The following publication Khan, W. A., Chung, S. H., Awan, M. U., & Wen, X. (2020a). Machine learning facilitated business intelligence (Part I): Neural networks learning algorithms and applications. Industrial Management and Data Systems, 120(1), 164–195 is published by Emerald and is available at https://doi.org/10.1108/IMDS-07-2019-0361.

Machine learning facilitated business intelligence (Part 1): Neural networks learning algorithms and applications

Abstract

Purpose- The purposes of this study include: (i) to conduct a comprehensive review of the noteworthy contributions made in the area of Feedforward Neural Network (FNN) to improve its generalization performance and convergence rate (learning speed); (ii) to identify new research directions that will help researchers to design new, simple and efficient algorithms, and the users to implement optimal designed FNN for solving complex problems; (iii) to explore the wide applications of the reviewed FNN algorithms in solving real-world management, engineering and health sciences problems, and demonstrate the advantages of these algorithms in enhancing decision making for practical operations.

Design/methodology/approach- FNN has gained much popularity during the last three decades. Therefore, the authors have focused on algorithms proposed during the last three decades. The selective database was searched with popular keywords: "generalization performance", "learning rate", "overfitting" and "fixed and cascade architecture". The combination of the keywords was also used to get more relevant results. The duplicated articles in the databases, non-English, and matched keywords but out of scope, were discarded.

Findings- The authors studied in a total of 80 articles and classified them into six categories according to the nature of algorithms proposed in these articles which aim at improving the generalization performance and convergence rate of FNN. To review and discuss all the six categories in one paper is too long in length. Therefore, the authors further divided the six categories into two parts (i.e., Part I and Part II). The current paper, Part I, investigates two categories that focus on learning algorithms (i.e., Gradient learning algorithms for Network Training, Gradient free learning algorithms). Besides, the remaining four categories which mainly explores optimization techniques are reviewed in Part II (i.e., Optimization algorithms for learning rate, Bias and Variance (Underfitting and Overfitting) minimization algorithms, Constructive topology Neural Networks, Metaheuristic search algorithms). This results in a division of 80 articles into 38 and 42 for Part I and Part II, respectively. After discussing FNN algorithms with their technical merits and limitations along with real-world management, engineering, and health sciences applications for each individual category, the authors suggested seven (three in Part I and other four in Part II) new future directions to contribute in strengthening the literature.

Research limitations/implications- The FNN contribution are numerous and cannot be covered in one study. The authors remain focused on learning algorithms and optimization techniques, along with their application on real-world problems, proposed to improve the generalization performance and convergence rate of FNN with characteristics of computing optimal hyperparameters, connection weights, hidden units, selecting appropriate network architecture rather than trial and error approaches and avoiding overfitting.

Practical implications- This study will help researchers and practitioners to deeply understand FNN existing algorithms merits with limitations, research gaps, application areas, and changes in a research study in the last three decades. Moreover, the user, after having in-depth knowledge by understanding the applications of

algorithms on the real world, may apply appropriate FNN algorithms to get optimal results at the shortest possible time with less effort for their specific application area problems.

Originality/value- The existing literature surveys are limited in scope by performing algorithms comparative study, studying application areas, and focusing on a specific technique. This implies that the existing surveys are focused on studying some specific algorithms or their applications (e.g. pruning algorithms, constructive algorithms, etc.). In this work, the authors made an effort to propose a comprehensive review of different categories, along with their real-world applications, that may affect FNN generalization performance and convergence rate. This makes the classification scheme more novel and significant.

Keywords Feedforward neural network, Machine learning, Learning algorithm, Industrial management, Data analytics

1. Introduction

The widespread popularity of Feedforward Neural Network (FNN) to solve problem exists in its universal approximation capability (Ferrari and Stengel, 2005; Hornik et al., 1989; Huang, Chen, et al., 2006). It can solve any complex nonlinear problems more accurately which are difficult for classical statistical techniques (Kumar et al., 1995; Tkáč and Verner, 2016; Tu, 1996). The FNN range of applications are numerous and some areas include regression estimation (Chung et al., 2017; Deng et al., 2019; Kummong and Supratid, 2016; Teo et al., 2015), image processing (Dong et al., 2016; Mohamed Shakeel et al., 2019), image segmentation (Chen et al., 2018), video processing (Babaee et al., 2018), speech recognition (Abdel-Hamid et al., 2014), text classification (Kastrati et al., 2019; Zaghloul et al., 2009), face classification and recognition (Yin and Liu, 2018), human action recognition (Ijjina and Chalavadi, 2016), risk analysis (Nasir et al., 2019) and many others. Business intelligence makes use of data analytics techniques to generate useful information from high dimensional data that may support in making better informed decisions. Machine learning is gaining popularity in all aspect from data gathering to discovering knowledge and its role in enhancing business decisions is gaining significant interest (Bottani et al., 2019; Hayashi et al., 2010; Kim et al., 2019; Lam et al., 2014; Li et al., 2018; Mori et al., 2012; Wang et al., 2005; Wong et al., 2018). The study explores machine learning FNN and its application in facilitating business intelligence. The application of FNN in the diverse topics is not simple and extensive knowledge is required to build an optimal network to achieve intended results in the shortest possible time. In its simplest form, FNN with single hidden layer are powerful to solve many problems, given that having a sufficient number of hidden units in the layer (Nguyen and Widrow, 1990).

The importance of FNN is increasing every day due to its property of processing big nonlinear data like human brains. It discovers a hidden pattern in the data by entering raw data at the input, passing layer by layer until it arrives at the output in the forward direction. The model is trained to correctly estimate the unseen data (also known as test data) known as generalization performance of FNN. The ideal FNN is considered to have better generalization performance and may require less learning time (also known as convergence rate) to train the model. Generalization performance can be defined as the ability of an algorithm to accurately predict values on

previously unseen samples (Yeung et al., 2007), whereas, learning time can be defined as the ability of the algorithm to train model quickly. Both "generalization performance" and "learning time" are key performance indicators for FNN and used by researchers to demonstrate the effectiveness of their proposed algorithms. The major drawback that influences FNN generalization performance and learning speed (time) are:

a) trap at a local minimum when the global minimum is far away
b) faces a problem of saddle point
c) convergence decreases at plateau surface
d) network performance affected by hyperparameters initialization and adjustment
e) need trial and error approaches and expert involvements
f) repeatedly tuning of connection weights
g) hidden unit and layers adjustments.

The drawbacks can be avoided and FNN can be improved to approximate any nonlinear complex problem by the implementation of the suitable algorithms. The several reasons which become causes of above drawbacks include:

a) What should be the network size and depth i.e. shallow or deep?
b) How many hidden units should be generated by each hidden layer?
c) How many hidden layers will be sufficient for deep learning?
d) What should be network initial connection weights and learning rate?
e) How hyperparameters should be adjusted?
f) What should be the size of the dataset during network training?
g) Which learning algorithm should be implemented?
h) Which network topology is more efficient i.e. fixed or cascade?
i) What should be the criteria for increasing or decreasing the global and local hyperparameters?
j) What type of activation function to be used in hidden units?

In the literature, the answers to the above questions are not so straightforward. Researchers have proposed several learning algorithms and optimization techniques, that benefit to improve the FNN, with the main motivation to get a better generalization performance in the shortest possible network training time. In the existing literature surveys, several authors have reviewed FNN algorithms by performing a comparative study of different algorithms within the same class (for instance: constructive algorithms comparison based on data and many others), studying application area (for instance: business, engineering, and many others) and specific class survey (for instance: network ensembles survey and many others). For instance, Zhang (2000) focused on and surveyed the recent development of neural networks for classification problems. The review includes the link between the neural and conventional classifier and demonstrated that neural networks are a competitive alternative to the traditional classifiers. Other contribution includes examining the issues of posterior probability estimation, feature selection, and the trade-off between learning and generalization. Hunter et al. (2012) perform a comparative study among different types of learning algorithms and network topology to select a proper neural network size and

architecture for better generalization performance. LeCun et al. (2015) review deep learning and provide in-depth knowledge of backpropagation, convolutional neural network, and recurrent neural network. The success in deep learning is in that it requires little engineering by hand and new algorithms will accelerate its progress much more. Tkáč and Verner (2016) provide a systematic review of neural network applications during two decades and disclose that most of its application areas include financial distress and bankruptcy. Cao et al. (2018) present a survey on tuning free random weights neural network in the perspective of deep learning. The traditional deep learning iterative algorithms are far slower and have the problem of local minima. The survey suggests that the computing efficiency of deep learning increases by the combination of traditional deep learning and tuning free random weights neural network.

In the above existing studies, the focus is only on the specific type of algorithms or their applications which limits their scopes. The existing studies are more focused on comparing and selecting the suitable algorithm within their class which is solely based on expertise and available application data. It does not clearly identify the research directions over the decades. Researchers have made efforts to reduce the drawbacks by critical thinking on the above problematic question, however, a comprehensive review is missing and an open challenge to gather the answers for the above questions in one platform. Therefore, this study carried out a comprehensive literature review and classified it into six categories based on the algorithms proposed, and investigated their applications in real-world management, engineering, and health sciences problems, to understand the researchers' current interest and directions to overcome FNN drawbacks. However, to review and discuss all the six categories in one paper is too long in length. Therefore, we further divided the six categories into two parts (i.e., Part I and Part II). The current paper, Part I, investigates two categories that focus on learning algorithms (i.e., Gradient learning algorithms for Network Training, Gradient free learning algorithms). On the other hand, the remaining four categories which mainly explores optimization techniques are reviewed in Part II (i.e., Optimization algorithms for learning rate, Bias and Variance (Underfitting and Overfitting) minimization algorithms, Constructive topology Neural Networks, Metaheuristic search algorithms). Moreover, we carefully examined the real-world applications in management, engineering, and health science problems that researchers used to demonstrate the effectiveness of the proposed algorithms. Artificial benchmarking data and real-world application data are two datasets types that researchers employ for the comparative study of their proposed algorithms with other similar and popular algorithms. Growing interests to extract a useful pattern from big dimensional, nonlinear and noisy data have enforced researchers to apply and demonstrate the effectiveness and applicability of their algorithms by solving real-world problems. Our review contributes to the existing literature not only by summarizing the recent developments in FNN algorithms and classifying them into six categories according to the nature of algorithms, but also by exploring the applications of the proposed algorithms in solving real-world management, engineering, and health sciences problems and demonstrating the great potential for their practical utilization. Moreover, we propose several interesting and crucial future research directions regarding FNN which are believed to be useful for the development of the area. For the sake of simplicity, the paper entitled "Machine learning facilitated business intelligence: Neural networks optimization techniques and applications" is referred to as Part II.

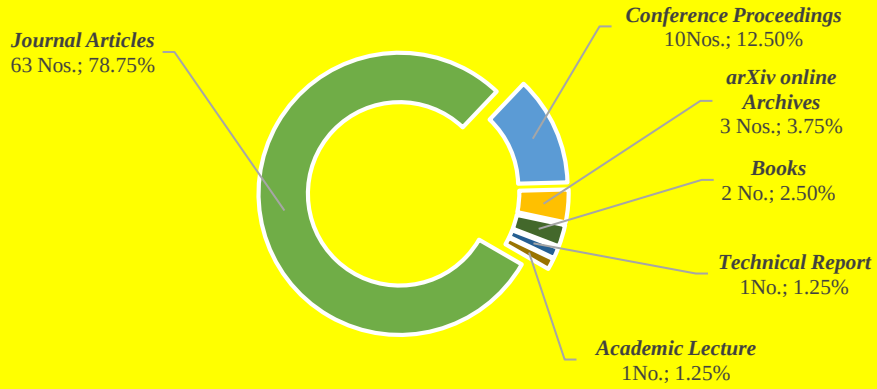

Figure 1. Articles Distribution

The paper is organized as follow: Section 2 is about survey methodology. Section 3 briefly overviews the FNN architecture. In Section 4, two categories that focus on learning algorithms are reviewed with a detail description of each algorithm in terms of its merits, limitations, and real-world management, engineering, and health sciences applications. Section 5 is about future directions to improve FNN generalization performance and learning speed. Section 6 concludes the paper.

2. Survey methodology

2.1Source of Literature

The objective of the study is to identify and classify the learning algorithms and optimization techniques that have contributed to improving the generalization performance and learning speed of FNN. Therefore, a comprehensive review has been conducted to get in-depth knowledge of the existing work and to understand researchers' contributions and work directions. Furthermore, the authors discuss the future research directions to contribute in strengthening the literature. To accomplish these objectives, the literature surveyed in the study was explored from seven different sources: IEEE Xplore -IEEE, ScienceDirect- Elsevier, Emerald Insight, arXiv- Cornell University, SpringerLink- Springer, Taylor & Francis and Google Scholar. The survey is based on articles in journals, conference proceedings, archives, technical reports, books, and academic lectures. The focus was to

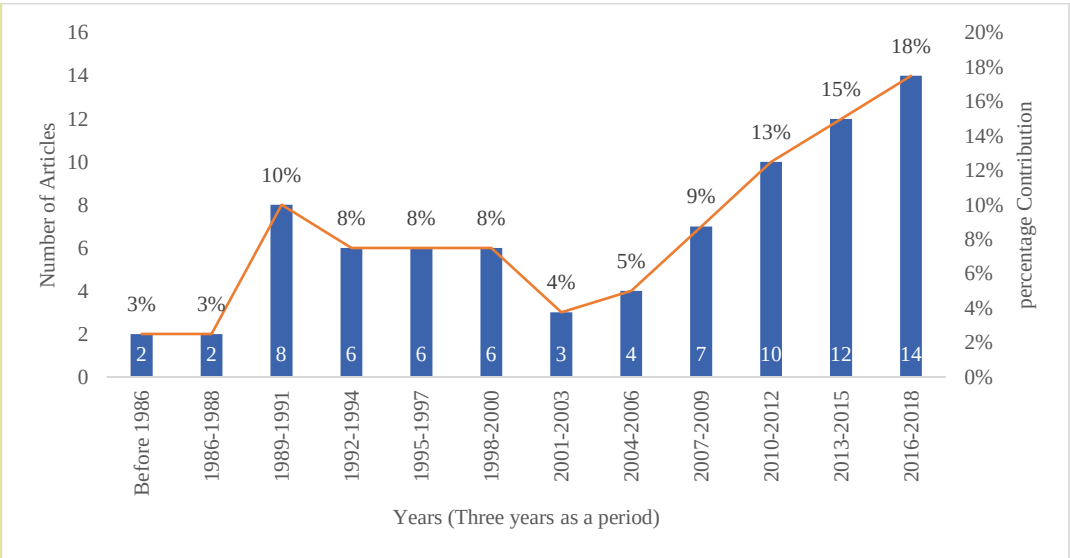

Figure 2. Articles Published over time

Source	Papers (Nos.)	Citation (Nos.)
1) Journal Article	**63**	**88144**
Artificial Intelligence	**42**	**48707**
IEEE Transactions on Neural Networks and Learning Systems	19	18778
Journal of Machine Learning Research	2	11513
Neurocomputing	6	8123
Neural Networks	6	5650
IEEE Transactions on Pattern Analysis and Machine Intelligence	3	4411
Artificial Intelligence Review	1	140
Neural Computing and Applications	3	62
Information Sciences	2	30
Multidisciplinary	**2**	**25654**
Nature	2	25654
Applied Mathematics	**6**	**6219**
Mathematics of Computation	1	3093
Technometrics	1	1752
SIAM Review	1	636
Applied Mathematics and Computation	2	543
Mathematical Programming	1	195
Arts and Humanities (Miscellaneous)	**1**	**3491**
Neural Computation	1	3491
Computer Science Applications	**7**	**3226**
IEEE Transactions on Cybernetics	2	2762
IEEE Transactions on Industrial Informatics	1	180
IEEE Transactions on Industrial Electronics	1	134
Journal of Chemical Information and Computer Sciences	1	88
IEEE Access	1	32
Industrial Management & Data Systems	1	30
Engineering (Miscellaneous)	**1**	**434**
Advances in Engineering Software	1	434
Computer Networks and Communication	**2**	**315**
IEEE Intelligent System	1	261
Neural Processing Letters	1	54
Statistics and Probability	**1**	**76**
American Statistician	1	76
Electrical and Electronics Engineering	**1**	**22**
IEEE Transactions on Circuits and Systems I: Regular Papers	1	22
2) Conference Proceedings	**10**	**41712**
IEEE	**4**	**32574**
International Symposium on Micro Machine and Human Science	1	13257
IEEE International Conference on Evolutionary Computation	1	11520
IEEE International Conference on Neural Networks	1	4793
International Joint Conference on Neural Networks	1	3004
MIT Press	**4**	**7492**
Advances in neural information processing systems	4	7492
IMLS	**1**	**1054**
International Conference on Machine Learning	1	1054
Morgan Kaufmann	**1**	**592**
Proceedings of the 1988 connectionist models summer school	1	592
3) arXiv Archive	**3**	**21519**
Cornell University Library	3	21519
4) Book	**2**	**14323**
The MIT Press	1	14009
Morgan & Claypool Publishers	1	314
5) Report	**1**	**1238**
School of Computer Science, Carnegie Mellon University	1	1238
6) Webpage	**1**	**118**
Coursera	1	118
Grand Total	**80**	**167054**

Table I. Articles Source Description

select articles published in the last three decades mainly in the period 1986-2018. However, the articles that contribute significant knowledge to the existing literature and out of time frame (For instance, the year 1985 and before) are also included to support and deepen the review.

The research contributions and its applications in FNN are numerous and cannot be covered in one study. Four keywords that are related to FNN were used to search articles in above-mentioned databases: “generalization performance”, “learning rate”, “overfitting”, and “fixed and cascade architecture”. Moreover, the combination of

keywords was also used to get relevant articles. The duplicated articles in the databases, non-English, and matched keywords but out of scope were discarded. The screening process was limited to articles belonging to Q1 category ranked journals, issued by either "Scientific Journal Rankings - SJR" or "Journal Citation Reports - JCR" in the year 2018. However, to strengthen the review, a small number of highly cited conference papers and articles belonging to Q2, Q3, and Q4 journals with more than 500Nos. citations, and articles from other sources (such as online achieves, books, technical reports, and websites) with more than 100Nos. citations were also considered. All the searched articles abstract, and the conclusion was completely reviewed along with full text to screen high-quality relevant literature. This results in a total of 80 articles, in which 38 are included in the current paper describing mainly learning algorithms and the remaining 42 are included in Part II. Figure 1 shows the distribution of the articles along with its number and percentage in each category. In 80 articles, 63 (78.75%) are journal papers, 10 (12.50%) conference papers, 3 (3.75%) online arXiv archives, 2 (2.50%) books, 1 (1.25%) technical report, and 1 (1.25%) online academic lecture. Table I explains the journals, conferences, archive, books, technical report, and academic lecture used in the literature along with a description of the type, publisher, the number of papers extracted, and citations. The content of the table illustrates the importance of screened article not only in journals but also in the conference and other sources. The main idea was to include a highly cited articles published in the reputed journals. However, a small number of articles from conferences and other sources with a high citation and unique ideas are also considered as a part of a survey to enrich the contents.

The 80 articles published over time is shown in Figure 2. It illustrates that in the year 1989 and before, the FNN was not the main research area because of the unavailability of efficient computational resources. In 1989-1994, it gains importance because of the explanation of the theory of backpropagation (BP) (Hecht-Nielsen, 1989). This created a significant interest in topic and researchers identified new research gap to improve the existing BP by proposing new learning algorithms, for instance: cascade correlation learning (Fahlman and Lebiere, 1990), probabilistic neural network (Specht, 1990) and general regression neural network (Specht, 1990). Although FNN history starts before the '50s but it gains importance in the '90s. In the modern era, the development of more efficient computational resources and the availability of big data made it a more promising research area which can be evident that the growth rate has increased from 2001 to onwards.

2.2The philosophy of the review work

The review work was conducted in five steps:

Step-1) Relevant literature explaining the learning algorithms and optimization techniques proposed to improve the generalization performance and learning speed of FNN was identified based on popular keywords used in FNN.

Step-2) Classified the algorithms into six categories. The algorithms are assigned to a category based on its problem identification, mathematical model, technical reasoning and proposed solution.

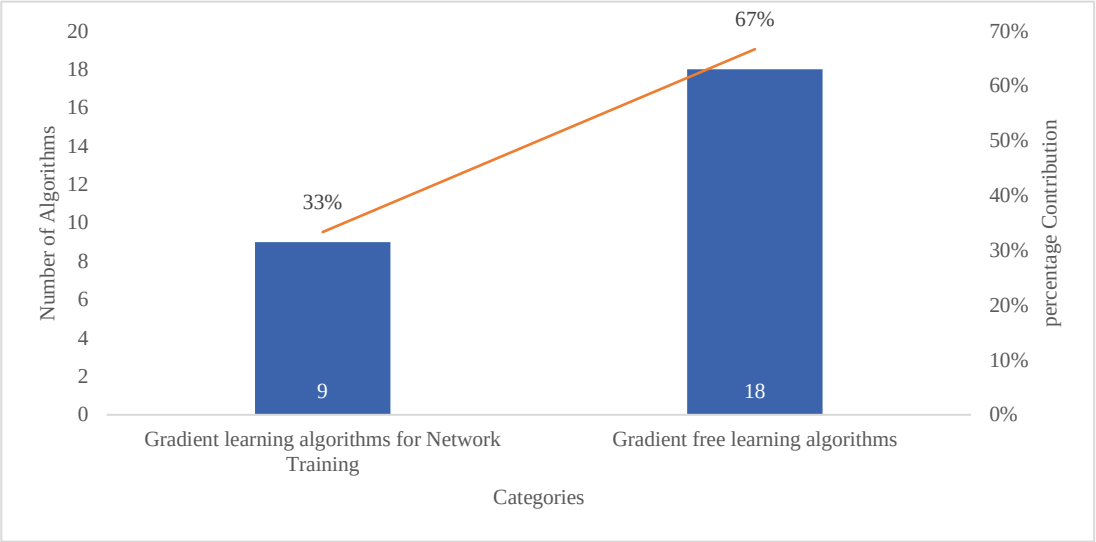

Figure 3(a). Algorithms Distribution Categories Wise

Step-3) The six categories are further classified into two main parts for the purpose of presentation. Part I review the two categories mainly exploring learning algorithms, whereas, the remaining four categories developing optimization algorithms (techniques) are reviewed in Part II.

Step-4) The algorithms are explained with their merits and technical limitations to suggest future research directions in FNN.

Step-5) The applications of the proposed algorithms in real-world are identified to show the success of FNN in management, engineering, and health sciences problem solving.

2.3Classification schemes

The classification scheme in existing literature surveyed in FNN is mainly focused on the comparative study of different algorithms within the same class (for instance: constructive algorithms comparison based on data, etc), studying application area (for instance: business, engineering, etc) and specific class survey (for instance: network

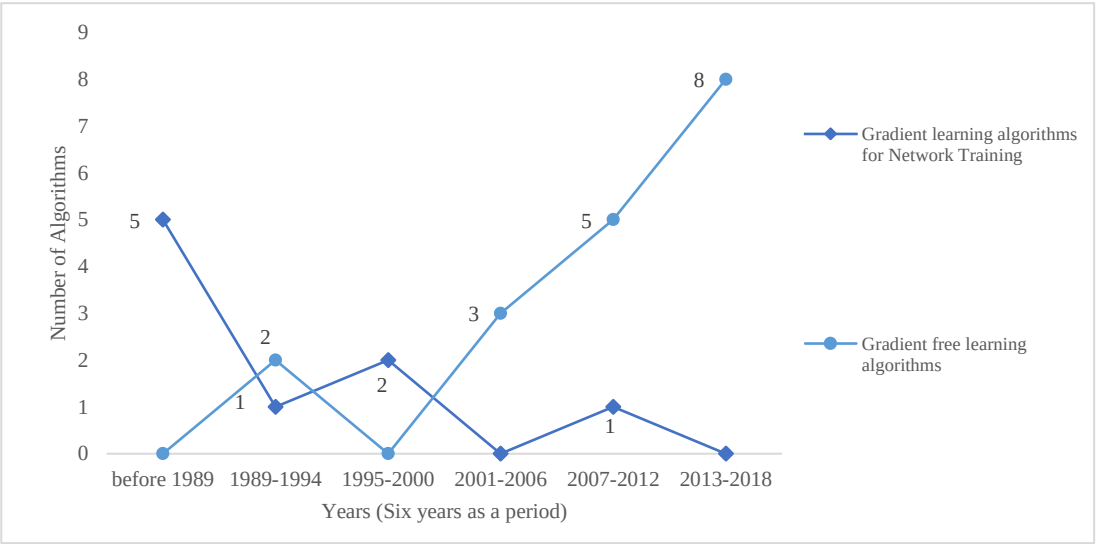

Figure 3(b). Algorithms Proposed over time

No.	Category	Algorithms Published	References
1	Gradient learning algorithms for Network Training	Gradient descent, stochastic gradient descent, mini-batch gradient descent, Newton method, Quasi-Newton method, conjugate gradient method, Quickprop, Levenberg-Marquardt Algorithm, Neuron by Neuron	(Hecht-Nielsen, 1989), (Bianchini and Scarselli, 2014), (LeCun et al., 2015), (Wilamowski and Yu, 2010), (Rumelhart et al., 1986), (Wilson and Martinez, 2003), (Wang et al., 2017), (Hinton et al., 2012), (Ypma, 1995), (Zeiler, 2012), (Shanno, 1970), (Lewis and Overton, 2013), (Setiono and Hui, 1995), (Fahlman, 1988), (Hagan and Menhaj, 1994), (Wilamowski et al., 2008), (Hunter et al., 2012)
2	Gradient free learning algorithms	Probabilistic Neural Network, General Regression Neural Network, Extreme learning machine (ELM), Online Sequential ELM, Incremental ELM (I-ELM), Convex I-ELM, Enhanced I-ELM, Error Minimized ELM (EM-ELM), Bidirectional ELM, Orthogonal I-ELM (OI-ELM), Driving Amount OI-ELM, Self-adaptive ELM, Incremental Particle Swarm Optimization EM-ELM, Weighted ELM, Multilayer ELM, Hierarchical ELM, No propagation, Iterative Feedforward Neural Networks with Random Weights	(Huang et al., 2015), (Ferrari and Stengel, 2005), (Specht, 1990), (Specht, 1991), (Huang, Zhu, et al., 2006), (Huang et al., 2012), (Liang et al., 2006), (Huang, Chen, et al., 2006), (Huang and Chen, 2007), (Huang and Chen, 2008), (Feng et al., 2009), (Yang et al., 2012), (Ying, 2016), (Zou et al., 2018), (Wang et al., 2016), (Han et al., 2017), (Zong et al., 2013), (Kasun et al., 2013), (Tang et al., 2016), (Widrow et al., 2013), (Cao et al., 2016)

Table II. Classification of FNN Published Algorithms

ensembles survey, etc). This study classification is unique as its focus is on learning algorithms and optimization techniques recommended in the last three decades to improve the generalization performance and learning speed of FNN. The algorithms are classified into six categories and further divided into two main parts. The current paper, Part I, includes the two categories mainly discussing the learning algorithms proposed to improve the generalization performance and learning speed of FNN. The two categories discussed in the current paper are:

1. Gradient learning algorithms for Network Training
2. Gradient free learning algorithms

The first category explains gradient learning algorithms that need first order or second order gradient information, whereas, the second category explains gradient free learning algorithms which analytically determine connection weights rather than first or second order gradient tuning. Figure 3(a) illustrates that authors identified in a total of 27 unique algorithms proposed in 38 articles. Figure 3(b) illustrates the number of algorithms identified in each category over time. Other than proposed algorithms, a small number of papers that support or criticize identified

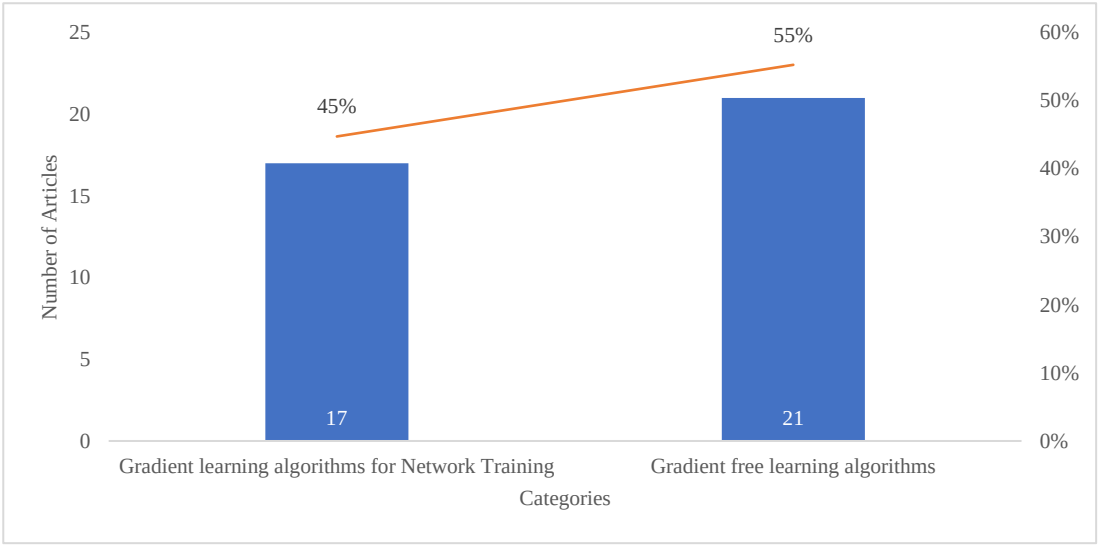

Figure 3(c). Papers Distribution Categories Wise

algorithms were also included to widen review. The unique algorithms, supportive and criticized papers result in a total of 38 articles. Figure 3(c) illustrates the total number of papers reviewed in each category. Table II provides a detailed summary of the algorithms identified in each category along with references to the total number of papers reviewed. The distribution in the figure and classification table explains the researchers' interest and trend in a specific category. The interest and trend seem to be shifting towards gradient free learning algorithms compared to gradient learning algorithms.

3. Feedforward Neural Network: An overview

FNN is a parallel information processing structure consisting of a processing element known as neurons (hidden units), interconnected together with unidirectional distributed channels known as connections. Each processing neuron receives an incoming connection from all input features, sum and activate it using nonlinear activation function, and branch it to as many connections as desired. The processing neuron output which can be of any mathematical type depends upon input features with its weighted sum and activation function (Hecht-Nielsen, 1989). The FNN concept originated by motiving from the human brain neuron functioning system. The human brain has approximately 100 billion neurons that communicate through thousands of electro-chemical connections and send signals to other neurons if the sum of connections exceeds a certain threshold. In this perspective, a simple FNN consists of a minimum of three layers interconnected by unidirectional channels known as connection weights: an input layer, hidden layer, and the output layer. FNN information process in one direction starting from the input layer, through the hidden units in the hidden layer and finally the output layer without any loop or cycle. Figure 4 illustrates a simple FNN with three layers. The number of hidden layers in FNN determines its architecture. FNN with one hidden layer is known as a shallow type, whereas more than one hidden layers are known as deep type (Bianchini and Scarselli, 2014; LeCun et al., 2015). The input layer consisting of input features x with an added bias b^u are connected to the hidden layers u through input connection weights w^{icw}. The hidden layer sums the product ($w^{icw}x$) and squash it through a nonlinear activation function $f^{hu}(z)$. The hidden layer with an added bias b^o and the output layer are connected by the output connection weight w^{ocw}. The output layer sums the product ($w^{ocw}f^{hu}(z)$) and squash it through a nonlinear activation function $f^{ou}(z)$ to estimate vector p. The p at the output layer is compared with target vector a and loss function E is determined. These all steps proceed in a forward phase and known as the *forward propagation*. This can be expressed mathematically as:

$$p = f^{ou}(w^{ocw}f^{hu}\left(w^{icw}x + b^u\right) + b^o) \tag{1}$$

Such that:

$$f(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

Equation (2) is a commonly used type of nonlinear activation function known as the sigmoid activation function. Other various types of activation function are differentiable such as hyperbolic tangent, rectified linear unit, leaky rectified linear unit, SoftMax and many others, and nondifferentiable such as a threshold and many others. The suitable choice of activation function in the hidden unit changes with the application problem under consideration. The FNN attempt to minimize the loss function E of the network by accomplishing p approximately equal to a:

$$E = \frac{1}{m}\sum_{h=1}^{m}(p_h - a_h)^2 \tag{3}$$

At each instance h the error e_h can be expressed as:

$$e_h = p_h - a_h \tag{4}$$

If E is larger than predefined expected error ε, the connection weights are backpropagated by taking derivative of E with respect to each weight in the direction of descending gradient. This update the connection weights in gradient descent direction so that p starts to become closer to a. The backward steps to calculate gradient information and updating weight to minimize error function is known as *backpropagation (BP)*. The forward propagation and backpropagation complete one iteration i and are known as FNN training. After each iteration, the error function E_i is recalculated and compared with E_{i-1}. If i reaches to its predefined maximum limit or E_i converges/start increasing the training is stopped, else continued. The training of FNN is influenced by several reasons as highlighted in the introduction section and may be overcome by various experimental trails with appropriate learning algorithms and optimization techniques for fast and efficient convergence.

4. Learning Algorithms

In this section, the authors made an effort to uncover the answers to the questions highlighted in an introduction section. The proposed algorithms in the selected literature are reviewed to understand the inspiration, research gaps, merits, limitations, and application areas. In current practice, any single algorithm in FNN is not sufficient for all types of applications. There is always a trade-off between network generalization performance and learning

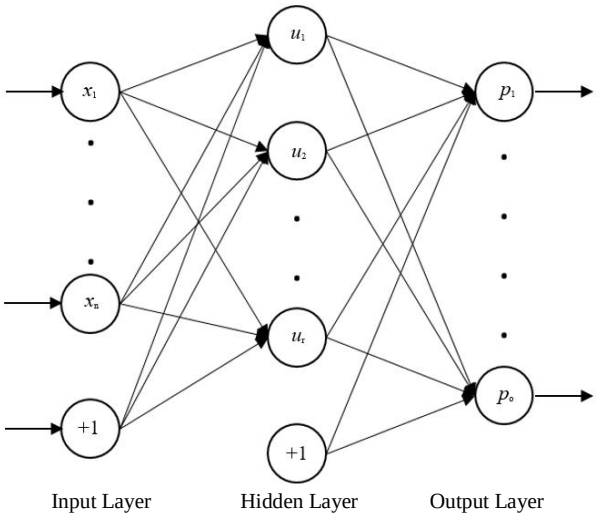

Figure 4. Feedforward Neural Networks with three layers (Input, hidden and Output)

speed. Some algorithms have the advantage of more efficient than others but maybe constrained by memory requirement, complex architecture, and/or more learning time. Since FNN development, many improvements have been made and many of them are mentioned in the below categories:

4.1Gradient learning algorithms for Network Training

The first category in this paper is the learning algorithms proposed based on the BP gradient information concept, which is considered the reason for creating significant interest in the FNN topic. The BP gradient learning algorithms can be further subcategorized into two types: *first order and second order*. The first-order gradient trains the FNN by calculating the gradient information and update weight to reach a minimum of a loss function. The first order derivative of the error with respect to weight is calculated at the output layer at each iteration and distributed back to the whole network. However, the first order BP is considered slow because of the computation of first-order gradient information at each iteration. This increase the learning time and possibility of the algorithm to stuck at a local minimum. Researchers made efforts to improve the learning speed by incorporating second-order gradient information to reach the loss function faster. Wilamowski and Yu (2010) explained that first order learning methods might need an excessive number of hidden units and iterations for convergence which can reduce their generalization performance for unseen data. Whereas second-order learning algorithms are powerful to learn but its complexity increases with increasing network size. A lot of computational memory is needed to store Jacobian **J** and Hessian Matrix **H** along with their inverse which can make it difficult for large training datasets.

4.1.1 First order gradient algorithms

The popular and well known first-order learning algorithms among the class of BP family is *Gradient descent (GD)*. It backpropagates the error with respect to connection weights *w* through layers to minimize a loss function E until it converges to minimal error (Rumelhart et al., 1986) :

$$\nabla E = \frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial out_i}\frac{\partial out_i}{\partial net_i}\frac{\partial net_i}{\partial w_i} \tag{5}$$

where ∇E is a partial derivative of the error with respect to weight w, out_i is the activation output and net_i is the weighted sum of inputs of the hidden unit. The updated weight w_{i+1} needed for the next iteration can be expressed as:

$$w_{i+1} = w_i - \propto \nabla E \tag{6}$$

where $\propto$ is a learning rate hyperparameter. For the sake of simplicity in this study, the connection weights *w* in the equations refer to all types of connection weights in the network, unless otherwise specified. The GD convergence is considered to be slow because the loss function is computed based on the whole training dataset. Increase in data size makes it slower and more time consuming for large application dataset (Wilson and Martinez, 2003). Therefore, an improved form of GD known as *Stochastic GD (SGD)* was proposed to compute the gradient on each training data instance. If the loss function is convex it converges faster than GD to a global minimum, otherwise, a local minimum is guaranteed (Wang et al., 2017). The issue with SGD is that its convergence is not

smooth compared to GD and overshoots during training on some iterations which may be controlled to some extent by adjustment of learning rate. The drawbacks of both GD and SGD can be overcome by *Mini Batch GD* that takes equal size mini-batches *N* of training data instead of whole training data or single training instance. Mini batch SD is more favorable due to its hybrid characteristics: stable and better convergence rate (Hinton et al., 2012).

For GD:

$$w_{i+1} = w_i - \propto \nabla E, (x, a) \tag{7}$$

For SGD:

$$w_{i+1} = w_i - \propto \nabla E, (x_h, a_h) \tag{8}$$

For Mini batch GD:

$$w_{i+1} = w_i - \propto \nabla E, (x_{h:h+N}, a_{h:h+N}) \tag{9}$$

The drawbacks of first-order GD and its variants algorithms (SGD and Mini batch GD) are that the number of iterations comparatively increases which make them far slower and stuck at a local minimum.

4.1.2 Second order gradient algorithms

The convergence problem of the first-order gradient was improved by applying the second order form. *Newton method (NM)*, second order derivative, was proposed to increase the convergence by modifying GD to second order Hessian inverse matrix $\mathbf{H^{-1}}$ along with first order ∇E to take larger steps towards the minimum of an objective function (Ypma, 1995):

$$w_{i+1} = w_i - \propto \mathbf{H^{-1}} \nabla E \tag{10}$$

NM make use of second order **H** and its inverse $\mathbf{H^{-1}}$ to minimize loss function which makes it computationally expensive and unfeasible for real large model applications (Zeiler, 2012). The small networks with fewer parameters may be trained with NM to take advantage of better convergence speed compared to GD. *Quasi Newton method (quasi NM)* was proposed to address the drawback of NM and simplified by approximating the inverse of the Hessian matrix **H** from the first order derivative. It updates the approximated **H** and its inverse $\mathbf{H^{-1}}$ after each iteration which make it computationally less expensive compared to NM (Shanno, 1970). Several techniques such as Davidin-Fletcher-Powell (DFP), Broyden–Fletcher–Goldfarb–Shanno (BFGS), Limited-memory BFGS (L-BFGS), Broyden's, Symmetric Rank 1 (SR1) and many other have been purposed in the literature to approximate **H** and its $\mathbf{H^{-1}}$ indirectly from the first order derivative of the loss function. Among all, BFGS has gain much popularity in the applications (Lewis and Overton, 2013). It computes **H** and $\mathbf{H^{-1}}$ expressed as:

$$\mathbf{H_{i+1}} = \mathbf{H_i} + \frac{o_i o_i^T}{o_i^T \sigma_i} - \frac{\mathbf{H_i} \sigma_i \sigma_i^T \mathbf{H_i}^{\boldsymbol{T}}}{\sigma_i^T \mathbf{H_i} \sigma_i} \tag{11}$$

Similarly, it's inverse $\mathbf{H^{-1}}$can be calculated from Sherman-Morrison formula:

$$\mathbf{H_{i+1}^{-1}} = \left(I - \frac{\sigma_i o_i^T}{o_i^T \sigma_i}\right) \mathbf{H_i^{-1}} \left(I - \frac{o_i \sigma_i^T}{o_i^T \sigma_i}\right) + \frac{\sigma_i \sigma_i^T}{o_i^T \sigma_i} \tag{12}$$

Such that:

$$o_i = \nabla E_{i+1} - \nabla E_i \tag{13}$$

$$\sigma_i = w_{i+1} - w_i \tag{14}$$

$$w_{i+1} = w_i + \propto d_i \tag{15}$$

$$d_i = -\mathbf{H_i^{-1}} \nabla E_i \tag{16}$$

The algorithm is initialized by the initial value of w_0 and $\mathbf{H}_0$. Mostly, $\mathbf{H}_0$ is initially given the value of the identity matrix $\mathbf{H}_0 = I$. The algorithm first computes the position d_0, as shown in Equation (16), from the initial inverse $\mathbf{H_0^{-1}}$ and ∇E_0, then determines new weights, as shown in Equation (15), based on optimal step size $\propto$. The change in weights σ_i, as shown in Equation (14), and change in first order derivative o_i, as shown in Equation (13), are used to approximate $\mathbf{H}$ and its inverse $\mathbf{H^{-1}}$, as shown in Equation (11) and (12), respectively. This algorithm continues until w_i converges. The quasi NM is more efficient than the NM but still, it requires computational memory which limits its applicability to medium-sized problems. To overcome the memory problem and contribute to improving the convergence rate (Setiono and Hui, 1995), a *conjugate gradient method (CG)* was recommended. For conjugate, the gradient of the two vectors needs to be orthogonal to reach a minimum of the cost function. If not orthogonal, it means the second vector need to travel along the previous vector to reach more nearer to a minimum point. Mostly, in GD, it takes slightly larger steps and deviates from the minimal point. The objective of conjugate gradient descent is to take a step along the gradient so that the next gradient vector should be orthogonal and nearer to zero error. The first orthogonal vector d_0 can be computed from the initial guess such that:

$$d_0 = \nabla E_0 \tag{17}$$

The next orthogonal vector d_{i+1} can be expressed as:

$$d_{i+1} = \nabla E_{i+1} + \beta d_i \tag{18}$$

Where β is used to calculate new orthogonal vector direction and is known as a conjugate hyperparameter. The weights are updated as per below rule:

$$w_{i+1} = w_i + \propto d_i \tag{19}$$

The conjugate gradient descent iteratively finds the best orthogonal gradient vector based on the previous vector with an inner product equal to zero and then update the weight parameters. The CG advantage over GD in that it converges faster but compared to other methods such as quasi NM and Levenberg Marquardt (LM), its convergence rate is less. In terms of memory, due to its second order, need more memory as compared to GD and less memory compared to quasi NM and LM (Hagan and Menhaj, 1994). To overcome the problem of both GD slow convergence and second-order gradient algorithms memory, *Quickprop (QP)* was proposed. QP is second order iterative learning algorithm based on Newton's method to find a minimum of the loss function. The purpose

of QP development was to speed up the convergence process by taking much larger steps to reach a minimum of loss function rather than GD infinitesimal small steps in weight space (Fahlman, 1988). The algorithm proceeds like GD, but for each weight, it keeps a copy of $\nabla E_{i-1}$and Δw_{i-1}:

$$\Delta w_i = \frac{\nabla E_i}{\nabla E_{i-1} - \nabla E_i} \Delta w_{i-1} \tag{20}$$

It explains that error vs. weight can be approximated by an upward parabola and change in the slope of error curve by each weight is not affected by all other weights, that are changing at the same time. For each weight, it computes gradient change and weight changes to determine the parabola: and rapidly jump to a minimum of this parabola. Although, QP convergence is faster than GD however it has some drawbacks. First, the previous gradient and weight information need to be stored after every iteration, secondly, it can behave haphazardly during convergence due to much larger steps which need to bring algorithm back to a minimum, and thirdly its zero-difference value in the denominator can overflow algorithm and may make it numerical unstable which needs to be solved by adding small constant value.

To make learning faster, *Levenberg-Marquardt Algorithm (LM)* was proposed by combining both Gauss-Newton (GN) and GD to compute the best gradient direction. It is a method to solve nonlinear least-squares problems to minimize the sum of squared error (Hagan and Menhaj, 1994). Instead of computing directly Hessian matrix **H**, it works with gradient vector and Jacobin matrix **J**. The gradient vector ∇E of the loss function can be computed as:

$$\nabla E = 2\mathbf{J}^T e \tag{21}$$

The **H** can be approximated from the equation below:

$$\nabla^2 E = \mathbf{H} = 2\mathbf{J}^T\mathbf{J} \tag{22}$$

The weight parameter improvement process in LM is iterative such as:

$$\Delta w_i = [\mathbf{J}^T\mathbf{J} + \mu I]^{-1}\mathbf{J}^T e \tag{23}$$

where I is an identity matrix and μ is damping hyperparameter factor. The μ is adjusted in each iteration to balance between GN and GD methods. If the objective function achievement is fast, μ is divided by some factor to bring algorithm closer to GN and if objective function achievement is slow in each iteration, μ is multiplied by some factor to move towards GD. In many applications, LM is very fast and converge to the local minimum rapidly, which may not be the global minimum. The LM has the disadvantage that it cannot be used with other loss function such as cross entropy and cannot be applied to constructive types of neural networks (Hunter et al., 2012). The Jacobian matrix becomes large and needs a lot of memory with an increase in network size with limit its application on a large dataset. Therefore the learning speed of LM is less evident compared to GD when network size increase (Wilamowski and Yu, 2010). The LM limits its application to a fixed topology neural network. Therefore, *Neuron by Neuron (NBN)* was proposed to compute the gradient vector and Jacobian matrix for arbitrarily constructive neural networks. Wilamowski et al. (2008) highlighted that several improvements have been proposed in second order learning algorithms, but much better results can be achieved from Newton and LM

methods. The NBN was proposed to simplify the Jacobian calculation like gradient and make it workable for constructive algorithms. Jacobian is expressed in a square matrix of first order partial derivative:

$$\mathbf{J} = \frac{\partial e}{\partial w} \tag{24}$$

NBN calculates Jacobian in gradient vector form instead of the matrix by performing: 1) Forward computation, 2) Backward computation and finally 3) Calculating Jacobian elements. In the forward computation, the inputs are processed to get neuron output, which is further processed to get the target output. During forward computation, the value of the slope of the neuron activation function is stored for the backward stage. In a backward step, the element of Jacobian is computed by multiplying the neuron delta with its slope and input weights. Finally, instead of using the Jacobian matrix to store values, they are summed into a gradient vector:

$$\nabla E = \frac{\partial e}{\partial w} e \tag{25}$$

This enabled NBN to use with the constructive algorithm but with the additional cost of more memory requirement compared to LM. Hunter et al. (2012) argue that NBN is not perfect, but it can compete with other similar algorithms. Their experimental work shows that NBN achieved much better results than GD and almost similar results to LM.

4.1.3 Application of Gradient Learning Algorithms

The gradient learning algorithms have gained much attention from the authors compared to the traditional statistical techniques. Gradient learning algorithms help to make a more informed decision from the available information. Table III highlights some of the applications of gradient learning algorithms that researchers used to demonstrate the effectiveness of algorithms during the comparative study. Before the year 2000, the range of real-world applications appears to be on less side. The gradient learning algorithms are among the early attempts that researchers investigated to build FNN. In the early attempt phase, the possible reason for the unavailability of public real-world application data sources and less research interest might enforce researchers to rely highly on using artificial benchmarking data.

The successful application of gradient learning algorithms is dependent on user expertise to decide and adjust hyperparameters correctly. The major concern of researchers and users in gradient learning algorithms is to find a method to converge network faster. It is believed that SGD helps to achieve generalization performance many times faster than batch learning. Wilson and Martinez (2003) work demonstrated that SGD was able to achieve required accuracy an average 20 times faster compared to batch learning during classifying real-world problems such as credit card requests, patient diabetes, flower species, beverages types, country religions, crime, voters, and various health diseases. Similar, while dealing with problems having more than 1000 instances such as satellite images, shuttle controls and displaying seven-segment digits, the SGD achieved required accuracy an average 70 times faster than batch learning. Increasing the data size further reduces the speed of batch learning and may take more than 300 times as long compared to SGD for problems having instance greater than 10,000.

The issue with first-order gradient learning is slow learning ability and their usage in a fixed topology neural network can make the task more time-consuming. Deciding too many hidden units in network may decrease the learning speed and cause network to become slow and unstable. Using second-order gradient learning algorithms overcome the limitation of first-order but are constrained with memory requirement. Setiono and Hui (1995) demonstrate that by using second-order learning algorithm such as quasi NM along with constructive neural network limits the growth of hidden units and are helpful in achieving requirement accuracy in less time. Their work on breast cancer problem was able to increase prediction accuracy rate to 2.92%-3.15%.

Similar to GD popularity, the another most widely used learning algorithm in many application areas and embedded in many simulation packages for training network is LM. The learning of LM is considered to be an average 16 – 136 times faster than another second order CG (Hagan and Menhaj, 1994), but limits their applicability to least square loss function and fixed topology neural networks. Hunter et al. (2012) explain that second order NBN can be applied to constructive neural network as an alternative which performance is identical to LM.

4.2Gradient free algorithms

Application	Description
Hepatitis	Predicting whether the patient will survive or die suffering from hepatitis (Wilson and Martinez, 2003)
Animals	Classifying animals into seven classes based on their physical characteristics (Wilson and Martinez, 2003)
Flowers species	Classifying the flowers into different species from available information on the width and length of petals and sepals (Wilson and Martinez, 2003)
Beverages	Identifying the type of beverages in term of its physical and chemical characteristics (Wilson and Martinez, 2003)
Country religion	Predicting the religion of the countries from the information such as population size and their flag colours (Wilson and Martinez, 2003)
Object detection	Predicting whether object is rock or mine from the signal information obtained from various sensors (Wilson and Martinez, 2003)
Crime	Identifying of glass type used in crime scene based on chemical oxide content such as sodium, potassium, calcium, iron and many others (Wilson and Martinez, 2003)
Voters	Classifying voters based on their education, crime, immigration, tax payers and many others (Wilson and Martinez, 2003)
Heart diseases	Diagnosing and categorizing the presence of heart diseases in a patient by studying the previous history of drug addiction, health issues, blood tests, and many others (Wilson and Martinez, 2003)
Liver disorder	Diagnosing alcohol-related liver disorder based on the reports of various blood tests (Wilson and Martinez, 2003)
Earth atmosphere	Determining the strength of ions and free electrons on the layer of earth atmosphere (Wilson and Martinez, 2003)
Outdoor objects segmentation	Segmenting the outdoor images into many different classes such as window, path, sky and many others (Wilson and Martinez, 2003)
Vowel recognition	Recognizing vowel of different or same languages in the speech mode (Wilson and Martinez, 2003)
Breast cancer	Diagnosing breast cancer as a malignant or benign based on the feature extracted from the cell nucleus (Setiono and Hui, 1995; Wilson and Martinez, 2003)
Credit card	Deciding to approve or reject credit card request based on the available information such as credit score, income level, gender age, sex, and many others (Wilson and Martinez, 2003)
Diabetes	Diagnosing whether the patient has diabetes based on certain diagnostic measurements (Wilson and Martinez, 2003)
Silhouette vehicle images classification	Classifying image into different types of vehicle based on the feature extracted from the silhouette (Wilson and Martinez, 2003)
Seven segment display	Predicting number one to nine in seven segmental display (Wilson and Martinez, 2003)
Mushroom	Differentiating poisonous and non-poisonous mushroom based on the mushroom different physical characteristics (Wilson and Martinez, 2003)
Shuttle	Deciding the type of control suitable for the shuttle during an auto landing rather than manual control (Wilson and Martinez, 2003)
English letters	Identifying black and white image as one of the English letters among twenty-six capital letters (Wilson and Martinez, 2003)

Table III. Applications of Gradient learning algorithms

The gradient learning algorithms randomly assign connection weights to the network and iteratively tune them to get an optimal weight for a network with better error minimization capability. The disadvantages associated with gradient learning algorithms are that it requires user expertise to build an optimal FNN. The better choice of hyperparameters such as learning rate, momentum, regularization, and initial weight may increase convergence but need a lot of trial and error approaches to get the best possible parameters. This may cause gradient learning algorithms to trap the FNN at a local minimum which may be far away from the global minimum and may affect the generalization performance. In terms of convergence rate, it can be increased by assigning a large learning rate, however, the algorithm will become unstable, whereas for a low learning rate it may converge slowly and may take even hours, days or months to solve large dataset with complex patterns. Another major issue associated with gradient learning algorithms is that it cannot take nondifferentiable activation function such as threshold in the hidden units due to its zero derivative. For this case, an alternative strategy is to use differentiable activation functions by multiplying a variable with some constant value in the exponential of the activation function. This causes an increase in learning time because of additional computations.

Researchers made an extensive effort to improve the gradient learning algorithms. Huang et al. (2015) comment that improvement in gradient learning lead to faster learning speed and better generalization performance, but still most of them cannot guarantee a global solution. Researchers proposed that the issues associated with gradient learning algorithms can be fixed in a state-of-the-art way by learning FNN without gradient learning algorithms known as gradient free learning in the forward steps. The gradient free algorithms eliminate the need for chain delta rule to calculate the derivative of the loss function with respect to each weight and updating them for the next iteration. Gradient free learning algorithms proposed to analytically calculate connection weights for FNN. Some of the advantages of gradient free algorithms are that: 1) They are simple, fast, and do not need complex hyperparameter adjustments, 2) no need of backpropagating error, and 3) can work directly with both differentiable and non-differentiable activation function (Ferrari and Stengel, 2005). The gradient free algorithms are useful in that they reduce training time significantly, however, it has a drawback in that it increases network complexity. The number of hidden unit's generation increase many times which may cause overfitting. Following are the popular algorithms which eliminate the use of gradient information.

4.2.1 Probability and General Regression

Probabilistic Neural Network (PNN) was proposed for classification problems (Specht, 1990), whereas, *General Regression Neural Network* for regression problems (Specht, 1991). Both algorithms were proposed to address the slowness of the backpropagation feedforward neural network (BPFNN) by doing one pass learning. They are similar in structure and do not need iterative tuning and work in a highly parallel structure. PNN finds the decision boundaries between pattern, whereas, GRNN estimate continuous dependent variable. The network architecture consists of four layers: input, pattern, summation, and output. The input contains the input features, pattern layer calculates the activation function from a Euclidian distance, summation layer takes summation of training output data and activation function in nominator part and summation of activation functions in denominator part, and output layer divides the nominator part by the denominator part of summation layer. The output layer can be expressed as:

$$p_h = \frac{\sum a_h \, e^{-(\frac{d_h^2}{2\sigma^2})}}{\sum e^{-(\frac{d_h^2}{2\sigma^2})}} \tag{26}$$

where d_h^2 is Euclidean distance and $e^{-(\frac{d_h^2}{2\sigma^2})}$ is the activation function. The best value of kernel σ is estimated by holdout or validation method. PNN and GRNN are considered faster than the BPFNN and learn in one pass which means in the forward direction. The major drawback is that it requires more memory space compared to BPFNN and separate algorithms are built for classification and regression problems.

4.2.2 Extreme Learning Machine

Huang, Zhu, et al. (2006) mentioned that the learning speed of traditional FNN is far slower which limits its applicability in many application areas. Two possible reasons are: 1) Slow learning ability of gradient-based BP

algorithms, and 2) Iteratively tuning of all parameters in the network by gradient learning algorithms. They proposed a simple algorithm known as *Extreme learning machine (ELM)* for single layer FNN (SLFN). It randomly chooses hidden units U, and analytically determine only output connection weights w^{ocw}. The U are considered in a linear relationship to the output unit a and w^{ocw} are calculated from the expression:

$$Uw^{ocw} = a_h \tag{27}$$

$$w^{ocw} = U^{\dagger}a_h \tag{28}$$

where $U^{\dagger}= (U^TU)^{-1}U^T$ is Moore-Penrose generalized inverse of U. This algorithm has gained much popularity due to its learning simplicity. It steps forward, and no BP gradient information is needed to compute weights. Their experimental results based on artificial and real-world regression and classification problems demonstrated that ELM can achieve better generalization performance in most cases and learn many times faster than traditional learning algorithms of FNN. In addition, more stable results and better generalization can be achieved by adding positive value $(1/\lambda)$ in the diagonal of (UU^T) or (U^TU) such that (Huang et al., 2012):

when $h < r$:

$$w^{ocw} = U^T(\frac{1}{\lambda} + UU^T)^{-1}a_h \tag{29}$$

when $h > r$:

$$w^{ocw} = (\frac{1}{\lambda} + U^TU)^{-1}U^Ta_h \tag{30}$$

where r is the number of hidden units. ELM limits its applicability to batch learning, whereas, one by one or chunk by chunk data (mini-batches) of fixed or varying size can be learned through *Online Sequential ELM (OS-ELM) algorithm* (Liang et al., 2006). OS-ELM working methodology idea is similar to ELM. The hidden units' parameters are randomly generated, and output weights are analytically calculated. Unlike other GD sequential algorithms with many hyperparameters, OS-ELM only specifies the number of hidden units. The OS-ELM has several advantages that it can learn chunk data of fixed or varying size, it does not depend on past data and only new arrived chunk data is learned, the chunk that has been learned is discarded from chunk size and is suitable even if there is no prior information that how large training example will be a chunk.

Like BP, the limitation of ELM is that the optimal number of hidden units are selected based on trial and error approach. The ELM is learned with some initial guess of hidden units and then experimental trials are performed with different hidden units to select the best optimal network having a hidden unit's capable of maximum error reduction. Although the learning speed of ELM is much faster than traditional BP, however the initial setup to find optimal hidden units may increase the total trial and error time (Han et al., 2017). *Incremental ELM (I-ELM)*, an extension of ELM, was proposed to solve the problem of hidden units allocation (Huang, Chen, et al., 2006). The key difference is that I-ELM is a constructive topology type, whereas ELM is a fixed topology type FNN. I-ELM initialize with one hidden unit and add one by one hidden unit until error converges or maximum hidden units are achieved. The output weight for the new hidden unit can be computed from the expression:

$$w_r^{ocw} = \frac{EU_r^T}{U_r U_r^T} \tag{31}$$

Initially, the error E is set to a_h such that E=a_h, and after adding a new hidden unit, the error is recalculated as:

$$E = E - w_r^{ocw} U_r \tag{32}$$

Adding hidden unit one by one results in redundant hidden units in I-ELM which will make network size large and complex. Some of the hidden unit's contribution to error reduction might be very low and can be omitted. Efforts were made to compact I-ELM network size without losing generalization accuracy. *Convex I-ELM (CI-ELM)* was proposed to recalculate the w^{ocw} based on Barron's convex optimization technique to improve the convergence rate of I-ELM (Huang and Chen, 2007). In CI-ELM, w_r^{ocw} for the randomly generated hidden unit is calculated as:

$$w_r^{ocw} = \frac{E.[E-(F-U_r]^T}{[E-(F-U_r].[E-(F-U_r]^T} \tag{33}$$

Where $F = a$ is the target vector. It recalculates the w^{ocw} of all existing hidden units if $r > 1$, and error as expressed below:

$$w_i^{ocw} = (1 - w_r^{ocw}) w_i^{ocw} \tag{34}$$

$$E = (1 - w_r^{ocw}) E - w_r^{ocw} (F - U_r) \tag{35}$$

The experimental results with the constraint of 200 hidden units demonstrated that CI-ELM achieved better generalization performance and approximately similar learning time compared to I-ELM. CI-ELM can converge faster with more compact architecture while maintaining I-ELM simplicity and efficiency. Similarly, *Enhanced I-ELM (EI-ELM)* was proposed to compact I-ELM by adding some set of candidate units and selecting a candidate unit as a hidden unit having a maximum capability of error reduction (Huang and Chen, 2008). The hidden unit addition in I-ELM might take it nearer or away from the loss function. The hidden units away from loss function may not contribute to error reduction and can be omitted. The EI-ELM add some number of candidate units and one nearer to the loss function is selected as a new hidden unit and added to the network. In such a case, a number of hidden units in EI-ELM will be less and the network size will be more compact compared to I-ELM with the same amount of training time.

Feng et al. (2009) address two main issues of ELM: 1) How to choose optimal hidden units in ELM, and 2) whether ELM computation complexity can be further reduced given large training examples requiring many hidden units. The issues were addressed by proposing *Error Minimized ELM (EM-ELM)* to automatically determine the number of hidden units rather than the trial and error approach. It works by adding hidden units one by one or group by group (with varying group size) and update output weights in a fast-recursive way. The advantage of EM-ELM is that it reduces the computational complexity by only updating the output weights incrementally each time rather than ELM which needs to recalculate the entire output weights when architecture is changed. The experimental work demonstrated that EM-ELM achieved similar generalization performance but

faster than ELM. The hidden units generated by EM-ELM were similar to ELM which implies that EM-ELM can directly calculate hidden units rather than the trial and error approach. Yang et al. (2012) added that the learning speed of ELM and I-ELM are faster, however, there are two major unsolved problems: 1) For ELM, the selection of an optimal number of hidden units is still unknown and trial and error approach are adopted, and 2) I-ELM has solved the problem of ELM by adding hidden units one by one. However, the learning speed of I-ELM increases many times compared to ELM. They proposed an incremental learning algorithm known as *Bidirectional ELM (B-ELM)* to compact the I-ELM architecture without affecting learning effectiveness. In B-ELM, some of the hidden units are not randomly generated, and it tries to find the best hidden unit parameters (w^{icw}, b^{u}) to reduce E as quickly as possible. In B-ELM, when hidden unit $r\epsilon\{2n+1, n\epsilon \boldsymbol{Z}\}$, the hidden units parameters are randomly generated similar to I-ELM, whereas, when hidden unit $r\epsilon\{2n, n\epsilon \boldsymbol{Z}\}$, the hidden units parameters are calculated instead of randomly generated to converge faster. The experimental results on several benchmarking and real-world examples demonstrate that B-ELM is ten to a hundred times faster than existing I-ELM, EI-ELM and EM-ELM with more compact architecture. This may make it more favorable in a real application by reacting to new observation faster after training and deployment.

Ying (2016) highlighted that I-ELM merits are obvious but have four drawbacks which need to be improved: 1) Generate redundant units, 2) number of hidden units are sometimes larger than training examples, 3) the solution is not least squares indicating that it is not optimal, and 4) rarely used to solve multiclass classification problems. The proposed CI-ELM and EI-ELM may learn faster and build more compact architecture; however, the drawbacks are not settled. They proposed *Orthogonal I-ELM (OI-ELM)* by incorporating a Gram-Schmidt orthogonalization method in I-ELM to obtain the least squares solution. It randomly generates one hidden unit similar to I-ELM and calculates its output U_r. The Gram-Schmidt orthogonalization method is applied to hidden unit output to determine the orthogonal vector V_r and if its norm is greater than the predefined value, it is added, else eliminated. For w_r^{ocw} calculation, the basic idea is similar to I-ELM with the replacement of V_r with U_r vector in Equation (31). Their experimental work demonstrates that OI-ELM achieved more a compact network and faster convergence compared to ELM, I-ELM, CI-ELM, and EI-ELM. Inspired from the idea of (Ying, 2016), Zou et al. (2018) proposed a new algorithm called *OI-ELM based on driving amount (DAOI-ELM)* to obtain better generalization performance with more compact architecture. DAOI-ELM determine V_r similar to OI-ELM with modification in w_r^{ocw}. It adds E_{r-1} to V_r while calculating w_r^{ocw}. There comparison of DAOI-ELM with I-ELM, OI-ELM and B-ELM on several benchmarking and real-world dataset demonstrated the effectiveness of DAOI-ELM.

Similarly, for ELM, Wang et al. (2016) explained that ELM is sensitive to the selection of an optimal number of hidden units in the layer and improper hidden units can lead to suboptimal accuracy. They proposed *Self-adaptive ELM (SaELM)* to find the best possible number of hidden units for the network. SaELM initializes by defining the minimum and maximum possible hidden units with its interval, width factor Q and scale factor L. The advantage of a self-adaptive mechanism is that it helps SaELM to search for the best possible hidden units with minimum error capability and the same was demonstrated in their experimental work. Han et al. (2017) argue that

much efforts have been dedicated to convergence accuracy of I-ELM, whereas, its numerical stability (condition) is generally ignored. The numerical stability is directly related to the input weight and hidden biases. The issue was addressed by combining particle swarm optimization (PSO) and EM-ELM called *IPSO-EM-ELM*. The algorithm proposed to add one by one hidden unit to the existing network. PSO is recommended to optimize the input weight and hidden bias in the new hidden unit. The optimal hidden unit is selected based on not only the minimum error of training data but also considering the condition value of the hidden unit output matrix. The output weight needs to be incrementally updated similar to EM-ELM. The experimental work on regression problems demonstrates the effectiveness of IPSO-EM-ELM in term of generalization performance and compact architecture compared to I-ELM, EI-ELM, EM-ELM and dynamic ELM (D-ELM), however, IPSO-EM-ELM requires more training time because of use of PSO to select optimal hidden units.

Zong et al. (2013) highlighted that ELM provides a better performance, however, none of the work in ELM mentioned the problem of unbalanced data distribution. Typically, imbalance class distributions are balanced by adopting either sampling techniques (oversampling or undersampling) or algorithmic approaches. They proposed an algorithm named as *Weighted ELM (W-ELM)* to handle both binary and multi-class imbalance data problems. Unlike, ELM which considers all training examples equal, W-ELM add a penalty term to errors corresponding to different inputs. Similar to Equations (29) and (30), it derived two versions of w^{ocw}:

when $h < r$:

$$w^{ocw} = U^T(\frac{1}{\lambda} + WUU^T)^{-1}Wa_h \tag{36}$$

when $h > r$:

$$w^{ocw} = (\frac{1}{\lambda} + U^TWU)^{-1}U^TWa_h \tag{37}$$

where W is a diagonal weight matrix defined for every training example. It determines what degree of re-balance user is concerned and how much boundary can further be pushed towards the majority class. When training example comes from minority class, it is assigned a relatively higher value of W than others. Experimental work demonstrates that W-ELM not only obtains better generalization performance compared to ELM on the imbalanced dataset by allocating importance to minority class compared to majority class but also maintained good performance on the well-balanced dataset.

ELM and its variant are mainly focused on classification and regression problems and still encounter difficulties in natural scenes (e.g., signals and visual) and practical applications (voice recognition and image classification) due to its shallow architecture which is unable to learn features even with a large number of hidden units. In many cases, a multilayer solution is required for feature learning before classification is performed (Tang et al., 2016). Kasun et al. (2013) proposed *Multilayer ELM (ML-ELM)* for classification based on extreme learning machine autoencoder (ELM-AE). ELM was modified to ELM-AE by keeping the output same as input for autoencoder and estimate weight for the hidden layer. The number of successive hidden layers were calculated in the same

manner as ELM methodology to create layer weights for ML-ELM. Finally, the output layer weight for ML-ELM is calculated using regularized least squares. Tang et al. (2016) highlighted that the encoded output from ELM-AE is directly fed into the last layer for decision making before least squares, without random feature mapping which violates the ELM universal approximation-based theories. Tang et al. (2016) proposed a new *Hierarchical ELM (H-ELM)* consisting of two parts: unsupervised feature encoding based on new l^1 regularized ELM autoencoder to extract multilayer sparse features of input data, and supervised feature classification based on ELM is applied for decision making. H-ELM is based on universal approximation capability theories of ELM and results demonstrates its superior performance over ELM and other FNN autoencoders.

4.2.3 Semi Gradient and Iterative Algorithms

The *No-propagation (No-Prop)* simplifies the learning mechanism of multi-layer BPFNN by randomly generating w^{icw} and hidden connection weights w^{hcw} and only iteratively train w^{ocw} by BP learning algorithm (Widrow et al., 2013). The algorithm cannot be considered as a complete gradient free learning because it uses gradient information in its last layer. However, due to its random generation of w^{icw} and w^{hcw}, and only tuning last layer w^{ocw} , it is named as No-Prop. The No-Prop guarantee to minimize the loss function when the number of training patterns is less than or equal to w^{ocw} connecting the last hidden layer to the output units. This criterion is referred to as the least mean square error capacity (LMS capacity). The No-Prop algorithm explains that when the training pattern is under or at LMS capacity, the output unit will deliver the desired output pattern perfectly and the generalization performance will be like BP with much faster results. However, if the training pattern is overcapacity, the BP works better than No-Prop. In such case, increasing the number of hidden units of the last layer will increase the number of output weights and again the training pattern will become under or at capacity and performance of No-Prop will increase.

Application	Description
Boston house price	Estimating the price of houses based on the availability of clean quality air (Feng et al., 2009; Han et al., 2017; Huang et al., 2012; Huang, Chen, et al., 2006; Huang and Chen, 2007, 2008; Ying, 2016)
California house price	Predicting the house prices based on geographical location and infrastructure of the house (Han et al., 2017; Huang, Chen, et al., 2006; Huang, Zhu, et al., 2006; Huang and Chen, 2007, 2008; Liang et al., 2006; Ying, 2016)
Species	Determining the age of species from their known physical measurements (Feng et al., 2009; Han et al., 2017; Huang et al., 2012; Huang, Chen, et al., 2006; Huang, Zhu, et al., 2006; Huang and Chen, 2007, 2008; Liang et al., 2006; Ying, 2016; Zong et al., 2013)
Aircrafts ailerons	Controlling the ailerons of a fighter aircrafts (Han et al., 2017; Huang, Chen, et al., 2006; Huang, Zhu, et al., 2006; Huang and Chen, 2007, 2008)
Aircrafts elevators	Controlling the elevators of a fighter aircrafts (Han et al., 2017; Huang, Chen, et al., 2006; Huang, Zhu, et al., 2006; Huang and Chen, 2007, 2008)
Computers system activity	Measuring the portion of time that central processing units is running in user mode, system mode, waiting mode and idle mode from the collection of computers systems activity (Huang, Zhu, et al., 2006; Huang and Chen, 2008)
House prices in specific region	Determining the median prices of houses based on prices in the region and demographic information. (Han et al., 2017; Huang, Chen, et al., 2006; Huang, Zhu, et al., 2006; Huang and Chen, 2007, 2008; Ying, 2016)
Adult income	Determining the income of adult based on demographic information (Zong et al., 2013)
Automobile price	Determining the prices of automobile based on various auto specifications, the degree to which auto is risky than price, and an average loss per auto per year (Feng et al., 2009; Huang et al., 2012; Huang, Chen, et al., 2006; Huang, Zhu, et al., 2006; Huang and Chen, 2007, 2008; Ying, 2016)
Cars fuel consumption	Determining the fuel consumption of cars in terms of engine specification and car characteristics (Liang et al., 2006; Yang et al., 2012)
Drug compound	Designing modern drug by predicting whether the compound is active or inactive to the binding target (Huang, Zhu, et al., 2006; Ying, 2016)
Computer machine	Estimating the relative performance of a computer central processing unit considering the memory and channels requirements (Feng et al., 2009; Han et al., 2017; Huang, Chen, et al., 2006; Huang, Zhu, et al., 2006; Huang and Chen, 2007, 2008; Yang et al., 2012; Ying, 2016)
Servomechanism rise time	Estimating servomechanism rise time in term of two choices of mechanical linkage and two gain setting (Huang, Zhu, et al., 2006; Huang and Chen, 2008; Ying, 2016)
Breast cancer	Diagnosing breast cancer as a malignant or benign based on the feature extracted from the cell nucleus (Huang, Zhu, et al., 2006; Ying, 2016; Zong et al., 2013)
Telemarketing	Measuring the accomplishment of telemarketing calls for marketing bank long term deposits (Huang, Zhu, et al., 2006; Huang and Chen, 2008; Yang et al., 2012)
Stock price	Discovering the stock price trend of the company based on information generated by similar competitive companies (Huang, Zhu, et al., 2006; Ying, 2016)
Diabetes	Diagnosing whether the patient has diabetes based on certain diagnostic measurements (Cao et al., 2016; Huang, Zhu, et al., 2006; Huang and Chen, 2008; Tang et al., 2016; Zong et al., 2013)
Soil classification	Classifying image according to a different type of soil such as grey soil, vegetation soil, red soil and many others based on a database consisting of the multi-spectral images (Feng et al., 2009; Huang et al., 2012; Huang, Zhu, et al., 2006; Liang et al., 2006; Tang et al., 2016; Ying, 2016; Zong et al., 2013)
Outdoor objects segmentation	Segmenting the outdoor images into many different classes such as window, path, sky and many others (Feng et al., 2009; Huang et al., 2012; Huang, Zhu, et al., 2006; Liang et al., 2006; Ying, 2016)
Shuttle	Deciding the type of control suitable for the shuttle during an auto landing rather than manual control (Huang et al., 2012; Huang, Zhu, et al., 2006; Zong et al., 2013)
Clustering	Clustering the dataset into different classes based on available target vector (Huang, Zhu, et al., 2006; Zong et al., 2013)
Credit card	Deciding to approve or reject credit card request based on the available information such as credit score, income level, gender age, sex, and many others (Tang et al., 2016)
Liver disorder	Diagnosing alcohol-related liver disorder based on the reports of various blood tests (Tang et al., 2016)
Cancer	Classification of the leukaemia cancer as acute lymphoblast leukaemia or acute myeloid leukaemia (Tang et al., 2016; Zong et al., 2013)
Gene expression level	Analysing the gene correlation expression level in different tissues of the tumor colon and normal colon (Tang et al., 2016; Zong et al., 2013)
Object discrimination	Discriminating stars from galaxy using broadband photometric information (Huang et al., 2012)
Mushroom	Differentiating poisonous and non-poisonous mushroom based on mushroom different physical characteristics (Tang et al., 2016)
Flowers species	Classifying the flowers into different species from available information on the width and length of petals and sepals (Huang et al., 2012; Tang et al., 2016; Wang et al., 2016; Zong et al., 2013)

Cao et al. (2016) argue that the random generation of hidden units' parameters, and analytically calculation of output weights become infeasible and generalization performance drops when the dataset is extremely large. *Iterative Feedforward Neural Networks with Random Weights (IFNNRWs)* was proposed to overcome the issues

Crime	Identifying glass type used in crime scene based on chemical oxide content such as sodium, potassium, calcium, iron and many others (Cao et al., 2016; Huang et al., 2012; Tang et al., 2016; Ying, 2016; Zong et al., 2013)
DNA splicing	Recognizing exon/intron and intron/exon boundaries in the DNA splicing (Huang et al., 2012; Liang et al., 2006; Tang et al., 2016; Zong et al., 2013)
Industrial strike volume	Estimating the industrial strike volume for the next fiscal year considering key factors such as unemployment, inflation and labor unions (Huang et al., 2012)
Weather forecasting	Forecasting weather in terms of cloud appearance (Huang et al., 2012)
Dihydrofolate reductase inhibition	Predicting the inhibition of dihydrofolate reductase by pyrimidines (Huang et al., 2012; Huang and Chen, 2008)
Human body fats	Determining the percentage of human body fats from key physical factors such as weight, age, chest size and other body parts circumference (Huang et al., 2012)
Heart diseases	Diagnosing and categorizing the presence of heart diseases in a patient by studying the previous history of drug addiction, health issues, blood tests, and many others (Huang et al., 2012)
Mental disorder	Testing mental behaviour of the patient from inflated balloons (Huang et al., 2012)
Earthquake strength	Forecasting the strength of earthquake given its latitude, longitude and focal point (Huang et al., 2012)
Presidential election	Estimating the proportion of voter in the presidential election based on key factors such as education, age, and income (Huang et al., 2012)
Robot end effector	Determining the distance of robot end effector from a target based on the robot positions and angles (Huang and Chen, 2008)
Concrete strength	Determining slump, flow and compressive strength of the concrete from influencing ingredients such as cement, water, ash, and many others (Yang et al., 2012; Zou et al., 2018)
Beverages quality	Determining quality of same class of the beverages based on relevant ingredients (Tang et al., 2016; Wang et al., 2016; Yang et al., 2012; Zou et al., 2018)
Industrial fault diagnosis	Diagnosing fault of the industrial systems such as Tennessee-Eastman Process (Zou et al., 2018)
Heating, ventilation and air-conditioning	Determining the heating load and cooling load of the residential building by considering the design layout of the walls, rooms, and surface (Zou et al., 2018)
Forest burned area	Predicting the burned area of the forest considering various environmental and weather conditions (Zou et al., 2018)
Stock exchange market	Studying relationship of the 100-index stock exchange market with other international stock market indices (Zou et al., 2018)
Protein localization	Predicting protein localization by studying the cell membranes characteristics (Zong et al., 2013)
Patient disease	Diagnosing whether the patient is suffering from hypothyroidism or hyperthyroidism (Zong et al., 2013)
Page block segmentation	Segmenting the type of page block as text, horizontal line, graphics, vertical line or picture (Zong et al., 2013)
Breast cancer	Studying the effect of breast cancer by predicting that the patient will survive less or more than five years (Zong et al., 2013)
Handwritten images classification	Classifying images of the handwritten digits (Kasun et al., 2013; Tang et al., 2016)
Object identification	Detecting whether the object is a car or not from its side view (Tang et al., 2016)
Hand gestures	Extracting useful information from the hand gesture (Tang et al., 2016)
Appearance changes	Modelling and tracking of appearance changes such as pose variation, shape deformation, illumination change, camera motion, and many others (Tang et al., 2016)
Energy particles	Classifying energy particles either as gamma or hadron (Cao et al., 2016)
Human faces gestures	Recognizing human faces gestures such as head pose, facial expression, eyes state, and many others (Cao et al., 2016)
Beverages	Identifying the type of beverages in term of its physical and chemical characteristics (Huang et al., 2012)
Vowel recognition	Recognizing vowel of different or same languages in the speech mode (Huang et al., 2012; Tang et al., 2016; Zong et al., 2013)
Silhouette vehicle images classification	Classifying image into different types of vehicle based on the feature extracted from the silhouette (Huang et al., 2012; Ying, 2016; Zong et al., 2013)
English letters	Identifying black and white image as one of the English letters among twenty-six capital letters (Feng et al., 2009; Huang et al., 2012; Tang et al., 2016; Ying, 2016)
Handwritten text recognition	Recognizing isolated, touching, overlapping and cursive handwritten text from digital images of the city, states, Zip codes, and alphanumeric characters (Huang et al., 2012; Tang et al., 2016; Zong et al., 2013)
Basketball winning	Predicting basketball winning team based on players, team formation and actions information (Huang et al., 2012)

Table IV. Applications of Gradient free learning algorithms

generated from random generation. The iterative algorithm IFNNRWs is developed which iteratively tune output connection weight based on l^2 model. It randomly generates input weight and hidden units but calculates iteratively w^{ocw} as expressed below:

$$w_{i+1}^{ocw} = \frac{1}{1+\lambda}((I - U^T U)w_i^{ocw} + U^T U) \quad (38)$$

The advantages of this algorithm are that l^2 regularization improve its generalization performance. Unlike algorithms which are based on Moore-Penrose generalized Inverse of hidden unit's which consumes a lot of memory with a number of examples increases, IFNNRWs is more stable and unaffected by increasing the number of hidden units.

4.2.4 Application of Gradient free learning Algorithms

Many authors have successfully demonstrated the effectiveness of the gradient free learning algorithms in a wide range of applications. The category consists of a variety of applications in the management, engineering, and health sciences domain. The notable applications include area, but is not limited to, such as supply chain and logistics, financial analysis, marketing and sales, management information systems, decision support systems, product and process improvements, manufacturing cost reduction, business improvements, and health services. Table IV illustrates the range of applications of gradient free learning algorithms. In literature, gradient free learning algorithms are mostly compared with gradient learning algorithms by considering the application areas mentioned in Table IV. The gradient free learning algorithms are relatively new compared to the gradient learning algorithms and continuously gaining the attention of researchers.

The gradient learning algorithms disadvantages in that it faces a problem of local minimum which can reduce the generalization performance, and iterative tuning of connection weights may cause the learning to be more time-consuming. Gradient free learning algorithms are considered to have faster convergence with more stable results on many application problems. For instance, the application of gradient free learning algorithm (ELM) on various problems such as predicting stock price, house price, automobile price, species age, cancer, diabetes drug compound, aircraft ailerons and elevators, and adult income found that generalization performance improves by an average 0.12 times with learning speed an average 20 times faster than gradient learning algorithms. Moreover, on large complex problems such as predicting soil types, segmenting objects, and shuttle control, the ELM improved accuracy an average of 6% and achieved prediction thousand times faster than the gradient learning algorithms.

Later, several variants were proposed to improve ELM and most of them are discussed in Section 4.2.2. The application of variants such as B-ELM on the problem of measuring telemarking calls, computer performance, car fuel, concrete strength, and beverages quality showed an improvement in learning speed of an average 34, 4 145 times faster than I-ELM, EM-ELM, and EI-ELM. Similarly, the application of another ELM variant named as DAOI-ELM in studying the stock market, forest burning, concrete strength, beverage quality achieved more stable and smooth results compared to the B-ELM and the fluctuating results of I-ELM and OI-ELM. More work on fault diagnosis of Tennessee-Eastman Process (TEP) demonstrates that DAOI-ELM improved the accuracy to 1.38%-4.54% for classes-2, 1.12%-6.39% for classes-4 and 4.36%-5.47% for classes-8 fault compared to BP, I-ELM, CI-ELM, and OI-ELM. The detail study gives clear direction that DAOI-ELM obtained better

generalization performance with compact architecture; however, the learning speed of DAOI-ELM is not clearly illustrated in the study. The ELM and many of its variants advantageous in that they randomly generate hidden units and analytically calculate output connection weights which make them simple and easy to train network. However, randomly generating hidden units may cause the network size to increase large enough which increases the chances of overfitting.

The application of semi gradient and iterative tuning algorithms such as IFNNRWs demonstrated its effectiveness on classification problems of identifying crime object, energy particles, human face gestures, and diabetes, but cannot approximate regression problems well. Another limitation is an increase in training time of IFNNRWs due to repetitively tuning of output connection weights.

5. Discussion of future research directions

The literature survey was conducted to get an in-depth insight of FNN and researcher contributions in improving its generalization performance and learning speed. The existing learning algorithms have a major contribution and further improvements in future research can create a significant contribution. In Section 4, it can be understood that existing improvement is not straightforward. The researchers are in continuous efforts to propose algorithms that are computationally efficient and has better generalization performance. By analyzing the existing research, this section provides a future research direction in that it will be beneficial to improve the FNN convergence.

5.1Activation function

In studies, few attempts have been made to study the effect of using various types of activation functions on FNN. Karlik and Olgac (2011) study the comparison of popular activation functions including uni-polar sigmoid, bi-polar sigmoid, tangent hyperbolic, radial bias function and conic section function along with gradient-based algorithms for fixed topology FNN. The limitation of this experimental work was using 10 hidden units and 40 hidden units with 100 iterations and 500 iterations respectively. Moreover, the data structure, normalization techniques, learning algorithms, and hyperparameters were not clearly stated in experimental work. The experimental results demonstrate that tangent hyperbolic activation function performance is better than the others. However, it is observed in various studies that sigmoid activation function application is more compared to tangent hyperbolic. The importance of hidden units and their activation functions cannot disagree. They are used in every type of FNN and the study of various activation functions performance on fixed and constructive topology along with gradient algorithm and gradient free algorithm still need a researcher's attention.

5.2Efficient and compact Algorithm with fewer hyperparameters

The gradient learning algorithms are favorable because of its compact size, whereas, gradient-free learning algorithms are favorable because of its delta free learning and fast convergence. The gradient free algorithms network size becomes much larger which increases its complexity and the chance of overfitting increases. The best FNN may be considered one having characteristics of compact architecture with a small number of hidden units and connection weight. It may analytically calculate hidden units and connection weights, need fewer

hyperparameters and reaches global minimum with lesser training time. There is always a trade-off between network generalization performance and learning speed. Some algorithms have the advantage of more efficient than the others but maybe constrained by memory requirement, complex architecture, and/or more learning time. Therefore, more efforts are needed to construct efficient and fast algorithms with fewer hyperparameter, computational simple and compact size.

5.3Connection weight initialization

The purpose of FNN is to find the best optimal connection weights that can generate optimal results. The question arises: What should be the best possible initial weights for the network? Traditional FNN is dependent on initial weight value because it calculates the derivative of the total error with respect to weight to get a minimum of an objective function (Hecht-Nielsen, 1989). Assigning suboptimal weights will cause the network to take more iterations and subsequently decreases its performance. The issue is resolved to some extent by a new approach that analytically calculates connection weight on the output side by randomly generating hidden units. However, the literature can be further strengthened by calculating all connection weights (including input connection and output connection) analytically to generate hidden units explaining maximum variance in the dataset. This may also help to further improve the generalization and learning speed by compacting the size of the network.

6. Conclusions

We conduct a review on Feedforward Neural Network (FNN) learning algorithms and optimization technique designed to achieve better generalization performance and fast learning speed. Traditional FNN is slow and it may take hours, days or even a week to generate results. The results are highly influenced by global (learning rate, initial weight, and a number of hidden units in the hidden layer) and local (proposed in the specific algorithm) hyperparameters. The repeatedly tuning of connection weight with hyperparameters creates a complex coadoption in the network which decreases generalization performance and trapped at a local minimum if a global minimum is far away. The convergence may be increased by the large learning rate, but it will make it unstable, whereas, small learning rate will slow convergence. The optimal FNN with maximum error reduction capability is always not evident. A lot of experimental trials are required to be performed with different combinations of hyperparameters to select the network with minimum error. The selected FNN may not perform well even on unseen data and generalization performance may decrease.

To study above FNN drawbacks and researchers' contributions, a comprehensive review was carried out by using four keywords: "generalization performance", "learning rate", "overfitting" and "fixed and cascade architecture". The combination of keywords was also searched to get more relevant results. After rejecting unrelated articles, a total of 80 was left in the scope of our work. To address the contribution in a more novel and significant way, the articles were classified into six categories (i.e., Gradient learning algorithms for Network Training, Gradient free learning algorithms, Optimization algorithms for learning rate, Bias and Variance (Underfitting and Overfitting) minimization algorithms, Constructive topology Neural Networks, and Metaheuristic search algorithms). Reviewing all six categories merits, limitation, and real-world applications in one paper is too lengthy. Hence, the

six categories were further divided into two main parts. Part I, the current paper reviewed 38 articles for the first two categories on learning algorithms (they are: Gradient learning algorithms for Network Training and Gradient free learning algorithms), whereas, the remaining articles on optimization techniques are reviewed in Part II. The major conclusions of the two categories reviewed in Part I are:

1. The convergence rate of FNN becomes slow by applying first-order gradient information comparative to a second order gradient. First order gradient requires much iteration which slows its learning and stuck at a local minimum, whereas, second-order gradient computes Hessian and its inverse matrix which may need higher computational memory for large features problems. The best approach is to approximate the Hessian matrix and its inverse from first order gradient information which may guarantee to converge at a local minimum, if not a global minimum. Another issue with the gradient-based algorithm is that it may not work with all types of loss function and network topology. For instance, the Levenberg-Marquardt Algorithm (LM) gradient algorithm is considered faster than gradient descent (GD) with limitation in that it can only be applied with least square loss function and fixed topology FNN. The application of stochastic GD (SGD) on real-world problem reveals that it is 20 times faster than the batch learning. When working with high dimensional data having instance more than 1,000 and 10,000, the speed of SGD is considered to be 70 times and 300 times better than batch learning. The application of quasi Newton Method (quasi NM) with a constructive neural network is able to improve the prediction accuracy rate of network to 2.92%-3.15%. The most widely known LM is considered to be 16-136 times faster than conjugate gradient (CG), however, LM applicability is limited to fixed FNN.
2. Gradient learning algorithms can be avoided, and connection weights can be calculated more analytically by gradient free learning algorithms. The learning speed and generalization performance (in most cases) of gradient free learning algorithms are considered better than gradient learning algorithms. However, the network complexity in gradient free algorithms increases because of an increase in a number of hidden units compared to gradient learning algorithms compact network size which increases chances of overfitting. The category includes a wide range of application in the area, but is not limited to, such as supply chain and logistics, financial analysis, marketing and sales, management information systems, decision support systems, product and process improvements, manufacturing cost reduction, business improvements, and health services. The learning speed of gradient free learning algorithms is a hundred times better than gradient learning algorithms with better generalization performance. On large complex high dimensional data, gradient free learning algorithms were able to improve prediction accuracy an average 6% with learning speed thousand times faster than gradient learning algorithms. This category is gaining significant interest and trend shows that researchers are in continuous efforts to further improve the generalization performance and learning speed of existing gradient free learning algorithms.

The researcher's contribution to improving FNN generalization performance and learning speed in the above categories are noteworthy. The successful application of FNN learning algorithms on real-world management, engineering, and health sciences problems demonstrate the advantages of algorithms in enhancing decision

making for practical operations. Lastly, based on our review and research trend in FNN, we proposed future research directions which can bring a significant contribution in performance and learning improvement, including studying the role of various activation functions, recommend efficient and compact algorithm with fewer hyperparameters, and optimal connection weight determination.

References

Abdel-Hamid, O., Mohamed, A., Jiang, H., Deng, L., Penn, G. and Yu, D. (2014), "Convolutional neural networks for speech recognition", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, Vol. 22 No. 10, pp. 1533–1545.

Babaee, M., Dinh, D.T. and Rigoll, G. (2018), "A deep convolutional neural network for video sequence background subtraction", *Pattern Recognition*, Elsevier Ltd, Vol. 76, pp. 635–649.

Bianchini, M. and Scarselli, F. (2014), "On the complexity of neural network classifiers: A comparison between shallow and deep architectures", *IEEE Transactions on Neural Networks and Learning Systems*, IEEE, Vol. 25 No. 8, pp. 1553–1565.

Bottani, E., Centobelli, P., Gallo, M., Kaviani, M.A., Jain, V. and Murino, T. (2019), "Modelling wholesale distribution operations: an artificial intelligence framework", *Industrial Management & Data Systems*, Emerald Publishing Limited, Vol. 119 No. 4, pp. 698–718.

Cao, F., Wang, D., Zhu, H. and Wang, Y. (2016), "An iterative learning algorithm for feedforward neural networks with random weights", *Information Sciences*, Elsevier Inc., Vol. 328, pp. 546–557.

Cao, W., Wang, X., Ming, Z. and Gao, J. (2018), "A review on neural networks with random weights", *Neurocomputing*, Elsevier, Vol. 275, pp. 278–287.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. and Yuille, A.L. (2018), "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, Vol. 40 No. 4, pp. 834–848.

Chung, S.H., Ma, H.L. and Chan, H.K. (2017), "Cascading delay risk of airline workforce deployments with crew pairing and schedule optimization", *Risk Analysis*, Wiley Online Library, Vol. 37 No. 8, pp. 1443–1458.

Deng, C., Miao, J., Ma, Y., Wei, B. and Feng, Y. (2019), "Reliability analysis of chatter stability for milling process system with uncertainties based on neural network and fourth moment method", *International Journal of Production Research*, Taylor & Francis, pp. 1–19.

Dong, C., Loy, C.C., He, K. and Tang, X. (2016), "Image super-resolution Using deep convolutional networks", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 38 No. 2, pp. 295–307.

Fahlman, S.E. (1988), *An Empirical Study of Learning Speed in Back-Propagation Networks*, School of

Computer Science, Carnegie Mellon University, Pittsburgh PA 15213.

Fahlman, S.E. and Lebiere, C. (1990), "The cascade-correlation learning architecture", *Advances in Neural Information Processing Systems*, pp. 524–532.

Feng, G., Huang, G.-B., Lin, Q. and Gay, R.K.L. (2009), "Error minimized extreme learning machine with growth of hidden nodes and incremental learning", *IEEE Trans. Neural Networks*, Vol. 20 No. 8, pp. 1352–1357.

Ferrari, S. and Stengel, R.F. (2005), "Smooth function approximation using neural networks", *IEEE Transactions on Neural Networks*, Vol. 16 No. 1, pp. 24–38.

Hagan, M.T. and Menhaj, M.B. (1994), "Training feedforward networks with the Marquardt algorithm", *IEEE Transactions on Neural Networks*, Vol. 5 No. 6, pp. 989–993.

Han, F., Zhao, M.-R., Zhang, J.-M. and Ling, Q.-H. (2017), "An improved incremental constructive single-hidden-layer feedforward networks for extreme learning machine based on particle swarm optimization", *Neurocomputing*, Elsevier, Vol. 228, pp. 133–142.

Hayashi, Y., Hsieh, M.-H. and Setiono, R. (2010), "Understanding consumer heterogeneity: A business intelligence application of neural networks", *Knowledge-Based Systems*, Elsevier, Vol. 23 No. 8, pp. 856–863.

Hecht-Nielsen. (1989), "Theory of the backpropagation neural network", *International Joint Conference on Neural Networks*, Vol. 1, IEEE, pp. 593–605 vol.1.

Hinton, G.E., Srivastava, N. and Swersky, K. (2012), "Lecture 6a- overview of mini-batch gradient descent", *COURSERA: Neural Networks for Machine Learning*.

Hornik, K., Stinchcombe, M. and White, H. (1989), "Multilayer feedforward networks are universal approximators", *Neural Networks*, Vol. 2 No. 5, pp. 359–366.

Huang, G.-B. and Chen, L. (2007), "Convex incremental extreme learning machine", *Neurocomputing*, Elsevier, Vol. 70 No. 16–18, pp. 3056–3062.

Huang, G.-B. and Chen, L. (2008), "Enhanced random search based incremental extreme learning machine", *Neurocomputing*, Vol. 71 No. 16–18, pp. 3460–3468.

Huang, G.-B., Chen, L. and Siew, C.K. (2006), "Universal approximation using incremental constructive feedforward networks with random hidden nodes", *IEEE Transactions on Neural Networks*, Vol. 17 No. 4, pp. 879–892.

Huang, G.-B., Zhou, H., Ding, X. and Zhang, R. (2012), "Extreme learning machine for regression and multiclass classification", *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, Vol.

42 No. 2, pp. 513–29.

Huang, G.-B., Zhu, Q.-Y. and Siew, C.-K. (2006), "Extreme learning machine: theory and applications", *Neurocomputing*, Vol. 70 No. 1–3, pp. 489–501.

Huang, G., Huang, G.-B., Song, S. and You, K. (2015), "Trends in extreme learning machines: A review", *Neural Networks*, Elsevier, Vol. 61, pp. 32–48.

Hunter, D., Yu, H., Pukish III, M.S., Kolbusz, J. and Wilamowski, B.M. (2012), "Selection of proper neural network sizes and architectures—A comparative study", *IEEE Transactions on Industrial Informatics*, Vol. 8 No. 2, pp. 228–240.

Ijjina, E.P. and Chalavadi, K.M. (2016), "Human action recognition using genetic algorithms and convolutional neural networks", *Pattern Recognition*, Elsevier, Vol. 59, pp. 199–212.

Karlik, B. and Olgac, V. (2011), "Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks", *International Journal of Artificial Intelligence And Expert Systems (IJAE)*, Vol. 1 No. 4, pp. 111–122.

Kastrati, Z., Imran, A.S. and Yayilgan, S.Y. (2019), "The impact of deep learning on document classification using semantically rich representations", *Information Processing & Management*, Elsevier, Vol. 56 No. 5, pp. 1618–1632.

Kasun, L.L.C., Zhou, H., Huang, G. and Vong, C. (2013), "Representational Learning with Extreme Learning Machine for Big Data", *IEEE Intelligent System*, Vol. 28 No. 6, pp. 31–34.

Kim, Y.-S., Rim, H.-C. and Lee, D.-G. (2019), "Business environmental analysis for textual data using data mining and sentence-level classification", *Industrial Management & Data Systems*, Emerald Publishing Limited, Vol. 119 No. 1, pp. 69–88.

Kumar, A., Rao, V.R. and Soni, H. (1995), "An empirical comparison of neural network and logistic regression models", *Marketing Letters*, Vol. 6 No. 4, pp. 251–263.

Kummong, R. and Supratid, S. (2016), "Thailand tourism forecasting based on a hybrid of discrete wavelet decomposition and NARX neural network", *Industrial Management and Data Systems*, Vol. 116 No. 6, pp. 1242–1258.

Lam, H.Y., Ho, G.T.S., Wu, C.-H. and Choy, K.L. (2014), "Customer relationship mining system for effective strategies formulation", *Industrial Management & Data Systems*, Emerald Group Publishing Limited, Vol. 114 No. 5, pp. 711–733.

LeCun, Y., Bengio, Y. and Hinton, G. (2015), "Deep learning", *Nature*, Nature Publishing Group, Vol. 521 No. 7553, pp. 436–444.

Lewis, A.S. and Overton, M.L. (2013), "Nonsmooth optimization via quasi-Newton methods", *Mathematical Programming*, Vol. 141 No. 1–2, pp. 135–163.

Li, M., Ch'ng, E., Chong, A.Y.L. and See, S. (2018), "Multi-class Twitter sentiment classification with emojis", *Industrial Management & Data Systems*, Emerald Publishing Limited, Vol. 118 No. 9, pp. 1804–1820.

Liang, N.-Y., Huang, G.-B., Saratchandran, P. and Sundararajan, N. (2006), "A fast and accurate online sequential learning algorithm for feedforward networks", *IEEE Transactions on Neural Networks*, Vol. 17 No. 6, pp. 1411–1423.

Mohamed Shakeel, P., Tobely, T.E. El, Al-Feel, H., Manogaran, G. and Baskar, S. (2019), "Neural Network Based Brain Tumor Detection Using Wireless Infrared Imaging Sensor", *IEEE Access*, IEEE, Vol. 7, pp. 5577–5588.

Mori, J., Kajikawa, Y., Kashima, H. and Sakata, I. (2012), "Machine learning approach for finding business partners and building reciprocal relationships", *Expert Systems with Applications*, Elsevier, Vol. 39 No. 12, pp. 10402–10407.

Nasir, M., South-Winter, C., Ragothaman, S. and Dag, A. (2019), "A comparative data analytic approach to construct a risk trade-off for cardiac patients' re-admissions", *Industrial Management & Data Systems*, Emerald Publishing Limited, Vol. 119 No. 1, pp. 189–209.

Nguyen, D. and Widrow, B. (1990), "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights", *1990 IJCNN International Joint Conference on Neural Networks*, IEEE, pp. 21–26 vol.3.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J. (1986), "Learning representations by back-propagating errors", *Nature*, Vol. 323 No. 6088, pp. 533–536.

Setiono, R. and Hui, L.C.K. (1995), "Use of a quasi-Newton method in a feedforward neural network construction algorithm", *IEEE Transactions on Neural Networks*, Vol. 6 No. 1, pp. 273–277.

Shanno, D.F. (1970), "Conditioning of quasi-Newton methods for function minimization", *Mathematics of Computation*, Vol. 24 No. 111, pp. 647–647.

Specht, D.F. (1990), "Probabilistic neural networks", *Neural Networks*, Vol. 3 No. 1, pp. 109–118.

Specht, D.F. (1991), "A general regression neural network", *IEEE Transactions on Neural Networks*, Vol. 2 No. 6, pp. 568–576.

Tang, J., Deng, C. and Huang, G.-B. (2016), "Extreme Learning Machine for multilayer perceptron", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 27 No. 4, pp. 809–821.

Teo, A.-C., Tan, G.W.-H., Ooi, K.-B., Hew, T.-S. and Yew, K.-T. (2015), "The effects of convenience and

speed in m-payment", *Industrial Management & Data Systems*, Vol. 115 No. 2, pp. 311–331.

Tkáč, M. and Verner, R. (2016), "Artificial neural networks in business: Two decades of research", *Applied Soft Computing*, Vol. 38, pp. 788–804.

Tu, J. V. (1996), "Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes", *Journal of Clinical Epidemiology*, Vol. 49 No. 11, pp. 1225–1231.

Wang, G.-G., Lu, M., Dong, Y.-Q. and Zhao, X.-J. (2016), "Self-adaptive extreme learning machine", *Neural Computing and Applications*, Springer, Vol. 27 No. 2, pp. 291–303.

Wang, J., Wu, X. and Zhang, C. (2005), "Support vector machines based on K-means clustering for real-time business intelligence systems", *International Journal of Business Intelligence and Data Mining*, Citeseer, Vol. 1 No. 1, pp. 54–64.

Wang, L., Yang, Y., Min, R. and Chakradhar, S. (2017), "Accelerating deep neural network training with inconsistent stochastic gradient descent", *Neural Networks*, Elsevier, Vol. 93, pp. 219–229.

Widrow, B., Greenblatt, A., Kim, Y. and Park, D. (2013), "The No-Prop algorithm: A new learning algorithm for multilayer neural networks", *Neural Networks*, Elsevier Ltd, Vol. 37, pp. 182–188.

Wilamowski, B.M., Cotton, N.J., Kaynak, O. and Dundar, G. (2008), "Computing gradient vector and Jacobian matrix in arbitrarily connected neural networks", *IEEE Transactions on Industrial Electronics*, Vol. 55 No. 10, pp. 3784–3790.

Wilamowski, B.M. and Yu, H. (2010), "Neural network learning without backpropagation", *IEEE Transactions on Neural Networks*, Vol. 21 No. 11, pp. 1793–1803.

Wilson, D.R. and Martinez, T.R. (2003), "The general inefficiency of batch training for gradient descent learning", *Neural Networks*, Elsevier, Vol. 16 No. 10, pp. 1429–1451.

Wong, T.C., Haddoud, M.Y., Kwok, Y.K. and He, H. (2018), "Examining the key determinants towards online pro-brand and anti-brand community citizenship behaviours: a two-stage approach", *Industrial Management & Data Systems*, Emerald Publishing Limited, Vol. 118 No. 4, pp. 850–872.

Yang, Y., Wang, Y. and Yuan, X. (2012), "Bidirectional extreme learning machine for regression problem and its learning effectiveness", *IEEE Transactions on Neural Networks and Learning Systems*, IEEE, Vol. 23 No. 9, pp. 1498–1505.

Yeung, D.S., Ng, W.W.Y., Wang, D., Tsang, E.C.C. and Wang, X.-Z. (2007), "Localized Generalization Error Model and Its Application to Architecture Selection for Radial Basis Function Neural Network", *IEEE Transactions on Neural Networks*, IEEE, Vol. 18 No. 5, pp. 1294–1305.

Yin, X. and Liu, X. (2018), "Multi-Task Convolutional Neural Network for Pose-Invariant Face Recognition",

IEEE Transactions on Image Processing, Vol. 27 No. 2, pp. 964–975.

Ying, L. (2016), "Orthogonal incremental extreme learning machine for regression and multiclass classification", *Neural Computing and Applications*, Springer, Vol. 27 No. 1, pp. 111–120.

Ypma, T.J. (1995), "Historical Development of the Newton–Raphson Method", *SIAM Review*, Society for Industrial and Applied Mathematics, Vol. 37 No. 4, pp. 531–551.

Zaghloul, W., Lee, S.M. and Trimi, S. (2009), "Text classification: neural networks vs support vector machines", *Industrial Management & Data Systems*, Vol. 109 No. 5, pp. 708–717.

Zeiler, M.D. (2012), "ADADELTA: An Adaptive Learning Rate Method", available at:https://doi.org/http://doi.acm.org.ezproxy.lib.ucf.edu/10.1145/1830483.1830503.

Zhang, G.P. (2000), "Neural networks for classification: a survey", *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, IEEE, Vol. 30 No. 4, pp. 451–462.

Zong, W., Huang, G.-B. and Chen, Y. (2013), "Weighted extreme learning machine for imbalance learning", *Neurocomputing*, Elsevier, Vol. 101, pp. 229–242.

Zou, W., Xia, Y. and Li, H. (2018), "Fault Diagnosis of Tennessee-Eastman Process Using Orthogonal Incremental Extreme Learning Machine Based on Driving Amount", *IEEE Transactions on Cybernetics*, IEEE, Vol. 48 No. 12, pp. 3403–3410.