$$Y = W^T D \tag{13}$$

Clustering algorithm with K-Means method is conducted to:

1. Determine the number of clusters = k
2. Determine the center point randomly as many as k
3. Calculate the distance of each data to the center of the cluster with the formula:

$$d_{ij} = \sqrt{\sum_{k=1}^{p} \{x_{ij} - x_{jk}\}^2} \tag{14}$$

4. Allocate data to the cluster based on the closest distance
5. Calculate the new cluster center which is calculated based on data in each cluster
6. Repeat steps 3 through 5, until there is no data to move the cluster

   {Calculates the total within-cluster variation with the formula

$$wss = \sum_{k=1}^{k} W(C_k) = \sum_{k=1}^{k} \sum_{x_i \in C_k} (x_i - \mu_k)^2 \tag{15}$$

   where x_i = data i^{th} in a particular cluster, and μ_k is the center of the cluster}

Algorithm of Elbow Criteria Application is performed to:

1. Run a clustering algorithm (eg K-Means) for different number of cluster (k), for example k from 1 to n
2. Calculate the wss value for each k
3. Draw a graph of wss based on the number of clusters (k)
4. Discover the locations where curves are seen in plots which are generally considered as an indicator of the correct number of clusters. In this research, the criteria were modified, which was the optimal number of clusters when three successive values of wss were relatively unchanged.

IV. EXPERIMENTATION STAGES

The stages of the experiment consist of: preparing a dataset, normalizing, setting parameters, determining the architecture, training and testing, and calculating performance.

A. DATASET PREPARATION

Actually, the proposed determination of neural network architecture can be applied to data in various real-life fields such as: finance, medicine, marketing, chemical industry, etc. This determination of architecture can also be applied to many forms of datasets such as: image, text, audio, tabular etc. or various types of datasets: numeric or non-numeric. In this research, only two datasets were exemplified, namely the dataset in the chemical industry, namely the Gas Sensor Array Drift and in the medical field, namely the Parkinson's Disease Classification dataset which was downloaded from the Machine Learning Repository website [36], [37]. Both datasets are tabular in form, adapted to the architecture to be determined which was MLP. In fact, any dataset form or type can be converted to either tabular form or numeric type. The Gas Sensor Array Drift dataset consists of 13910 data. The input attributes of this dataset were data from 16 chemical sensors, each of which had 8 attributes, while the output attribute was a concentration level divided into 6 classes. The dataset actually consisted of 10 batches; however in this study, the ten batches were combined into one. The Parkinson's Disease Classification dataset consists of 754 data, with 754 input attributes, and the output attribute consists of two classes.

B. NORMALIZATION

Normalization is performed to obtain data with the appropriate scale attributes. Normalization involves Min-Max with a value between 0 and 1. Min-Max normalization formula is as follows:

$$v' = \frac{v - min_A}{max_A - min_A} (new\_max_A - new\_min_A) + new\_min_A \tag{16}$$

v' = new data, v = old data, min_A = the smallest value of an attribute, max_A = the greatest value of an attribute, new_min_A = the smallest value of a new attribute (= 0), new_max_A = the largest value of a new attribute (= 1).

After normalization, all input and output attributes of the dataset depicted a range of values between 0 and 1.

C. PARAMETER SETTING

Besides determining the number of hidden layers and hidden neurons, there are several parameters that should be considered in determining the structure of a neural network, namely: activation function, learning rate, momentum, etc. However, determining the number of hidden layers and hidden neurons has a significant role which affects the performance of

deep neural networks [8], [9]. So, this research only aims to see changes in accuracy by changing the architecture of neural networks, namely the number of hidden layers and the number of neurons for each hidden layer, while the other parameters are fixed. Some parameters with a fixed value are learning rate, momentum and activation function, where the value uses the default ones from the application used. In this research, the machine learning process was carried out by using WEKA 3.7 software. The parameters include: learning rate, momentum, and activation functions by using the default value of WEKA 3.7.

D. DETERMINATION OF NEURAL NETWORKS ARCHITECTURE

As seen in section I, the neural network architecture to be determined in this research is MLP. It is the quintessential architecture in neural networks or deep learning [17]. To date, there is not one guidance to which the determination of neural network architecture can refer, even for the most basic architecture like MLP. Some of the methods proposed by previous researchers cannot be used as guidance by subsequent researchers because they were not generally applicable due to their funneled focus only on the number of input attributes or the number of their outputs without paying attention to variations and relationships between attributes. So, currently determining the MLP architecture is usually still performed on a trial and error basis or using the rule of thumbs, even then it only applies to certain datasets. The proposed determination of neural network architecture is expected to be the new universal guidance in determining neural network architecture, especially for MLP. In general, MLP can indeed process many forms of data such as images, tabular, texts etc. However, in certain cases, for instance, in complex image data or text sequence data, the MLP does not perform really well. Therefore, a derivative architecture of MLP was developed such as Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN). To process complex image data, CNN is usually employed. CNN is a derivative of MLP that can capture spatial features of images. So, if MLP treats image input in one-dimensional format, CNN treats image input in two-dimensional format. To process data that have sequential order, i.e. text processing, RNN is typically used. RNN is a derivative of MLP in which the hidden layer has a loop in each neuron that functions as feedback. Since MLP is the most basic architecture, and other architectures are a derivative of MLP, the guidelines for determining the MLP architecture performed in this research can be the basis for determining the CNN, RNN, and several other architectures by making some adjustments and modifications. So, this research will only discuss the determination of MLP neural network architecture. Figure 3 is an illustration of MLP. An MLP has four input attributes and two hidden layers. The first hidden layer has four neurons and the second hidden layer has three neurons. From the input attribute to the neuron, from the neuron to the neuron and from the neuron to the output attribute is connected by a line that represents a certain weight. The value δ represents the bias value for each neuron.

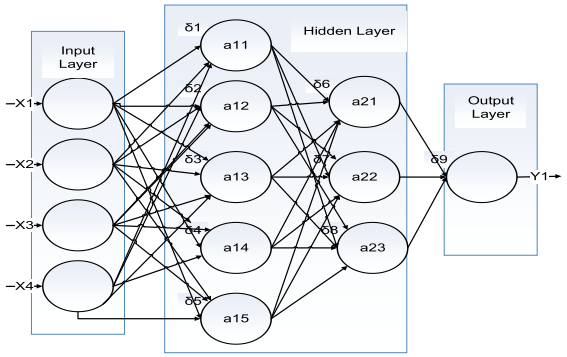


FIGURE 3. Multi-Layer Perceptron (MLP).

Neural networks architecture is determined by using principal component analysis and K-Means clustering. The use of principal component analysis is to determine the number of hidden layers. The results of PCA obtained by principal component, each of which has certain variance. Determination in the number of components is based on the number of cumulative variants of these components. For example, for the multi classification objective function, a cumulative variance of around 60% or 70 % which will be used [38], thus in order to achieve the cumulative variance, three components are required. It means that three component are used as a benchmark to determine the number of hidden layers in neural networks. Thus, the number of hidden layers is determined based on the number of components to reach the specified number of cumulative variance. For each component, clustering is conducted by K-Means through changing the number of clusters between 2 to 50; thus, the optimal number of clusters can be selected.

E. TRAINING AND TESTING

Training is conducted with 70% of data from the dataset, while testing is performed with 30% of the data from the dataset. The training and testing process for each architecture is repeated ten times by changing the initial weight value.

F. PERFORMANCE CALCULATION

Neural networks architecture performance is calculated based on the level of accuracy. Accuracy is calculated by using a formula as follows:

$$Accuracy = \frac{Number\ of\ correct\ prediction}{Number\ of\ total\ prediction} \quad (17)$$

The performance of the architecture determined using the proposed approach will be compared to one of the methods used by Master that used geometric pyramid rule [19], stating that the number of neuron for each hidden layer will form a pyramid, where the number of neurons keep decreasing going from input to output. According to Master, there are no theoretical or practical reasons to use more than two hidden layers. The number of neurons in hidden layers follow the

geometric pyramid rule, where the number of hidden neurons close to the input that is close to output will be fewer. For a neural network with a single hidden layer, the number of neurons is

$$Nh = \sqrt{n * m} \tag{18}$$

where n is the number of input features, and m is the number of output. While a neural network with two hidden layers the number of neurons can be calculated as follows:

$$r = \sqrt[3]{\frac{n}{m}} \tag{19}$$

$$\mathrm{Nh}_1 = \mathrm{mr}^2 \tag{20}$$

$$\mathrm{Nh}_2 = \mathrm{mr} \tag{21}$$

Nh_1: the number of neurons in the first hidden layer, Nh_2: the number of neuros in the second hidden layer.

V. EXPERIMENT RESULT AND DISCUSSION

In this section, the results of processing both datasets are presented including the results of PCA, K-Means clustering and Elbow criteria application to the dataset used, as well as performance comparisons.

A. GAS SENSOR ARRAY DRIFT DATASET

One of the conditions that a dataset can be analyzed by PCA is that all attributes must be numeric. Since all input attributes are already numeric, no conversion process is needed. Before carrying out PCA analysis, clustering and implementation of Elbow criterion all input attributes were normalized using equation 16 so that all attributes had a value between 0 and 1.

1) PCA AND CLUSTERING

After normalizing, both the input and output attributes have a range of values between 0 and 1. Principal component analysis is performed on all of the normalized input attributes. The results of the PCA are depicted in column 2 and 3 of Table 3. The second column demonstrates the variance values for each component, while the third column presents the cumulative variance values of the components. The first principal component has a variance of 0.697 (69.7%), where the cumulative variance is also 0.697. The second principal component has a variance of 0.144 (14.4%), where the cumulative variant is $0.697 + 0.144 = 0.841$. Likewise, the third and fourth principal components are illustrated in column 2 and 3 of Table 4. In fact, the number of components is as many as input attributes, which are 128; however in Table 4, there are only first six principal components which are depicted due to classification purposes. It means that the required cumulative variants are around 60% or 70%, fulfilled by only the first principal component. The second to the 6th component is also displayed for comparison and analysis purposes.

From each component that has been formed from PCA, clustering is performed until the optimal number of clusters is found. Clustering is conducted by the K-Means method, while the modified Elbow criteria are utilized to determine

TABLE 3. Application of clustering and the modified elbow criteria for the PCA first component of the Gas Sensor Array Drift Dataset.

N	wss	N	wss	N	wss	N	wss
2	0.292	15	0.007	27	0.002	39	0.001
3	0.173	16	0.007	28	0.002	40	0.001
4	0.106	17	0.006	29	0.002	41	0.001
5	0.067	18	0.005	30	0.002	42	0.001
6	0.045	19	0.004	31	0.002	43	0.001
7	0.037	20	0.004	32	0.002	44	0.001
8	0.027	21	0.004	33	0.001	45	0.001
9	0.020	22	0.003	34	0.001	46	0.001
10	0.015	23	0.003	35	0.001	47	0.001
11	0.012	24	0.003	36	0.001	48	0.001
12	0.009	25	0.003	37	0.001	49	0.001
13	0.009	26	0.002	38	0.001	50	0.001
14	0.008						

N=the number of cluster

TABLE 4. PCA and clustering for Gas Sensor Array Drift dataset.

Component	Variant	Cumulative	Clustering
PC 1	0.697	0.697	19
PC 2	0.144	0.841	15
PC 3	0.043	0.883	8
PC 4	0.036	0.919	10
PC 5	0.023	0.942	7
PC 6	0.014	0.956	7

the optimal number of clusters. Modification to determine the optimal number of clusters is obtained when the value of wss in a row (minimum of three in a row) is relatively unchanged. Samples of the optimal number determination of clusters for the first component is presented in Table 3, starting from the number of clusters = 19 with unchanged wss values. The results of clustering with the Elbow criteria for each component are demonstrated in column 4 of Table 4, indicating the first component the optimal number of clusters which is 19; while the second component the optimal number of clusters is 15, and so on.

Table 4 demonstrates the number of hidden layers along with their respective number of neurons. Previously, the numbers were decided based on the cumulative variance of PCA as the basis to determine the number of component to be used. For the purpose of classification functions, the cumulative variance as initialization is 60% or 70%.

The results obtained from PCA and clustering are utilized to determine the architecture of neural networks. This architecture determination consists of determination from the number of hidden layers and neurons in each hidden layer. For example: if one component with the optimal number of clusters is 19, then the neural network architecture in a hidden layer is 19 neurons. Thus, if using two components, then the number of hidden layers is two where the first hidden layer has 15 neurons and the second hidden layer has 19 neurons; and so on. For the record, the principal component with the largest variance corresponds to the hidden layer which is close to the output layer.

The results of PCA analysis from the Gas Sensor Array Drift dataset shows that achieving a cumulative variance of around 60% or 70% only requires one principal component. Therefore, the clustering process would be shown in detail only for the first component. For instance, the results of the clustering of the first principal components which were then mapped into the first hidden layer of the neural networks were shown in Table 5.

TABLE 5. The relationship between clustering results and hidden neurons for the PCA first principal component of the Gas Sensor Array Drift dataset.

The i^{th} cluster	Value range	Neuron
1	-4.733..-4.310	h_{11}
2	-4.295..-3.751	h_{12}
3	-3.558..-2.996	h_{13}
4	-2.941..-2.421	h_{14}
5	-2.406..-1.978	h_{15}
6	-1.975..-1.642	h_{16}
7	-1.638..-1.348	h_{17}
8	-1.347..-1.107	h_{18}
9	-1.106..-0.885	h_{19}
10	-0.884..-0.644	h_{110}
11	-0.643..-0.384	h_{111}
12	-0.383..-0.136	h_{112}
13	-0.135..0.092	h_{113}
14	0.093..0.3335	h_{114}
15	0.3341..0.569	h_{115}
16	0.570..0.768	h_{116}
17	0.769..0.952	h_{117}
18	0.953..1.134	h_{118}
19	1.135..1.379	h_{119}

The mapping results in Table 5 are used as a guideline for determining the architecture of neural networks, so that the architecture is obtained as shown in Figure 4. Figure 4 shows a Gas Sensor Array Drift dataset that has 128 input attributes that will be processed using machine learning using neural networks. The neural networks used have one hidden layer consisting of 19 neurons. The purpose of this machine learning is to classify the gas into six output classes.

For instance, the linear combination results between W_1 weight matrix and each input attribute that has a range between $c_1..c_2$ are compatible with the first hidden neuron on the first hidden layer (h_{11}), the one with a range between $c_3..c_4$ are compatible with the second hidden neuron on the first hidden layer (h_{12}), and so on.

2) PERFORMANCE COMPARISON

To assess the performance of neural networks by using PCA and clustering, analysis of architecture is performed based on the number of hidden layers and neurons in each layer. There are four architectures to be compared, including: architecture I: 19, architecture II: 15, 19, architecture III: 8, 15, 19, and architecture IV: 10, 8, 15, 19. Examples of explanations are: architecture I uses one hidden layer with 19 neurons; architecture II uses two hidden layers, where the first hidden layer has 15 neurons and the second hidden layer has 19 neurons, and so on. Table 6 presents the average of each architecture determined by PCA and clustering, where each

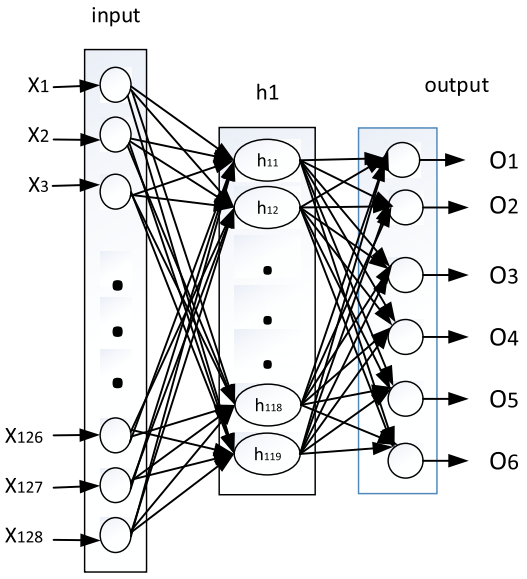


FIGURE 4. Neural Network architecture using one hidden layer for Gas Sensor Drift Array dataset.

architecture is conducted by a learning process using neural networks with 100 cycles. Each experiment is conducted ten times by changing the seeds. The column entitled Architecture indicates the number of hidden layer and the number of neuron for each architecture. For example, the column I heading depicts an architecture that consists of a hidden layer that has 19 neurons. The values in the table include the accuracy(in %) values, while the bottom row presents the average accuracy(in %) of 10 repetitions. Performance comparisons were made of architecture I, II, III, and IV.

Similarly, the experiments in Table 6 are also conducted for the number of cycles: 200, 500, and 1000. A summary of the mean values in each cycle value is presented in Table 7, while the graph is illustrated in Figure 5.

TABLE 6. Accuracy(%) of Architecture Determined by PCA and Clustering with Cycles=100 for Gas Sensor Array Drift Dataset.

No	Seed	Architecture			
		I	II	III	IV
1	0	96.7649	93.3381	88.7611	83.2734
2	1	94.728	89.3362	93.8174	83.5131
3	2	95.6626	89.9353	86.5564	88.2578
4	3	90.7021	93.5059	91.6127	84.3039
5	4	95.7824	92.6432	89.4321	74.7903
6	5	97.0046	92.4275	83.8006	74.5267
7	10	97.6755	92.2118	93.0985	84.7592
8	15	97.0525	90.127	93.8414	88.7371
9	20	92.3556	94.3686	91.2773	82.9619
10	25	96.8128	95.6147	95.5188	70.9801
	Means	95.4541	92.35083	90.77163	81.61035

TABLE 7. Average Accuracy(%) of Architecture (PCA + Clustering) for Gas Sensor Array Drift Dataset.

Seed	Architecture			
	I	II	III	IV
100	95.45	92.35	90.77	81.61
200	95.61	93.84	92.57	82.18
500	97.15	95.99	93.82	80.63
1000	96.95	96.64	96.64	77.24
1500	98.11	97.24	93.88	69.93

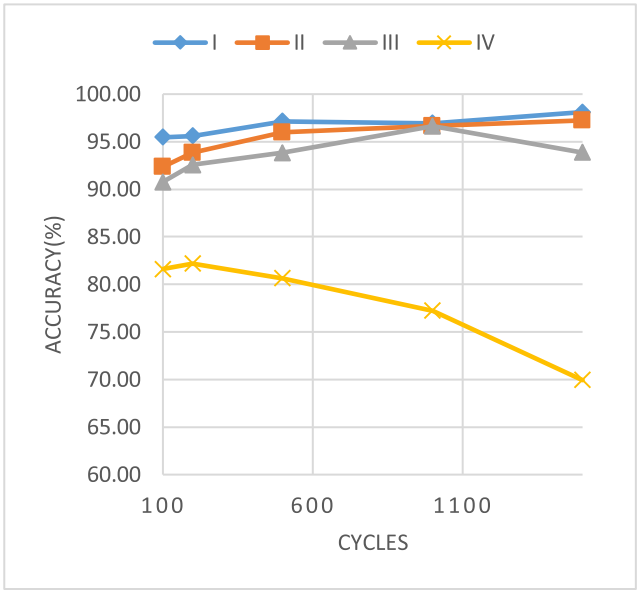


FIGURE 5. Accuracy for architectures determined by PCA and clustering for Gas Sensor Array Drift Dataset.

Figure 5 indicates architecture I which uses one hidden layer with 19 neurons with a higher accuracy value than other architectures, followed by architecture II, architecture III and IV. The graph also indicates that the increasing number of hidden layers does not always increase the accuracy value; even in this case it decreases accuracy. This is consistent with the initial assumption for classification purposes, in which the cumulative variance of PCA is 60% with one principal component of PCA. Since the required number of PCA is only one, the corresponding number of hidden layers is one (in line with the explanation in the section V.A). Increasing the number of cycles to a certain extent also tends to increase the accuracy value for architecture I and II, with only a few exceptions to architecture I, when the number of cycles = 1000, it slightly decreases accuracy. In architecture IV, more number of cycles has the tendency to decrease accuracy.

One approach to determine the architecture of neural networks to be used as the comparison against the proposed approach is the approach proposed by Master, which uses the geometric pyramid rule. This approach used one or two hidden layers. For one hidden layer, the number of hidden neurons is based on equation 18, which is:

$$Nh = \sqrt{n * m} = \sqrt{128 * 6} \approx 28$$

While for an architecture with two hidden layers, the formula for calculating the number of hidden neurons for each hidden layer can be seen in equation 19,20 and 21. Using these two formulas, then:

$$r = \sqrt[3]{\frac{128}{6}} \approx 3$$
$$\mathrm{Nh}_1 = \mathrm{mr}^2 = 6^*3^*3 = 54$$
$$\mathrm{Nh}_2 = \mathrm{mr} = 6^*3 = 18$$

Thus, for Master's approach, there are two architectures, which are architecture I: 28 and architecture II: 54,18. With the same method, the accuracy will be calculated in the same way as performed using the proposed method. Table 8 presents the average of each architecture determined by Master's method, where each architecture is conducted by a learning process using neural networks with 100 cycles. Each experiment is conducted ten times by changing the seeds.

TABLE 8. Accuracy value of the Architectures determined using Master's method with Cycles=100 for the Gas Sensor Array Drift dataset.

No	Seed	Architecture	
		I	II
1	0	95.7105	95.5907
2	1	97.364	93.9851
3	2	87.7067	92.1879
4	3	88.6892	85.5979
5	4	93.0745	91.8284
6	5	96.6211	94.3206
7	10	93.2183	93.5298
8	15	90.3666	94.6082
9	20	92.116	90.8938
10	25	85.2864	95.3271
	Means	92.01533	92.78695

Similarly, the experiments in Table 8 are also conducted for the number of cycles: 200, 500, and 1000. A summary of the mean values in each cycle value is presented in Table 9, while the graph is illustrated in Figure 6.

Figure 6 shows that architecture I which uses one hidden layer with 28 neurons tends to have a higher accuracy value than architecture II which uses two hidden layers with the number of neurons respectively 54 and 18. The graph also shows that the increase in the number of hidden layers is not always increase the accuracy value. Increasing the number of cycles up to a certain limit also tends to increase the accuracy value for architecture I and decrease when the number of

TABLE 9. Average Accuracy(%) of the Architectures determined using Master's method for Gas Sensor Array Drift Dataset.

Seed	Architecture	
	I	II
100	92.02	92.79
200	94.63	91.82
500	96.26	91.92
1000	97.67	92.92
1500	97.50	91.64

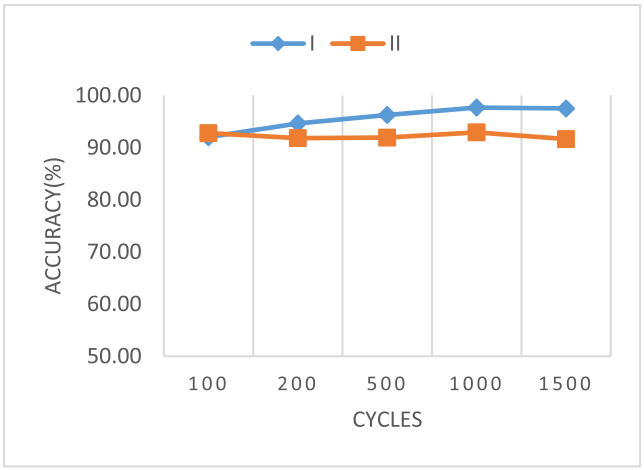


FIGURE 6. Accuracy value for the architectures determined using Master's Method for Gas Sensor Array Drift Dataset.

cycles = 1500. For architecture II, the trend of the graph, the accuracy does not increase with increasing number of cycles.

To compare the performance of Master's architecture and the proposed architecture, the architecture with the best performance was selected from each architecture. For both Master's architecture or the proposed architecture, the architecture that used one hidden layer shows the best performance. The comparison of each architecture with the best performance can be seen in Figure 7.

Figure 7 indicates that the performance of the proposed method or approach is generally better or has better accuracy than that of Master's method. Only at the number of cycles=1000 the accuracy of the proposed method was slightly lower. The highest accuracy is achieved by the proposed approach which is 98.11% which is achieved at cycles = 1500.

B. PARKINSON'S DISEASE CLASSIFICATION

One condition that a dataset can be analyzed by PCA is that all attributes must be numeric. In the Parkinson's Disease Classification dataset, there is one input attribute that is not numeric type, namely the gender attribute which is a categorical type. For this reason, the conversion is done first with the One Hot Encoding (OHE) method [39], so that previously the

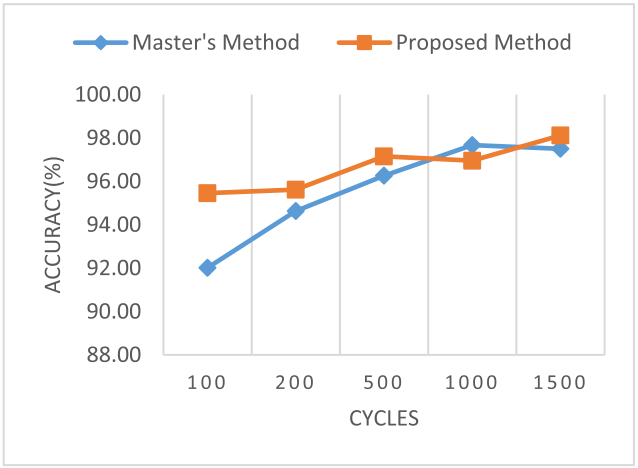


FIGURE 7. Accuracy comparison between the proposed method and Master's method for the Gas Sensor Array Drift dataset.

categorical type is converted to numeric. Before performing PCA analysis, clustering and implementation of the Elbow criterion all attributes would be normalized using equation 16 so that all attributes have a value between 0 and 1. The objective function to be applied to this dataset is a binary classification which only consisted of two output classes.

1) PCA AND CLUSTERING

After normalizing, both the input and output attributes have a range of values between 0 and 1. Principal component analysis is performed on all of the normalized input attributes. The results of the PCA are depicted in column 2 and 3 of Table 11. The second column demonstrates the variance values for each component, while the third column presents the cumulative variance values of the components. The first principal component has a variance of 0.188 (18.8%), where the cumulative variance is also 0.188. The second principal component has a variance of 0.131 (13.1%), where the cumulative variance is 0.188 + 0.131 = 0.319. Likewise, the third and fourth principal components are illustrated in column 2 and 3 of Table 11. In fact, the number of components is as many as input attributes, which are 128; however in Table 11, there are only first six principal components which are depicted. For binary classification purposes, fewer cumulative variance is required compared to multiclass classifications. Binary classification requires a cumulative variance of about 40%. The second to the 6^{th} component is also displayed for comparison and analysis purposes.

From each component that has been formed from PCA, clustering is performed until the optimal number of clusters is found. Clustering is conducted by the K-Means method, while the modified Elbow criteria are utilized to determine the optimal number of clusters. Modification to determine the optimal number of clusters is obtained when the value of wss in a row (minimum of three in a row) is relatively unchanged. Samples of the optimal number determination of clusters for the first component is presented in Table 10,

TABLE 10. Application of clustering and the modified elbow criteria for the PCA first component of the Parkinson's Disease Classification dataset.

N	wss	N	wss	N	wss	N	wss
2	0.703	15	0.016	27	0.005	39	0.002
3	0.336	16	0.015	28	0.004	40	0.002
4	0.193	17	0.013	29	0.004	41	0.002
5	0.136	18	0.011	30	0.004	42	0.002
6	0.096	19	0.009	31	0.004	43	0.002
7	0.071	20	0.008	32	0.003	44	0.002
8	0.059	21	0.008	33	0.003	45	0.002
9	0.046	22	0.007	34	0.003	46	0.002
10	0.036	23	0.007	35	0.003	47	0.002
11	0.031	24	0.006	36	0.003	48	0.001
12	0.026	25	0.006	37	0.003	49	0.001
13	0.022	26	0.005	38	0.002	50	0.001
14	0.019						

N=the number of cluster

TABLE 11. PCA and clustering for Parkinson's Disease Classification dataset.

Component	Variant	Cumulative	Clustering
PC 1	0.188	0.188	28
PC 2	0.131	0.319	23
PC 3	0.070	0.389	22
PC 4	0.059	0.447	24
PC 5	0.038	0.485	17
PC 6	0.034	0.519	19

starting from the number of clusters = 28 with unchanged wss values. The results of clustering with the Elbow criteria for each component are demonstrated in column 4 of Table 11, indicating the first component the optimal number of clusters which is 28; while the second component the optimal number of clusters is 23, and so on.

From Table 11 above, the number of hidden layers and each number of neurons can be determined. Previously, the numbers were decided based on the cumulative variance of PCA as the basis to determine the number of component to be used. For the purpose of binary classification objective function, the cumulative variance as initialization is about 40%.

The results obtained from PCA and clustering are utilized to determine the architecture of neural networks. This architecture determination consists of determination from the number of hidden layers and neurons in each hidden layer. For example: if one component with the optimal number of clusters is 28, then the neural network architecture in a hidden layer is 28 neurons. Thus, if using two components, then the number of hidden layers is two where the first hidden layer has 23 neurons and the second hidden layer has 28 neurons; and so on. For the record, the principal component with the largest variance corresponds to the hidden layer which is close to the output layer.

The results of PCA analysis from the Parkinson's Disease Classification dataset shows that achieving a cumulative variance of around 40 % requires three principal component. Mapping from the results of the clustering of PCA principal components to neurons in hidden neurons for the Parkinson's Disease Classification dataset is done in the same way as was done for the Gas Sensor Drift Array dataset but is not shown in this paper.

2) PERFORMANCE COMPARISON

To assess the performance of neural networks by using PCA and clustering, analysis of architecture is performed based on the number of hidden layers and neurons in each layer. There are four architectures to be compared, including: architecture I: 28, architecture II: 23, 28, architecture III: 22, 23, 28, and architecture IV: 24, 22, 23, 28. Examples of explanations are: architecture I uses one hidden layer with 28 neurons; architecture II uses two hidden layers, where the first hidden layer has 23 neurons and the second hidden layer has 28 neurons, and so on. Table 12 presents the average of each architecture determined by PCA and clustering, where each architecture is conducted by a learning process using neural networks with 100 cycles. Each experiment is conducted ten times by changing the seeds. The column entitled Architecture indicates the number of hidden layer and the number of neuron for each architecture. For example, the column I heading depicts an architecture that consists of a hidden layer that has 28 neurons. The values in the table include the accuracy(in %) values, while the bottom row presents the average accuracy(in %) of 10 repetitions. Performance comparisons were made of architecture I, II, III, and IV.

TABLE 12. Accuracy(%) of Architecture Determined by PCA and Clustering with Cycles=100 for Parkinson's Disease Classification dataset.

No	Seed	Architecture			
		I	II	III	IV
1	0	74.0088	61.2335	76.2115	81.4978
2	1	83.7004	84.141	85.4626	85.4626
3	2	74.4493	77.533	73.1278	80.1762
4	3	83.2599	73.1278	85.022	85.022
5	4	81.9383	84.5815	80.6167	77.533
6	5	81.4978	81.4978	84.141	85.4626
7	10	74.0088	80.1762	76.652	77.9736
8	15	78.4141	77.533	81.4978	77.533
9	20	85.022	83.2599	81.9383	83.2599
10	25	80.6167	84.141	85.022	83.7004
	Means	79.69161	78.72247	80.96917	81.76211

Similarly, the experiments in Table 12 are also conducted for the number of cycles: 200, 500, and 1000. A summary of the mean values in each cycle value is presented in Table 13, while the graph is illustrated in Figure 8.

Figure 8 indicates architecture III had higher accuracy and tended to be more stable than other architectures. Architecture III also achieves the highest accuracy compared to other architectures, namely 85.77%, achieved at cycles=1500. The graph also indicates that the increasing number of hidden

TABLE 13. Average Accuracy(%) of Architecture (PCA + Clustering) for Parkinson's Disease Classification dataset.

Seed	Architecture			
	I	II	III	IV
100	79.69	78.72	80.97	81.76
200	80.53	83.48	83.08	81.85
500	83.57	84.76	84.76	81.19
1000	82.73	85.42	85.20	84.85
1500	84.63	84.71	85.77	85.15

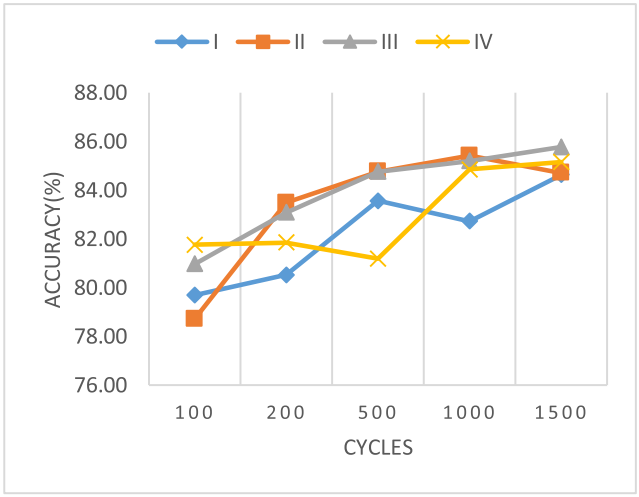


FIGURE 8. Accuracy for architectures determined by PCA and clustering for Parkinson's Disease Classification dataset.

layers does not always increase the accuracy value. This is in line with the initial hypothesis that for binary classification purposes, the cumulative variance of PCA principal components needed is around 40%. The architecture with three hidden layers corresponds to three principal components that has a cumulative variant of 38.9%.

One approach to determine the architecture of neural networks to be used as the comparison against the proposed approach is the approach proposed by Master, which uses the geometric pyramid rule. This approach used one or two hidden layers. For one hidden layer, the number of hidden neurons is based on equation 18, which is:

$$Nh = \sqrt{n * m} = \sqrt{754 * 2} \approx 39$$

While for an architecture with two hidden layers, the formula for calculating the number of hidden neurons for each hidden layer can be seen in equation 19,20 and 21. Using these two formulas, then:

$$r = \sqrt[3]{\frac{754}{2}} \approx 8$$
$$\mathrm{Nh}_1 = \mathrm{mr}^2 = 2^*8^*8 = 128$$
$$\mathrm{Nh}_2 = \mathrm{mr} = 2^*8 = 16$$

TABLE 14. Accuracy value of the Architectures determined using Master's method with Cycles=100 for Parkinson's Disease Classification dataset.

No	Seed	Architecture	
		I	II
1	0	71.3656	75.7709
2	1	85.022	84.141
3	2	74.8899	83.2599
4	3	83.2599	86.3436
5	4	80.6167	83.2599
6	5	72.2467	80.6167
7	10	77.9736	78.8546
8	15	81.0573	84.5815
9	20	85.022	84.5815
10	25	85.9031	80.1762
	Means	79.73568	82.15858

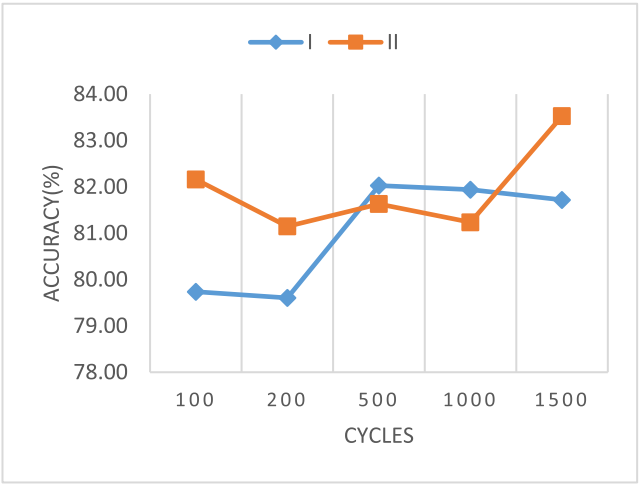


FIGURE 9. Accuracy value for the architectures determined using Master's Method for Parkinson's Disease Classification Dataset.

Thus, for Master's approach, there are two architectures, which are architecture I: 39 and architecture II: 128,16. In the same way, the accuracy will be calculated in the same way as performed using the proposed method. Table 14 presents the average of each architecture determined by Master's method, where each architecture is conducted by a learning process using neural networks with 100 cycles. Each experiment is conducted ten times by changing the seeds.

Similarly, the experiments in Table 14 are also conducted for the number of cycles: 200, 500, and 1000. A summary of the mean values in each cycle value is presented in Table 15, while the graph is illustrated in Figure 9.

Figure 9 indicates that both architecture I and architecture II has a tendency towards instability, however, architecture II achieved the highest accuracy at cycles=1500 with an accuracy value of 83.52 %. Although less stable, in general the architecture using two hidden

TABLE 15. Average Accuracy(%) of the Architectures determined using Master's method for Parkinson's Disease Classification Dataset.

Seed	Architecture	
	I	II
100	79.74	82.16
200	79.60	81.15
500	82.03	81.63
1000	81.94	81.23
1500	81.72	83.52

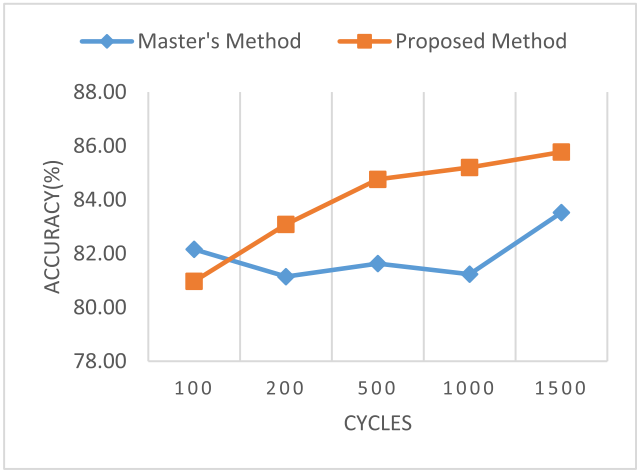


FIGURE 10. Accuracy comparison between the proposed method and Master's method for Parkinson's Disease Classification dataset.

layers with 128 and 16 neurons each has a higher level of accuracy than the architecture using one hidden layer consisting of 39 neurons. Increasing the number of cycles in either architecture I or architecture II does not always increase the accuracy value.

To compare the performance of Master's architecture and the proposed architecture, the architecture with the best performance is selected from each architecture. For Master's architecture, the highest performing architecture is the one that uses two hidden layers, while for the proposed architecture, it turned out that the architecture with three hidden layers performs the best performance. The comparison of each architecture with the best performance is presented in Figure 10.

Figure 10 shows that the performance of the proposed approach is better in general, in terms of accuracy, compared to Master's approach. Only at the number of cycles=100 the accuracy of the proposed method is slightly lower. The highest accuracy is achieved by the proposed approach which is 85.77% which is achieved at cycles = 1500.

VI. CONCLUSION AND FUTURE WORK

This paper has proposed the determination of neural networks architecture by using principal component analysis (PCA) and K-Means clustering. PCA is utilized to determine the number of hidden layers in neural networks, while K-Means clustering is applied to determine the number of hidden neurons for each hidden layer. Determination for the number of hidden layers is based on the cumulative number of variants of PCA. For multi-class classification purposes, the cumulative variance required is around 60% to 70%, while for binary classification purposes the cumulative variance required is around 40%. The architecture that has been obtained from the researcher's proposed approach aims to classify data with large number of attributes, by using Gas Sensor Array Drift and Parkinson's Disease Classification datasets.

A performance comparison is conducted between the proposed architecture and Master's architecture. In general, the proposed architecture has better performance than Master's architecture, both for the Sensor Gas Array Drift dataset and the Parkinson's Disease Classification dataset. For the Sensor Gas Array Drift dataset, the best performance of the proposed architecture is achieved when using one hidden layer that is in line with the cumulative variant of PCA of 69.7%, while for the Parkinson's Disease Classification dataset it is obtained when using three hidden layers that is in line with the cumulative variant of 38.9%. From the results of the performance comparison, it is evident that the addition of the number of hidden layers in the neural network does not guarantee the increasing accuracy for classify data with large number of attribute. Thus, the appropriateness of using the number of hidden layers is needed to obtain a high level of accuracy.

In future research, before conducting PCA analysis and clustering, it is encouraged to analyze the variation of input and output attributes, and the relationship or correlation between each input attribute and output attribute. By considering the variation and correlation of attributes, it is expected that the architectural design of neural networks that has better performance can be obtained. It is also necessary to make a comparison between the computation times of each architecture for future research. The choice of dataset is also expected to be more varied in terms of form and type, for example datasets in the form of images, text, etc. or datasets that are non-numeric. Future research can also utilize datasets from various fields such as medicine, economy, finance etc. The determination of neural network architectures using the proposed approach is expected to be the preliminary guide on how to determine other deep neural network or other deep learning architecture that are the derivative of the MLP architecture, such as CNN, RNN etc. If the determination of these derivative architecture can be achieved, someday it is hoped that there will also be an architecture that is a special form of that architecture, for example U-Net (a special form of CNN) or Attention Net (a special form of RNN). The determination of the number of hidden layers and neurons as well as altering values of other parameters in structural design of neural networks, such as: learning rate, momentum etc. would be interesting to conduct in future research.

REFERENCES

[1] M. Madhiarasan and S. N. Deepa, "A novel criterion to select hidden neuron numbers in improved back propagation networks for wind speed forecasting," *Int. J. Speech Technol.*, vol. 44, no. 4, pp. 878–893, Jun. 2016, doi: 10.1007/s10489-015-0737-z.

[2] M. Madhiarasan and S. N. Deepa, "Comparative analysis on hidden neurons estimation in multi layer perceptron neural networks for wind speed forecasting," *Artif. Intell. Rev.*, vol. 48, no. 4, pp. 449–471, Dec. 2017, doi: 10.1007/s10462-016-9506-6.

[3] B. Qolomany, M. Maabreh, A. Al-Fuqaha, A. Gupta, and D. Benhaddou, "Parameters optimization of deep learning models using Particle swarm optimization," in *Proc. 13th Int. Wireless Commun. Mob. Comput. Conf. IWCMC*, Jun. 2017, pp. 1285–1290, 2017, doi: 10.1109/IWCMC.2017.7986470.

[4] A. J. Thomas, S. D. Walters, S. M. Gheytassi, R. E. Morgan, and M. Petridis, "On the optimal node ratio between hidden layers: A probabilistic study," *Int. J. Mach. Learn. Comput.*, vol. 6, no. 5, pp. 241–247, Oct. 2016, doi: 10.18178/ijmlc.2016.6.5.605.

[5] A. J. Thomas, M. Petridis, S. D. Walters, S. M. Gheytassi, and R. E. Morgan, "Two hidden layers are usually better than one," in *Proc. Eng. Appl. Neural Netw., 18th Int. Conf. EANN*, vol. 2017, pp. 279–290, 2017.

[6] N. J. Guliyev and V. E. Ismailov, "On the approximation by single hidden layer feedforward neural networks with fixed weights," *Neural Netw.*, vol. 98, pp. 296–304, Feb. 2018, doi: 10.1016/j.neunet.2017.12.007.

[7] N. J. Guliyev and V. E. Ismailov, "Approximation capability of two hidden layer feedforward neural networks with fixed weights," *Neurocomputing*, vol. 316, pp. 262–269, Nov. 2018, doi: 10.1016/j.neucom.2018.07.075.

[8] T. Nitta, "Resolution of singularities introduced by hierarchical structure in deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2282–2293, Oct. 2017, doi: 10.1109/TNNLS.2016.2580741.

[9] A. Koutsoukas, K. J. Monaghan, X. Li, and J. Huan, "Deep-learning: Investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data," *J. Cheminformatics*, vol. 9, no. 1, pp. 1–13, Dec. 2017, doi: 10.1186/s13321-017-0226-y.

[10] J. Patterson and A. Gibson, *Deep Learning a Practitioner's Approach*. Sebastopol, CA, USA: O'Really Media, 2017.

[11] L. N. Hoang and R. Guerraoui, "Deep learning works in practice. But does it work in theory?" 2018, *arXiv:1801.10437*. [Online]. Available: https://arxiv.org/abs/1801.10437

[12] K. Kawaguchi, L. P. Kaelbling, and Y. Bengio, "Generalization in deep learning," 2017, *arXiv:1710.05468*. [Online]. Available: https://arxiv.org/abs/1710.05468

[13] S. Lee, J. Ha, M. Zokhirova, H. Moon, and J. Lee, "Background information of deep learning for structural engineering," *Arch. Comput. Methods Eng.*, vol. 25, no. 1, pp. 121–129, Jan. 2018, doi: 10.1007/s11831-017-9237-0.

[14] Z. Liu, Q. Zuo, G. Wu, and Y. Li, "An artificial neural network developed for predicting of performance and emissions of a spark ignition engine fueled with butanol-gasoline blends," *Adv. Mech. Eng.*, vol. 10, no. 1, pp. 1–11, 2018, doi: 10.1177/1687814017748438.

[15] J. M. Moreira, A. C. P. de Leon Carvalho, and T. Horváth, *A General Introduction to Data Analytics*. Puducherry, India: Wiley, 2019.

[16] T. Hastie, R. Tibshirani, and J. Friedman, "Springer Series in Statistics The Elements of Statistical Learning," *Math. Intell.*, vol. 27, no. 2, pp. 83–85, 2009, doi: 10.1007/b94608.

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[18] M. Arai, "Bounds on the number of hidden units in binary-valued three-layer neural networks," *Neural Netw.*, vol. 6, no. 6, pp. 855–860, Jan. 1993.

[19] T. Master, *Practical Neural Network Recipes in C++*. San Diego, CA, USA: Academic, 1993.

[20] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: Four layers versus three," *IEEE Trans. Neural Netw.*, vol. 8, no. 2, pp. 251–255, Mar. 1997, doi: 10.1109/72.557662.

[21] M. M. Islam and K. Murase, "A new algorithm to design compact two-hidden-layer artificial neural networks," *Neural Netw.*, vol. 14, no. 9, pp. 1265–1278, Nov. 2001, doi: 10.1016/S0893-6080(01)00075-2.

[22] Z. Zhang, X. Ma, and Y. Yang, "Bounds on the number of hidden neurons in three-layer binary neural networks," *Neural Netw.*, vol. 16, no. 7, pp. 995–1002, 2003, doi: 10.1016/S0893-6080(03)00006-6.

[23] C. A. Doukim, J. A. Dargham, and A. Chekima, "Finding the number of hidden neurons for an MLP neural network using coarse to fine search technique," in *Proc. 10th Int. Conf. Inf. Sci., Signal Process. Their Appl. (ISSPA)*, May 2010, pp. 606–609, doi: 10.1109/ISSPA.2010.5605430.

[24] K. G. Sheela and S. N. Deepa, "Selection of number of hidden neurons in neural networks in renewable energy systems," *J. Sci. Ind. Res. (India)*, vol. 73, no. 10, pp. 686–688, 2014.

[25] T. Nakama, "Systematic comparisons of single- and multiple-hidden-layer neural networks," *Neurocomputing Learn. Archit. Model.*, no. 1, pp. 147–162, 2012.

[26] P. R. Lorenzo, J. Nalepa, L. S. Ramos, and J. R. Pastor, "Hyper-parameter selection in deep neural networks using parallel particle swarm optimization," in *Proc. Genetic Evol. Comput. Conf. Companion*, Jul. 2017, pp. 1864–1871, doi: 10.1145/3067695.3084211.

[27] P. R. Lorenzo, J. Nalepa, M. Kawulok, L. S. Ramos, and J. R. Pastor, "Particle swarm optimization for hyper-parameter selection in deep neural networks," in *Proc. Genetic Evol. Comput. Conf.*, Jul. 2017, pp. 481–488, doi: 10.1145/3071178.3071208.

[28] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012, doi: 10.1162/153244303322533223.

[29] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," *ACM Int. Conf. Proc. Ser.*, vol. 227, no. 2006, pp. 473–480, 2007, doi: 10.1145/1273496.1273556.

[30] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, "Exploring strategies for training deep neural networks," *J. Mach. Learn. Res.*, vol. 10, no. 1, pp. 1–40, Jan. 2009, doi: 10.1145/1577069.1577070.

[31] M. L. Tej and S. Holban, "Determining neural network architecture using data mining techniques," in *Proc. Int. Conf. Develop. Appl. Syst. (DAS)*, May 2018, pp. 156–163, doi: 10.1109/DAAS.2018.8396089.

[32] M. L. Tej and S. Holban, "Determining optimal neural network architecture using regression methods," in *Proc. Int. Conf. Develop. Appl. Syst. (DAS)*, May 2018, pp. 180–189, doi: 10.1109/DAAS.2018.8396093.

[33] T. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.

[34] A. Bhardwaj, W. Di, and J. Wei, *Deep Learning Essentials: Your Hands-On Guide to the Fundamentals of Deep Learning and Neural Network Modeling*. Birmingham, U.K.: Packt Publishing, 2018.

[35] C. C. Aggarwal, *Neural Networks and Deep Learning*. New York, NY, USA: Springer, 2018.

[36] A. Vergara, S. Vembu, T. Ayhan, M. A. Ryan, M. L. Homer, and R. Huerta, "Chemical gas sensor drift compensation using classifier ensembles," *Sens. Actuators B, Chem.*, vols. 166–167, pp. 320–329, May 2012, doi: 10.1016/j.snb.2012.01.074.

[37] C. O. Sakar, G. Serbes, A. Gunduz, H. C. Tunc, H. Nizam, B. E. Sakar, M. Tutuncu, T. Aydin, M. E. Isenkul, and H. Apaydin, "A comparative analysis of speech signal processing algorithms for Parkinson's disease classification and the use of the tunable Q-factor wavelet transform," *Appl. Soft Comput.*, vol. 74, pp. 255–263, Jan. 2019, doi: 10.1016/j.asoc.2018.10.022.

[38] C. R. M. Ibnu, J. Santoso, and K. Surendro, "Determining the neural network topology: A review," in *Proc. 8th Int. Conf. Softw. Comput. Appl. ICSCA*, 2019, pp. 357–362, doi: 10.1145/3316615.3316697.

[39] J. T. Hancock and T. M. Khoshgoftaar, "Survey on categorical data for neural networks," *J. Big Data*, vol. 7, no. 1, pp. 1–41, Dec. 2020, doi: 10.1186/s40537-020-00305-w.

MUHAMMAD IBNU CHOLDUN RACHMATULLAH received the B.Eng. degree in informatics and the master's degree in industrial engineering from the Institute Teknologi Bandung (ITB), Indonesia, in 1995 and 2001, respectively, where he is currently pursuing the Ph.D. (by Research) degree. He is also a Lecturer of Information Systems with Politeknik Pos Indonesia. His research areas include neural networks, machine learning, and soft computing.

JUDHI SANTOSO (Member, IEEE) received the B.Sc. degree in mathematics from Institut Teknologi Bandung (ITB), Indonesia, in 1987, the master's degree in computer science from Universitas Indonesia (UI), in 1991, and the Ph.D. degree in informatics from the School of Electrical Engineering and Informatics, ITB, in 2006. He is currently an Associate Professor of Computer Science with Institut Teknologi Bandung (ITB), Indonesia. His primary research interests include modeling and simulation, soft computing, and numerical analysis.

KRIDANTO SURENDRO received the B.Eng. and master's degrees in industrial engineering from Institut Teknologi Bandung (ITB), in 1987 and 1991, respectively, and the Ph.D. degree in computer science from Keio University, Japan, in 1999. He is currently an Associate Professor of Computer Science with Institut Teknologi Bandung (ITB), Indonesia. His research interests include the field of data science, soft computing, and IT governance.