Review

Artificial neural network applications in the calibration of spark-ignition engines: An overview

Richard Fiifi Turkson\textsuperscript{a,b,c,*}, Fuwu Yan\textsuperscript{a,b}, Mohamed Kamal Ahmed Ali\textsuperscript{a,b,d}, Jie Hu\textsuperscript{a,b}

\textsuperscript{a}School of Automotive Engineering, Wuhan University of Technology, Wuhan 430070, China
\textsuperscript{b}Hubei Key Laboratory of Advanced Technology for Automotive Components, Wuhan University of Technology, Wuhan 430070, China
\textsuperscript{c}Mechanical Engineering Department, Ho Polytechnic, P.O. Box HP 217, Ho, Ghana
\textsuperscript{d}Automotive and Tractors Engineering Department, Faculty of Engineering, Minia University, 61111 El-Minia, Egypt

\begin{abstract}
Emission legislation has become progressively tighter, making the development of new internal combustion engines very challenging. New engine technologies for complying with these regulations introduce an exponential dependency between the number of test combinations required for obtaining optimum results and the time and cost outlays. This makes the calibration task very expensive and virtually impossible to carry out. The potential use of trained neural networks in combination with Design of Experiments (DoE) methods for engine calibration has been a subject of research activities in recent times. This is because artificial neural networks, compared with other data-driven modeling techniques, perform better in satisfying a majority of the modeling requirements for engine calibration including the curse of dimensionality; the use of DoE for obtaining few measurements as practicable, with the aim of reducing engine calibration costs; the required flexibility that allows model parameters to be optimized to avoid overfitting; and the facilitation of automated online optimization during the engine calibration process that eliminates the need for user intervention. The purpose of this review is to give an overview of the various applications of neural networks in the calibration of spark-ignition engines. The identified and discussed applications include system identification for rapid prototyping, virtual sensing and electronic control look-up table surrogates, emerging control strategies and On-Board Diagnostic (OBD) applications. The demerits of neural networks, future possibilities and alternatives were also discussed.
\end{abstract}

\begin{keywords}
Artificial neural networks
Applications
Spark-ignition engines
Calibration
\end{keywords}

\begin{document}
\maketitle

\begin{section}
1. Introduction
\end{section}

\begin{section}
2. Artificial neural network theory
\end{section}

\begin{subsection}
2.1. The perceptron model
\end{subsection}

\begin{subsection}
2.2. Single- and multi-layer perceptron feedforward networks
\end{subsection}

\begin{subsection}
2.3. Radial-basis function neural networks
\end{subsection}

\begin{subsection}
2.4. Recurrent neural networks
\end{subsection}

\begin{subsection}
2.5. Neural network learning algorithms
\end{subsection}

\begin{subsubsection}
2.5.1. Learning algorithms for multi-layer feedforward networks
\end{subsubsection}

\begin{subsubsection}
2.5.2. Learning in recurrent neural networks
\end{subsubsection}

\begin{section}
3. Neural network applications in the development of engine management systems
\end{section}

\begin{subsection}
3.1. System identification for accelerated system development
\end{subsection}

\begin{subsection}
3.2. Virtual sensing and electronic control look-up table surrogates
\end{subsection}

\begin{subsection}
3.3. Emerging control strategies
\end{subsection}

\begin{footnotes}
* Corresponding author at: School of Automotive Engineering, Wuhan University of Technology, Wuhan 430070, China.
E-mail addresses: riturkus@yahoo.com, rturkson@hopoly.edu.gh (R.F. Turkson).

Peer review under responsibility of Karabuk University.
\end{footnotes}

\begin{doi}
http://dx.doi.org/10.1016/j.jestch.2016.03.003
\end{doi}

\begin{copyright}
2016 Karabuk University. Publishing services by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
\end{copyright
1. Introduction

An ever tightening emission legislation requirements around the globe coupled with an increasing customer desire for improved vehicle performance has been among the drivers of new engine technologies [74,108,125]. New technologies include the use of variable cam timing/phasing, variable intake manifold length, charge motion valves for inducing swirl and direct gasoline injection systems with variable injection timing [65,59,53,139,43,100,3,88,20,52,73,133,48]. As the number of actuators associated with these new technologies increases, it introduces an exponential dependency between the time and cost outlays for mapping such complex (high degree of freedom) engines. Furthermore, the increase in the number of the degrees of freedom increases the number of input and output combinations that would have to be tested in order to achieve the calibration objective, making the calibration task virtually impossible. One way of dealing with the increasing complexity in the engine calibration and mapping task is to use the model-based calibration approach which involves the use of statistical methods of Design of Experiments (DoE) methodology [80,115,57,76,36,40], for reducing the number of engine tests that would be representative of the entire operating space. The traditional look-up tables resulting from the calibration and engine mapping process increase in size with an increase in engine complexity leading to an increase in the electronic control unit (ECU) memory space required for storing the look-up tables [5,122]. The use of look-up tables does not guarantee output accuracy over the operating range of an engine. Furthermore, it is not easy to improve accuracy by establishing the complex relationships between the input variables. The noisy nature of parameters used in the data acquisition system of a dynamometer test grid may compound the accuracy problem further. Trained artificial neural networks present a black-box approach [136] for matching sensor inputs with outputs in a non-linear manner that is free from noise [121,86]. For this reason, artificial neural networks are gaining attention for producing outputs that are accurate regardless of the variation in input parameters [91,8,7,137].

Artificial neural networks represent one of three modeling techniques. These techniques include white-, gray- and black-box modeling with neural networks belonging to the latter technique. The difference between these techniques is the level of physical knowledge available about the model and the extent to which the user could interpret the models. The white-box modeling technique relies on the use of physical laws in establishing the relationships between variables and parameters. Similarly, in a black-box model, it is possible to match inputs to outputs without any clear knowledge about the relationship between the inputs and corresponding outputs. Furthermore, gray-box models are in-between the white- and black-box modeling techniques, in terms of the level of model transparency [77,50,104,96,37,17].

It is evident from Fig. 1 that white-box models have the highest transparency level regarding the extent to which the model is understood, with black-box models having the lowest transparency levels. In-between these two extremes, come gray-box models with a transparency level lying between that of white- and black-box models. The lack of transparency on the part of artificial neural networks comes in handy in situations where it is virtually impossible to obtain a description of the various parts of a system using physical equations as in the case of white- and gray-box modeling. The black-box modeling approach presented by artificial neural networks allows the input and output relationship to be modeled with no concern over the physical details of system parts.

Furthermore, artificial neural networks represent one of two classical data-driven modeling techniques, with polynomial regression models representing the other. While polynomial regression models are able to approximate the input–output relationship fairly well for steady-state engine operation, it is not able to capture the non-linearity in a global engine model satisfactorily. This is normally due to the fact that polynomial models could be prone to measurement errors including noise and outliers. As an alternative, artificial neural networks are able to capture the non-linear behavior better than polynomial regression models. However, a great deal of know-how is required for training the artificial neural networks in a manner that avoids overfitting and free from measurement errors [58,112].

According to Berger [17], while additional layers of neurons could be used to ensure that an artificial neural network model is flexible enough to capture the non-linear behavior of an engine, Bayesian regularization methods could be used to avoid overfitting [49] which results as the number of model parameters (weights and biases) increases. Regularization would make the neural network behave as if it had fewer parameters for the avoidance of overfitting and combining this with an increased number of neuron layers makes an artificial neural network robust to errors resulting from noisy data measurements. The requirements of engine modeling include the curse of dimensionality; the use of DoE for obtaining few measurements as practicable, with the aim of reducing engine calibration costs; the required flexibility that allows model parameters to be optimized to avoid overfitting; and the facilitation of automated online optimization during the engine calibration process. While the Gaussian Process modeling technique [16] satisfies the aforementioned modeling requirements for engine calibration, there are situations in which other data-driven non-linear modeling techniques could be useful. Berger [17] proffered that in the event that it is difficult to establish a Gaussian Process model because of a large number of measurements, alternative non-linear modeling methods like artificial neural networks, relevance vector machines (RVM) or support vector machines (SVM) could be used. Furthermore, while it is prudent to use a polynomial regression model for problems with a low degree of complexity, it is important to use a local linear model to solve problems that require human intervention for a large number of measured data. The aim of this review is to give an overview of relevant neural network theory and outline the main applications in the area of spark-ignition engine management system development.
output signal of the neuron model within a range of $[0,1]$ or $[-1,1]$. The input to the neuronal model also includes a bias $b$, for controlling the magnitude of the input $g$ for the activation function $f$. The magnitude of $g$ is increased when the bias is positive and is decreased when the bias is negative [22,105,60,93].

The following equations could be used to describe the neuronal model shown in Fig. 2:

$$t_k = \sum_{i=1}^{n} x_i w_{ik}$$

(1)

$$a = f(t_k + b)$$

(2)

where, $x_i$ represents the inputs and $w_{ik}$ the weights. The sum of the weighted inputs is represented by $t_k$; the activation function by $f$; the bias by $b$ and the model output by $a$. The following equation is also valid for the neuron model in Fig. 2:

$$g_k = t_k + b$$

(3)

where $g_k$ represents the output of the addition block shown in Fig. 2.

The output of the neuron model in Fig. 2 is defined by the activation function, also transfer function denoted by $f$. The activation function is usually one of different types including the Heaviside, Piecewise-linear, Log-sigmoid and the Tan-sigmoid functions as shown in Table 1.

The Tan-sigmoid function is derived from the Log-sigmoid function with both functions representing the most commonly used activation functions for the design of artificial neural networks. According to Haykin [60], allowing the output of a neuronal model (using the Tan-sigmoid function) to assume values between $-1$ and $1$, rather than between $1$ and $0$ (for the Log-sigmoid function) yields additional analytical benefits.

Another type of activation function is the Gaussian type of function that is normally used in radial basis function (RBF) neural networks. The Gaussian function is the probability density function of a randomly normally distributed variable having an expected value or mean $(\mu)$ and a variance $(\sigma^2)$ as indicated in Table 1 [79,107].

### 2.2. Single- and multi-layer perceptron feedforward networks

As mentioned earlier, single- and multi-layer perceptron networks are types of feedforward neural networks [66,13,69,70]. For a single-layer feedforward network, there is an input layer and an output layer. The input layer is not counted since no computation takes place in this layer. There is a unidirectional connection between the input nodes, in a single-layer network, and the output layer of neurons. Fig. 3 shows the layout of a single-layer feedforward network having three nodes in the input layer and three neurons in the output layer.

The main distinction between a single-layer and multi-layer feedforward network is that the latter could have either two or more layers of hidden neurons. Multi-layer feedforward neural networks have input signals applied to the first hidden layer of computational neurons. The weighted output from the first layer of neurons serves as the input for the next layer of neurons. It has been established that increasing the number of hidden layers improves the ability of the network (in terms of capturing the non-linear dependencies between the inputs and outputs), especially in the case when there is a large number of network inputs [32,60]. The layout of a multi-layer feedforward network is shown in Fig. 4. This network could be described as a 3-3-2 type of network because it has three input nodes, three neurons in the only hidden layer and two neurons in the output layer.

The benefits that may accrue via the use of multi-layer feedforward artificial neural networks include the following:

- Increased ability to capture non-linear relationships.
- Better generalization to unseen data.
- Improved performance in complex tasks.

### 2.1. The perceptron model

Perceptron models are considered as the building blocks for artificial neural networks. The perceptron model (Fig. 2) comprises synaptic links which allow the inputs $(x_1, x_2, \ldots, x_n)$ to be weighted by applying the weights $(w_1, w_2, \ldots, w_n)$. Other features of the perceptron model include the block for adding the weighted input signals and an activation function for limiting the amplitude of the signal. This network could be described as a 3-3-2 type of network because it has three input nodes, three neurons in the only hidden layer and two neurons in the output layer.

The benefits that may accrue via the use of multi-layer feedforward artificial neural networks include the following:

- Increased ability to capture non-linear relationships.
- Better generalization to unseen data.
- Improved performance in complex tasks.

### 2. Artifical neural network theory

A neural network comprises a single or a number of interconnected neurons similar to those found in biological nervous systems. A neural network can be trained to respond to a number of given inputs without any particular mathematical relationship between the inputs and outputs. The taxonomy of artificial neural networks is such that they could be classified broadly into two classes: feedforward and recurrent neural networks. Feedforward neural networks could further be classified into single-layer perceptron networks, multi-layer perceptron networks and radial basis function networks. Similarly, recurrent neural networks could also be classified further into competitive and Hopfield type of networks, together with other network types [75,34]. Variants of both neural network classifications find applications in the design and development of engine management systems and would therefore be described in subsequent sections.

#### 2.1. The perceptron model

Perceptron models are considered as the building blocks for artificial neural networks. The perceptron model (Fig. 2) comprises synaptic links which allow the inputs $(x_1, x_2, \ldots, x_n)$ to be weighted by applying the weights $(w_1, w_2, \ldots, w_n)$. Other features of the perceptron model include the block for adding the weighted input signals and an activation function for limiting the amplitude of the signal. This network could be described as a 3-3-2 type of network because it has three input nodes, three neurons in the only hidden layer and two neurons in the output layer.

The benefits that may accrue via the use of multi-layer feedforward artificial neural networks include the following:
1. With the availability of data, feedforward networks could be used for modeling a broad spectrum of systems.

2. Feedforward neural networks could be useful for applications where analytical methods are yet to be discovered.

3. Using a suitable neural network size, it is possible to capture the non-linear dependencies between the input and outputs to a reasonable degree of accuracy.

2.3. Radial-basis function neural networks

Radial basis function neural networks are one of the variants of feedforward neural network types. Such neural networks are used for executing mapping functions of the form $f: \mathbb{R}^n \rightarrow \mathbb{R}$ based on the following equation:

$$f(x) = \sum_{i=0}^{m} W_i \phi(||x - c_i||)$$  \hspace{1cm} (4)

where $x \in \mathbb{R}^n$ represents the input vector, $\phi(.)$ is a non-linear transformation function, $||.||$ represents the Euclidean distance, $W_i$ having a range of $1 \leq i \leq m$ represents the weights, $c_i \in \mathbb{R}^n$ having a range of $1 \leq i \leq m$ denotes the kernel nodes or centers, with $m$ representing the number of kernel nodes. A radial basis function network based on Eq. (4) is shown in Fig. 5.

For the radial basis function neural network shown in Fig. 5, the least square method [6,31,132,25] could be used in determining the weights $W_i$ by fixing the centroids $c_i$ and the function for non-linear transformation of the inputs $\phi(.)$ and applying the inputs and target outputs to the network [26,107,135,27,18]. The prime benefit of using radial basis function feedforward neural networks is the reduced time required for training which results from the simpler structure of these networks compared to the multi-layer perceptron networks described earlier. Radial-basis functions also have comparatively low extrapolation errors and are generally more reliable.

2.4. Recurrent neural networks

As mentioned earlier, recurrent neural networks represent the second broad classification of neural networks. These network types will normally have one or more feedback loops with unit-delay operators represented by $z^{-1}$ (Fig. 6). In its simplest form, a recurrent neural network comprises a single layer of neurons with output signals from each serving as input signals for other neurons of the network as shown in Fig. 6.

Another variant of this network type is to have the output of each neuron channeled back to its input. Other recurrent neural networks may have one or more hidden layers akin to multi-layer
feedforward networks and are normally used for modeling the non-linear dynamical behavior of systems [129,46,60]. Recurrent
neural networks have a very high level of computational power
and could be used to model virtually any non-linear dynamical
system to any degree of accuracy. With the use of sufficient hidden
layers, the modeling is normally without any restrictions on the
size of the state space.

2.5. Neural network learning algorithms

Apart from knowledge representation and reasoning, the ability
to learn is a vital feature of artificial intelligence systems. The def-
nition of learning could be varied and relative. However, from an
artificial neural network viewpoint, learning could be described
as the process of adjusting the network’s parameters (weights
and biases) such that desired outputs could be predicted based
on a given number of inputs and within acceptable limits of error.
The manner in which the network weights are adjusted determines
the learning algorithm type [72,60]. Since the solution to most
practical problems involve the use of multi-layer feedforward
and recurrent neural networks, the subsequent sections will
attempt a discussion of the learning algorithms used for these
two major classes of neural networks.

2.5.1. Learning algorithms for multi-layer feedforward networks

The backpropagation learning algorithm is popular with super-
vised learning paradigms and has proven to be useful for solving a
spectrum of practical problems. This learning algorithm involves
the forward propagation of input signals, computing the error
signal (difference between target output and actual output) and
backpropagating the error until the error signal is minimized for
a number of training iterations or epochs. For a feedforward
network having an output \( y^{(k)}_i \) in the \( i \)-th neuron of layer \( k \) and a bias
\( b^k_i \), the relationship between the input and output of the network
could be represented as follows:

\[
y^{(k)}_i = \phi \left( \sum_{j=1}^{k-1} w^{(k-1)}_{ij} y^{(k-1)}_j + b^k_i \right)
\]  

where \( \phi \) is the activation function, \( w^{(k-1)}_{ij} \) represents the weight
input from the neuron \( j \) of the layer denoted by \( k - 1 \) to the neuron
\( i \) of the layer represented by \( k \) with \( y^{(k-1)}_j \) denoting the output of the
neuron \( j \) in layer \( k - 1 \).

If a training set of data has vector inputs \((x_1, x_2, \ldots, x_n)\) and tar-
ger vector outputs \((y_1, y_2, \ldots, y_n)\), then the weights of the network
could be iteratively adjusted until the network is able to match
each network input \( x(k) \) with a target output \( y(k) \) with a minimum
of error. For the backpropagation algorithm, the error between the
input signal and the target output could be defined as follows:

\[
E = \sum_{n=1}^{n} ||y_k - y_k||^2
\]

From Eq. (6), it could be said that for an input value of \( x(k) \) we
have a corresponding network output vector of \( y(k) \). The training of a
neural network via the backpropagation algorithm starts with
setting the network weights to small random values and then updating the weights repeatedly until the error \( E \) is minimized \([72,111,24]\). The weights and biases are adjusted using the gradient descent method \([10,21,15]\) as follows:

\[
w^{(k-1)}_w(p+1) = w^{(k-1)}_w(p) - \eta \frac{\partial E}{\partial w^{(k-1)}_w}
\]

(7)

\[
b^{(k)}(p+1) = b^{(k)}(p) - \eta \frac{\partial E}{\partial b^{(k)}}
\]

(8)

where \( \eta \) represents the step size.

With weight update according to the delta rule \([113,116]\), the network weights and biases could be updated according to the following equations:

\[
w^{(k-1)}_w(p+1) = w^{(k-1)}_w(p) + \eta \delta^{(k)}_j
\]

(9)

\[
b^{(k)}(p+1) = b^{(k)}(p) + \eta \delta^{(k)}
\]

(10)

For Eqs. (9) and (10), \( \delta^{(k)}_j \) represents the error for the \( k \)th time step and \( b^{(k)} \) is the layer \( k \). The following equation could be used to determine the error in the last layer \( K \):

\[
\delta^{(K)}_j = \varphi'[\sum_{k=2}^{K} w^{(k-1)}_j y^{(k-1)} + b^{(k)}] (y_j - y_j^K)
\]

(11)

The error term for the rest of the layers is determined through backpropagation based on Eq. (12).

\[
\delta^{(k-1)}_j = \varphi'[\sum_{k=2}^{K} w^{(k-2)}_j y^{(k-2)} + b^{(k-1)}] \sum_{k=1}^{K} (w^{(k-1)}_j \delta^{(k)}_j)
\]

(12)

2.5.2. Learning in recurrent neural networks

One distinguishing feature of a recurrent neural network is its ability to characterize non-linear dynamical systems. One of the learning algorithms used for recurrent neural networks is the Backpropagation Through Time (BPTT) algorithm \([128,12,33]\). This algorithm is derived from the backpropagation learning algorithm used for feedforward neural networks based on gradient descent methods. The BPTT algorithm requires that the network weights and biases are adjusted based on network states and inputs at previous time steps. For a recurrent neural network which is trained for a time interval that spans between \( t_1 \) and \( t_n \), Eq. (13) could represent the total cost function.

\[
E(t_1, t_n) = \sum_{t=t_1}^{t_n} E_{\text{step}}(t)
\]

(13)

where the total cost function is the sum of errors for each time step, \( E_{\text{step}}(t) \), from \( t_1 \) to \( t_n \).

Applying the gradient descent method, the network weights are adjusted from previous time steps based on Eq. (14).

\[
\Delta w_j = -\eta \sum_{t=t_1}^{t_n} \frac{\partial E(t_1, t_n)}{\partial w_j}
\]

(14)

With the BPTT algorithm, online learning takes place in which the weights are adjusted for each time step. A pre-requisite for this to take place is to memorize previous input states at previous time steps.

3. Neural network applications in the development of engine management systems

Artificial neural networks have received a lot of attention in recent years for solving scientific problems including prediction, system control, pattern recognition and system optimization as well as model-based engine calibration applications. The subsequent sections will focus on artificial neural networks and their applications in the area of spark-ignition engine calibration.

3.1. System identification for accelerated system development

According to Rutherford and Cole \([114]\) artificial neural networks have become attractive for non-linear system identification for modeling and optimal control purposes. The methods of Design of Experiments (DoE) have been used to obtain a reduced number of experiments that could be conducted using a dynamometer but this is only ideal for steady state conditions. In order to obtain information about the dynamic behavior of an engine (including transients), it would be necessary to run the engine under defined operating transient cycles and other conditions. The data collected this way could then be used in training neural networks that would represent the dynamic response of the real engine to inputs and could be used for offline optimization of the calibration process. It is also possible to use neural networks in the area of feedback control systems. For this particular application, neural networks are used to capture the non-linear relationships between inputs and outputs of the identified system in a way that makes it easy to calibrate the PID control gains \([97]\). Wu et al. \([130]\) proposed the use of an artificial neural network for determining the air flow rate for a variable valve timing spark-ignition engine. The proposed method involved the training of an artificial neural network (ANN) model with data obtained from the steady-state testing of the engine. The inputs for training the ANN model included engine speed, manifold absolute pressure, intake and exhaust camshaft phasing, with the network output being the air flow rate. After using a heuristic approach it was shown in the study \([130]\) that the chosen ANN architecture (4-8-8-1) for the model was able to predict air-flow rate accurately (within an error margin of ±10%) for all the input variables considered. Even though the method used for selecting the preferred neural network structure employed a trial and error approach, it corresponds to the explicit method of optimizing the structure of an artificial neural network. The explicit method involves the gradual increase in the number of hidden neurons and/or layers and observing the training and test errors as the size of the network grows larger.

Another way in which neural networks could be used for system identification for speeding up the calibration process through the use of a virtual engine which correlates well with a target engine and the subsequent use of the virtual engine for running high fidelity simulations was reported in the study conducted by Wu et al. \([131]\). In this study \([131]\) an attempt was made to maximize engine torque at selected engine speeds at a wide throttle opening. Four input variables were optimized at the same time but the focus was on achieving an optimum cam phasing as this had a direct effect on the volumetric efficiency. The inputs used in this case included intake cam position, exhaust cam position, throttle angle and engine speed with engine torque as the output. The optimization framework used by the study required the determination of the various input combinations for the high-fidelity engine model using a Latin Hypercube Sampling technique \([61,44,71]\) which assumes that there is a uniform distribution of input combinations over the entire operating range under consideration \([131,95]\). In order to reduce the computational time, which may result if the high-fidelity engine model were used directly for the optimization effort, the simulation results were used for the training of an artificial neural network surrogate model which had the capability of predicting the response from the high-fidelity engine model as a function of the various independent input variable combinations. After comparing neural network models with different levels of complexity in terms of the number
of neurons in the hidden layers and the number of hidden layers, the chosen neural network had a structure of 5–10–10–1. It is worth noting that the difference between the works reported by Wu et al. [130] and Wu et al. [131] lies in the introduction of a virtual system (high-fidelity engine model) for generating data for training the artificial neural network surrogate model. It is also worth noting that the number of neuron layers was the same (4) for both studies except for an increase in the number of neurons in the first three layers from 4, 8, 8 to 5, 10, 10 in the latter study [131]. This indicated that a slightly more flexible neural network structure was required for capturing the non-linear behavior of the system being modeled in the latter study. Although there was no mention of specific fit statistics in the study carried out by Wu et al. [131], it was stated that the use of three-hidden-layer neural networks yielded lower mean squared errors for both training and testing in comparison with two- and one-hidden layer neural networks. Furthermore, it is also worth pointing out that in both studies [130,131] the number of inputs (4) for predicting single outputs were the same and consistent with the input dimensional requirements for engine calibration. While the study conducted by Wu et al. [130] generated experimental data involving 4789 operating points for neural network training and testing purposes, Wu et al. [131] employed an automated system for generating data for fewer number of test combinations (1025 via Latin Hypercube Sampling) required for training the neural networks. The former approach for data collection would require an extensive use of dynamometer tests, the cost of which could be justified by a reduced neural network model error as a result of the higher number of experimental data collected (See Section 4.1). It is also worth pointing out that while the fewer test combinations obtained by the latter approach would be statistically representative of the operating space considered, the use of the high-fidelity engine model could also avoid the extensive dynamometer tests, ultimately leading to a reduction in the cost outlays for data collection.

An example of system identification and characterization using neural networks is demonstrated by a study conducted by Papadimitriou et al. [106] in which a metamodeling or model reduction process [120] was used to simplify computationally expensive engine parts. This way, it was possible to increase the computational speed of the model while preserving its fidelity to a large extent. The objective of the model reduction process was to achieve a fast-running model for rapid control prototyping applications involving Hardware-In-the-Loop (HIL) and Software-In-the-Loop (SIL) testing [134,119,2,64]. Model reduction via metamodeling was achieved by replacing the intake and exhaust manifolds and the engine cylinders with mean value models. This form of system identification used by Papadimitriou et al. [106] was similar to that reported by Wu et al. [131] with respect to the use of DoE simulation results for training neural networks. The difference was in the replacement of the high-fidelity engine model with a fast-running mean-value engine model in the work of Papadimitriou et al. [106]. It is worth noting that a distinguishing feature of the study conducted by Papadimitriou et al. [106] was the iterative construction and training of different neural networks and selecting the best performing network based on speed and accuracy criteria. The use of a fast-running mean-value engine model in place of a high-fidelity one could lead to a reduction in the required simulation time for data collection. The resulting benefits would be more valuable given that in the shift to the use of the fast-running model, fidelity is preserved to a large extent. The inputs for training the neural network models included engine speed, intake manifold pressure, intake manifold temperature and air–fuel ratio. The selection of the preferred neural networks was based on both the explicit and implicit (involving the use of regularization techniques applied to change the model complexity without changing the nominal number of parameters used) methods of model structure optimization in addition to a novel approach for automatically determining the quality of fit. The study [106] reported the qualitative and quantitative preservation of engine performance metrics of the detailed model, with the error for the average fuel consumption model being around 0.54%.

A study conducted by Müller and Schneider [99] described how recurrent neural networks proposed by Narendra and Parthasarathy [102] could be used to approximate the output torque of an engine. Recurrent neural networks were preferred for the study due to their dynamic nature which allowed the network to give an output based on the inputs at any given time and the history of the system [5,4]. Torque estimation by the recurrent neural networks was carried out by the extraction of relevant sensor signals obtained over the operation range of the engine. The sensor input signals considered included in-cylinder combustion pressure, air–fuel ratio, ignition timing, throttle angle, engine temperature, mass of air flow via intake manifold, manifold pressure, injection timing and gear shift information. This information was used to train an artificial neural network model to establish the nonlinear relationship between the sensor inputs and engine torque. This way, it was possible for the dynamic neural network model to approximate the engine output torque based on the inputs. Torque control was achieved using a combination of a controller and dynamic neural network models in a manner that feeds the controller with target inputs and the actual output from the model network. The controller network computed the squared error between its inputs and attempted to minimize the squared error via the use of the gradient descent learning algorithm proposed by Puskorius and Feldkamp [109]. In order to simplify the neural network for the purpose of training the controller used for the study [99], only the throttle was selected as the actuator for controlling the engine output torque. The simplified neural network model was then trained until the network was able to give the desired torque output within acceptable measurement tolerances. A similar study conducted by Wu et al. [131] used fewer inputs (4) compared with the increased number of inputs (9) for the study conducted by Müller and Schneider [99]. However, the increased number of inputs for the latter study was still consistent with the required input dimensions (5–10 inputs) for engine calibration tasks [17]. For the same application (prediction of torque), the latter study used a recurrent neural network having a single hidden layer of 12 neurons. This is consistent with the inherent characteristic of recurrent neural networks regarding the ability to approximate any dynamic system regardless of the number of hidden layers and/or neurons used. Offline training of the controller neural network was achieved by feeding the output of the controller neural network to the simplified model of the engine with the controller having the target torque, actual torque and the last control action as inputs for deciding on how to compute the next output as illustrated in Fig. 7.
During the training process the difference between the target and actual output torque was minimized through the gradient descent method proposed by Puskorius and Feldkamp [109] with the aim of adjusting the controller neural network weights that would give the desired controller performance. According to Müller and Schneider [99], the training of the controller could also be done online using a real engine.

3.2. Virtual sensing and electronic control look-up table surrogates

It is important to use sensors to monitor the operating condition of an engine in a manner that will allow the control system to make adjustments to the various actuators in order to achieve the best combination between driveability, fuel economy and emissions. When sensors are used in this manner, a number of look-up tables are used for backup and plausibility checks for the engine management system [110,97]. Previous studies [97,101] proffer that an alternative to the use of sensors in an engine management system is to use trained neural networks as surrogates for physical sensors or used as additional sensors for implementing emerging control strategies. It was also stated that it was possible to achieve cheap torque based engine management systems with the use of neural networks as feedback sensors.

Virtual sensing methods for designing reliable diagnostic and fault-tolerant control systems using recurrent neural networks were investigated by Kamat et al. [78]. The work involved the development and implementation of virtual models of the important spark-ignition engine input variables namely manifold absolute pressure (MAP), mass airflow rate (MAF), normalized air–fuel ratio (Lambda) and engine output torque (T). Each of the neural network models developed had a multi-input and a single output. For the MAP model, the inputs were fuel pulse width (FPW), MAF, engine speed (RPM) and throttle position sensor (TPS) signals. The MAF model had the same inputs as the MAP model except for the replacement of the MAP input with the MAP input signal. Similarly the Lambda model had the FPW, MAP, MAF, TPS and RPM signals as inputs. The torque prediction model also had the spark-timing angle, FPW, MAF, TPS and RPM as inputs. All the recurrent neural network models were trained with experimental data collected on a real engine, using gradient descent methods that aimed at reducing the normalized mean squared error between the target outputs and the actual network output. After the training (comparing recurrent neural network models having a single layer with different number of neurons from 8 to 14), the neural network structures for all the models of the virtual sensors for the prediction of manifold absolute pressure, manifold air flow rate and torque were of a single layer of 14 neurons. For the Lambda model the lowest normalized mean squared error (NMSE) of 0.0011 was achieved while using the Federal Test Protocol (FTP) in combination with a tuned controller. Similarly, the minimum NMSE of 0.002 was achieved for the torque model using the FTP and a tuned controller. The MAP and MAF models had the same and lowest NMSE (0.0006) for the same conditions under the FTP and the use of the tuned controller. On the other hand, inferring the air fuel ratio (lambda) from other sensor inputs did not work well for closed loop control because of the reduced frequency range. It was however found that the virtual lambda sensor could be used successfully for diagnostic and fault-tolerant control purposes. Kamat et al. [78] proposed that increasing the operating frequency range of the virtual lambda sensor should be the focus of further research. In an attempt to investigate the failure modes of the oxygen (lambda) sensor used on a gasoline engine and its effect on fuel metering and emissions, Hu et al. [68] used a virtual oxygen sensor based on an Elman recurrent neural network. The virtual oxygen sensor predictions were used to make adjustments to the air–fuel ratio in the failure modes of the physical sensor used for the fuel system.

New technologies (such as variable cam phasing, charge motion valves and variable intake manifold runner lengths) for improving engine performance, fuel economy and emissions do not only make the calibration process complex and prohibitive, but also leads to a large memory space for storing the look-up tables resulting from the calibration process. According to Meyer and Greff [97], it has been established through investigations that artificial neural networks occupy less of the random access memory space than conventional look-up tables and that equation-based neural networks will replace look-up tables in the future. In the work of Malacynski et al. [91], artificial neural networks were used to model real-time volumetric efficiency of a target engine. The possibility of using an artificial neural network surrogate to derive volumetric efficiency for an engine equipped with variable valve actuation was investigated. The inputs to the neural network model included valve lift, intake valve phasing, exhaust valve phasing and the ratio of absolute exhaust pressure to absolute intake manifold pressure. While the study reported the use of a single hidden-layer of neurons no information regarding the number of neurons was reported for the feedforward neural network. The choice of a single layer of hidden neurons for this application in combination with the linear activation function for the output neuron was motivated by the desire to operate the neural network in real time without consuming a high amount of processing resources. In an attempt to reduce the mean squared error that could result from the continuously variable nature of the inputs, vector quantization was employed for the approximation of continuous-amplitude signals with discrete amplitude signals. It was shown that the trained neural network surrogate proved to be a good replacement for look-up tables, with the training absolute mean squared error below 5% for a majority of the cases considered. However, implementation of the neural network algorithm for evaluating volumetric efficiency was prone to the limitations of available automotive computers in terms of random access memory, read only memory, computational resources and fixed-point math [41]. In the work of El Hadef et al. [40] an artificial neural network was proposed for predicting the volumetric efficiency of a downsized, turbo-charged spark-ignition engine. Steady-state and transient engine operating points were used for model validation with only steady-state engine operating points being used for model calibration. It was found that the model predicted volumetric efficiency satisfactorily while satisfying the computational requirements of modern automotive processors. El Hadef et al. [40] also noted that in spite of the fact that statistical Design of Experiments could be used to reduce the training data set without any adverse effect on model fidelity, it is imperative to further analyze and determine the minimum number of points that would yield the best results for each case.

3.3. Emerging control strategies

Proportional, Integral and Derivative (PID) control systems have been used in the past for closed-loop engine management systems. The limitation of such PID systems is the effort required to calibrate the PID gains to give the required system response for a given set of controller inputs. Feedback control systems based on neural networks have been identified for an enhanced control system performance, particularly in situations where there is a delay introduced by slow sensor dynamics in relation to the measured quantity [97,14,89,90]. In order to ensure that the catalytic converter efficiency stays fairly high, it is important to use an air–fuel ratio that is close to stoichiometric. Air–fuel control was normally achieved using the signal from a lambda sensor for either enriching or weakening the air–fuel mixture. However, the dynamic nature
of engines coupled with its changing operating conditions makes it difficult for the control system to reconcile the oxygen content in the exhaust gas and the gravimetric air–fuel ratio under dynamic conditions. Neural networks can be used to achieve control systems that will minimize the deviation of the air–fuel ratio from that which is stoichiometric while keeping under control factors that may cause a variation in the exhaust gas composition notwithstanding the dynamic behavior of the engine [127,28,29,30,51,62,85,97]. The layout of a neural network system for control purposes was proposed in the work of Meyer and Greff [97] in which a neural network controller could be used to compensate for the steady state and dynamic behavior of the plant.
by making adjustments to the weights of the neural network controller based on the difference, $e$, between the actual plant output, $Y_r$ (feedback signal) and the target output, $Y_d$ as illustrated in Fig. 8.

Another control application is to use a trained inverse neural network model which uses actuator set points as inputs such that the network could predict emissions as an output. This trained neural network could be used as a controller in a feedforward structure as illustrated in Fig. 8 [9,97].

In Fig. 9, the engine is fed with a number of inputs, $U$, and generates an output, $Y_r$. The inverse neural network attempts to establish the reverse non-linear relationship between $Y_r$, as input and the original number of inputs, $U$. This is achieved by monitoring the error, $e$, between the outputs of the inverse neural network model, $U_M$ and $U$, and adjusting the weights of the neural network until the error between $U_M$ and $U$ is minimized. This way, a copy of the trained inverse neural network model could be used for feedforward control of the engine.

In an attempt to minimize the error in the estimation of the required air–fuel ratio by an engine for stationary and dynamic operating conditions, Wendeker and Czarnigowski [127] developed a hybrid control system. This involved the use of an adaptive control system and a trained inverse neural network similar to the type described in the work reported by Meyer and Greff [97]. In the study reported by Wendeker and Czarnigowski [127] the performance of the traditional jump-ramp, adaptive and hybrid control systems were compared, with the results indicating that using only the adaptive control system reduced the error of air–fuel ratio estimation relative to the use of the traditional jump-ramp control system that relied on the lambda sensor input. The hybrid control system was found to be the best among the three types of systems as it performed satisfactorily under all engine operating conditions to stabilize the air–fuel ratio. Fig. 10 shows the layout of the hybrid system used for fuel injection control.

3.4. On-Board Diagnostic applications

Virtual sensing using neural networks, as discussed earlier, could be used to develop alternative OBD algorithms. For a spark-ignition engine, estimating the emissions is possible with a neural network virtual sensing system that does not depend on the lambda sensor input. This way, emissions could be estimated in a precise manner, which might not be the case with the use of a physical lambda sensor over a given period of time because of the aging effect and response delay characteristics of the sensor. With this kind of virtual sensing system, it is possible to determine the beginning of sensor deterioration with age. An advantage of using virtual sensing is the ease with which the network compares its inputs and produces the desired output without the need for look-up tables resulting from a cumbersome calibration process. Modern OBD systems monitor all sensors and emission control related devices for correct operation. The application of neural networks comes in handy for providing viable alternative diagnostic algorithms for monitoring OBD system sensors and emission control devices. Using such monitoring systems while the engine is working facilitates the detection of faulty sensors and emission control devices as a consequence of age and mileage [97,54]. In a study conducted by Mcdowell et al. [94], an autoassociative neural network [81,92,35], with an identity mapping feature, was used for fault detection on a spark-ignition engine. A semi-physical model was used to reduce the number of inputs (sensor signals) for the artificial neural network which would otherwise require a great deal of time in building and be computationally complex. The neural network was trained with fault-free sensor data collected over the operating range of the engine and the fault detection capability of the neural network improved by computing the Q statistic (the difference between the measured and predicted variables) of the training data. A fault is detected if the output of the neural network is such that it goes beyond the applied confidence limit. Fig. 11 illustrates fault detection technique using the Q statistic having a confidence limit of 99%.

As a continuation of research work carried out by Grimaldi and Mariani [54] as well as Grimaldi and Mariani [55], an On-Board Diagnostic fault detection system was developed by Grimaldi and Mariani [56] that used trained artificial neural networks to detect and isolate faults on a spark-ignition engine. Experimental data regarding throttle position ($\alpha$), engine speed ($N$), output torque ($T$) and manifold pressure ($P$) were used in training the neural networks. Fault detection and the generation of fault codes was achieved by computing the difference between the neural network calculated values of $\alpha$, $N$, $T$ and $P$ and modified experimental values. The layout of the neural networks and other system units for fault detection is shown in Fig. 12.

In the work reported by Grimaldi and Mariani [56], it was found that the developed fault detection system worked well offline and stated that testing the system online, to make it more robust, would be the focus of future research. At a later date, Capriglione et al. [23] investigated the use of neural networks for online fault detection on automobile engines. Basically, the work done by Capriglione et al. [23] was similar to the study conducted by Grimaldi and Mariani [56] with regard to the use of residuals (difference between measured experimental sensor data and neural network output) and the application of thresholds for the detection of faults. Similarly, the major difference between both investigations lies in the online testing of the fault detection capability of the system developed by Capriglione et al. [23]. In the latter study, testing of the fault detection system was more detailed in the sense that apart from presenting perturbed sensor data to the neural networks, Capriglione et al. [23] went a step further to test the fault detection system online by simulating a short in the sensor circuit by fixing its signal to zero; an open in the sensor circuit by fixing its signal to the maximum possible; holding the sensor output constant and at the previous fault-free value; in addition to simulating a short circuit between two sensors by imposing the output of one sensor on the other. The fault detection system was able to achieve a 100% success in fault identification and detection, with no deceiving signals about faults. Yan and co-worker [67] successfully used a feedforward neural network, based on backpropagation of network errors, for the detection of the engine diagnostic condition of misfire. The neural network used was trained with experimental data collected for normal and abnormal (misfire) engine operation. It was possible for the neural network to detect the different failure modes identified for the study.

![Fig. 11. Use of Q statistic for fault detection](94).
4. Demerits and challenges of artificial neural networks

4.1. Model generalization ability and prediction accuracy

The ability of a neural network model to match input to output data outside the training data set is referred to as generalization. An artificial neural network will generalize satisfactorily if under- and over-fitting could be eliminated. While under-fitting implies the situation where there is a high degree of statistical bias, overfitting refers to the situation where there is a high level of statistical variance between the actual output from a neural network and the target output [83,84,11,103].

The model error $E$ can be decomposed into the bias and variance error components as shown in Eq. (15).

$$E^2 = E_b^2 + E_v$$

where $E_b =$ Bias error and $E_v =$ variance error.

The bias error component is basically due to the restricted model’s structural flexibility that results from the inability of the model to represent a complex system in an exact manner. This error type will always be present even if the model’s parameters (weights and biases) were set to their optimal values. On the other hand, the variance error is due to the number of the model parameters and increases with an increase in the number of neurons for a neural network model. This is in line with the parsimony principle, otherwise known as the Occam’s razor which states that a for different models being considered for system identification, the model with fewer parameters is relatively more accurate in predicting a target output based on a training data set [19,63,45,126]. The variance error component can be approximated by Eq. (16), for an infinitely large training data set.

$$\text{Variance error} \approx \sigma^2 \frac{nn}{N}$$

where $\sigma^2$ represents the variance, $n$ the number of parameters with $N$ representing the number of training data samples. It is worth noting that for a large training data set, the variance error varies almost directly as the number of model parameters, $n$.

The bias error $E_b$ shown in Eq. (15) is negligible for highly flexible models and in such a case the square of the model’s error can be equal to the variance error $E_v$. That is

$$E \approx \sqrt{E_v} \approx \sigma \sqrt{\frac{nn}{N}}$$

The relationship between the model error and the variance error shown in Eq. (17) underscores the importance of the number of data samples collected for the training of a model. This shows how the number of data sets $N$ and the quality as determined by $\sigma$ affects the magnitude of model error even for real situations where the bias error component may be present [103].

The relationship between model, bias and variance errors is shown in Fig. 13. It is evident from Fig. 13 that increasing the number of model parameters reduces the bias error while increasing the variance error. On the other hand, a decrease in the number of model parameters will lead to an increase in the bias error. It follows, therefore, that the conflicting nature of the bias and variance errors
errors makes it impossible to reduce (improve) one error type without worsening (increasing) the other. However, a conflicting multi-objective optimization could be carried out such that the optimum number of parameters (optimal model complexity) is a compromise between the bias and variance errors (the bias/variance trade-off). In order to achieve an optimal model complexity, it is important to perform a bias/variance trade off which could be achieved by employing explicit and/or implicit model complexity optimization methods. The explicit method (as explained earlier) involves the gradual increase in the model parameters and observing the performance in each case in terms of model error. On the other hand, the implicit method of optimization involves the use of regularization techniques that applied to change the model complexity without a change in the nominal number of parameters used.

According to Wu et al. [130] the process of selecting a particular type of network architecture in terms of the number of hidden layers and the number of neuronal nodes in each hidden layer is highly heuristic. Wu et al. [130] proposed that in order to ensure generalization of the neural network model for representing the air-flow rate through the spark-ignition engine under study, the network architecture should be selected based on the following guide: network selection starting by considering the simplest networks; and that it is important to observe the change in the training error between the actual network outputs and the target outputs as the number of hidden layers and neurons are increased (explicit method of model optimization); the fact that the best neural network should be selected based on the type of architecture that has the smallest network size (Occam's razor) and predicts with a minimum of error; and finally check and validate the model for its generalization ability. Combining an appropriate training algorithm, like the Levenberg-Marquardt algorithm [98,87,123], and Bayesian regularization has the effect of minimizing overfitting and improving generalization ability [130,38,82,42]. In a study conducted by Shamekhi and Shamekhi [118], it was noted that the prediction ability of a neural network was adversely affected by an increase in the complexity of the system being modeled and that the prediction error could be minimized if the complex system was divided into a number of sub-systems. This way, a neural network could be used for modeling each sub-system of the complex non-linear model.

5. Conclusions and outlook

Artificial neural networks have gained attention for a spectrum of applications including spark-ignition engine calibration. This is because trained artificial neural networks are able to satisfy a majority of the modeling requirements for engine calibration. The modeling requirements include the ability to deal with the curse of high-input dimensionality; the use of DoE for obtaining few measurements as practicable, with the aim of reducing engine calibration time and resource outlays; flexibility that allows model parameters to be optimized to avoid overfitting; and the facilitation of automated online optimization during the engine calibration process that eliminates the need for user intervention. In the area of spark-ignition engine calibration, these networks have been applied in the areas of system identification for rapid prototyping, use of neural networks as look-up table surrogates, emerging control strategies and OBD applications. While multi-layer feedforward networks could be used for system identification, single-layer recurrent neural networks with an increased number of inputs could be used for the identification and characterization of a dynamic system. The use of recurrent neural networks makes the execution of the neural network faster and relatively easy to implement. Even though there was a variation in the number of inputs to the various neural networks for the various applications, all the inputs were within the range (5–10 inputs) considered appropriate for engine calibration tasks. To this end, it is worth mentioning that the process of selecting a suitably trained neural network, in terms of architecture, still remains highly heuristic and requires a great deal of expertise in the choice of a neural network for a particular application. However, explicit and/or implicit methods could be employed for the selection of an optimal neural network model for a particular application. The Gaussian process modeling holds a promising future for modeling for the purpose of engine calibration because it also satisfies almost all the aforementioned calibration requirements. However, the decision to use a particular type of modeling technique for engine calibration depends on the particular nature of the problem at hand. In the event that it is difficult to establish a Gaussian Process model because of a large number of measurements, alternative non-linear modeling methods like artificial neural networks, relevance vector machines (RVM) or support vector machines (SVM) could be used. Furthermore, while it is prudent to use a polynomial regression model for problems with a low degree of complexity, it is important to use a local linear model for solving problems that require human intervention for a large number of measured data.

Acknowledgments

The authors would like to express their profound gratitude to the Hubei Key Laboratory of Advanced Technology for Automotive Components, Wuhan University of Technology, PR China, for providing financial support for the study. This work was also supported by the Chinese Scholarship Council (CSC) Grant numbers 2013GXZ993 and 2014GF032. M.K.A. Ali also appreciates financial support from the Egyptian Government.

References
