IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 27, 2021, accepted April 28, 2021, date of publication May 4, 2021, date of current version May 14, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3077275

Optimized Deep Stacked Long Short-Term Memory Network for Long-Term Load Forecasting

TAMER AHMED FARRAG[1] AND EHAB E. ELATTAR[2], (Senior Member, IEEE)
[1]Department of Computer Engineering, MISR Higher Institute for Engineering and Technology, Mansoura 35511, Egypt
[2]Department of Electrical Engineering, College of Engineering, Taif University, Taif 21944, Saudi Arabia

Corresponding author: Tamer Ahmed Farrag (tfarrag2000@hotmail.com)

This work was supported by Taif University Researchers Supporting Project through Taif University, Taif, Saudi Arabia, under Grant TURSP-2020/86.

ABSTRACT Long-term load forecasting (LTLF) is an essential process for strategical planning of the future needed extension in the power systems of any country. Besides, deep learning has become the heart of the machine learning paradigm, which is wildly used nowadays in many fields, and it also has become the current revolution in Artificial Intelligence (AI). In this paper, an optimized deep learning model based on Stacked Long Short-Term Memory Network (SLSTMN) is proposed. The architecture of the model is optimized to get the best configuration using Genetic Algorithm (GA). In addition, the hyperparameters of the model network are optimized using many deep learning techniques. During the optimization process, hundreds of model configurations are tested. The accuracy of this model is compared with many deep learning models and is compared against the related work in the field of LTLF. The dataset of the South Australia State (SA) power system is used to test the compared models. This data includes maximum daily load, daily maximum temperature, daily minimum temperature, weekday, the month, and holidays for 12 years from 2005 to 2016. SLSTMN achieves excellent accuracy and the lowest error value (almost 1%) when compared with other models on the same dataset and with related work models on different datasets.

INDEX TERMS Artificial intelligent, machine learning, deep learning, load forecasting, staked long short-term memory network, recurrent neural network.

NOMENCLATURE

h_t	Output vector
U	Matrix of the recurrent connections' weights
W	Matrix of the input connections' weights
b	Bias vector
x_t	input vector to the LSTM uni
f_t	forget gate component vector
i_t	update gate component vector
c_t	Unit input activation vector
o_t	Unit output activation vecto
h_t	Output vector of the LSMT unit
A_i	the outputs actual value
P_i	The forecasted output value
N	The test dataset siz
X'	normalized value
X	the value to be normalized
$\bar{A}_i$	the average of outputs actual value

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

I. INTRODUCTION

Building a successful electrical load forecasting model has been one of the hottest research areas for many years. It is crucial for any daily electrical network operator and for the strategical planer of the national network of any country to get a correct prediction about the behavior and the changes of network loads in the short, mid, and long term [1].

Electrical load forecasting has a significant influence on the operation of the electric power system, beginning from generation planning, power flow studies, unit commitment, and the economical operation of the electric power system. The necessity of electrical power changes every day which makes the forecasting of electrical load is very difficult. The electrical load is complex and affected by various exogenous

factors such as weather, social activities, seasonal factors and previous patterns [R1].

Load forecasting can be divided into three categories according to the targeted forecasting period: 1) short-term: day range 2) mid-term: week to few months range 3) long-term: year to several years range [2], [3]. This research focuses on long-term load forecasting (LTLF). The proposed forecasting model is based on deep learning techniques to get accurate load forecasting results.

The electrical load forecasting methods can be divided into two main methods. They are traditional methods (statistical techniques) and artificial intelligence (AI)-based methods. The former methods such as time series methods, linear regression, and autoregressive integrated moving average (ARIMA) have some weaknesses. If there are unexpected variations in environmental parameters, and daily patterns, a large forecast error will appear. Also, the nonlinear characteristics of complex electrical load cannot be characterized precisely using the traditional methods [4].

Artificial intelligence (AI) is one of the computer science branches that deal with simulating natural (human) intelligence by computers [5]. From the 1950s, AI starts to be one of the hottest researching topics. Symbolic logic, expert systems, and computer vision, natural language processing are examples of AI types that are developed. Starting from the 1980s, the concept of machine learning (ML) appears. The key difference between ML and other AI types is that ML enables computers to act without being explicitly programmed. Artificial neural networks (ANN) were an essential approach to apply ML. The idea behind ANN is to simulate the neural system in humans. Limiting of data and computational power stood in the way of the development of neural networks until the great leap occurred in the last ten years by appearing of cloud computing [6], big data [7], and revolutionary development in GPUs and computing chips manufacturing [8]. Deep learning (DL) techniques appear to benefit from this leap by helps the researchers to propose and to try deep and very deep ANN learning models [9]. In the past five years, many open source DL frameworks have become available such as TensorFlow [10], Keras [9], [11], and PyTorch [12].

DL is one of the main pillars of the current revolution of AI, many researchers using its techniques to build models for load forecasting. Most of the researches in the field of load forecasting target for short- and mid-term load forecasting [13]–[24], this because the long-term long forecasting (LTLF) needs to take many factors into account and to manipulate the massive amount of historical data which were collected during many past years.

According to *Andrew Ng*, one of the DL leaders: "I believe Deep Learning is our best shot at progress towards real AI." Applying DL to any problem faces many challenges. In addition to the overfitting problem discussed in the previous section, DL is a data-intensive method, so there is a need to own a massive amount of data to apply DL. Training a DL model also required high computational power which is available nowadays by using GPUs and cloud computing. DL models, like any NN model, are a black-box[1] model that usually is not understood how they get their output. DL model is highly specialized to solve a specific problem, and it cannot be used to solve another problem in a different domain without retraining the model.

The AI methods for LTLF include ANN, Fuzzy and neuro-fuzzy, SVM,..etc [25]–[27]. In [2], [3], the authors introduce an excellent review of the different techniques that were developed for long-term load forecasting. In this paper, the proposed model is a deep neural network (DNN) model. So, in this section, some of the researches in the field of LTLF are highlighted, which is based on AI models (especially ANN-based models).

In [28], the authors use 10 factors as inputs for their proposed ANNs which are: (1) gross national product, (2) gross domestic product, (3) population, (4) number of households, (5) amount of CO_2 pollution, (6) number of air-conditioners, (7) index of industrial production, (8) oil price, (9) energy consumption, and (10) electricity price. The target was finding the peak daily load in 2010 and 2020 in Japan.

The authors tested ten various feed-forward ANN-based models in [22] to forecast Greek long-term energy consumption. The results showed that ANN model predictions are much more accurate than the linear regression model and similar to those obtained by the support vector machine model. Also, the authors in [29] presented the hybrid model of ANN and biogeography-based optimization technique for long-term forecasting of India's sector-wise electrical energy demand. The using of this optimization technique to train the ANN improves forecasting accuracy. Besides, a fuzzy logic model was proposed in [24] for long-term forecasting. A reliably forecasted result is obtained, and MAPE equal to 6.9%. The model training period is not mentioned clearly in the paper, and the fuzzy logic rules appear to be very trivial and based only on the humidity and temperature.

A three stages model was proposed in [25]. The time series decomposed to three components by X12- ARIMA algorithm; then, the model used three NAR neural networks for the forecasting process. The output of these networks was combined using a feed-forward neural network (FFNN). The proposed model had conclusive results compared to several models. Neural network-based approach for LTLF of the Jordanian power system from 2015 to 2029 was presented in [26]. Two types of FFNN are examined, namely, the back-propagation and the radial basis function neural networks.

In [30], two ANN-based models have been applied to forecast the long-term electricity demands of Turkey. But there is a doubt that these models achieve the same success if they are used to forecast the daily electricity consumption for a long-term period. The design of the models is very simple to solve such a complicated problem. In [31], a collaborative fuzzy-neural model is proposed. This model was used to predict annual energy consumption in Taiwan. The disadvantage of

[1] https://www.sciencedirect.com/topics/engineering/black-box-model

this model is that it is not a fully automated model. The forecasting performance was improved through the site experts' collaboration.

The generalized regression neural network (GRNN) is employed in [32] to solve the problem of LTLF. The GRNN has some advantages over conventional ANN where it has the ability to estimate the absolute function between input and output data sets directly from training data. A hybrid data mining driven algorithm for long-term peak load forecasting is proposed in [R4] where the particle swarm optimization is employed to optimize the parameters of support vector regression. The hybrid model is then applied to solve the forecasting problem in a real-life grid. The author in [R5] proposed a feature-fusion-kernel-based Gaussian process model for probabilistic long-term load forecasting where the probabilistic distribution of the forecasting is specifically revised to overpass further the mismatch between the forecast and the original data set.

Table 1 presents a comparison between LTLF related works. The key problem with the mentioned related works is that LTLF models are designed to forecast the annual electricity consumption or annual peak load as one total number. On the contrary, our proposed model is designed to forecast the daily peak load for years in the future, which is important to study the changes in load during the year. Also, It can be concluded that no standard data set was used in the field of LTLF. The comparison between the different models in this context is difficult, and the only standard evaluation metric is the mean absolute percentage error (MAPE).

The majority of proposed forecasting methods in the literature are developed for short-term load forecasting. Although few approaches have been proposed and utilized for LTLF in literature, none of them consistently provides precise forecasts. In addition, proposing a precise and practical model of LTLF is still a very significant research issue because of the importance of LTFT for strategic planning of the future needed extension in the power systems of any country, the extreme complexity of electrical load data, the power system requirements, the precise power quality conditions, and deregulation. This motivates us to propose an accurate and practical forecasting model to obtain a high forecasting accuracy for LTLF. So, in this paper, a model based on the Stacked Long Short-Term Memory network (SLSTMN) is proposed to solve the LTLF problem. The proposed model is tested and applied to forecast the electrical load of the South Australia state (SA) for a long-term period.

The key contributions of this paper are:

- Proposing a hybrid model based on the Stacked Long Short-Term Memory network (SLSTMN) to solve the LTLF problem where its architecture is optimized to get the best configuration using GA. Besides, the hyperparameters of the model network are optimized using many deep learning techniques.
- The LTLF models in the literature predict annual peak load or annual energy consumption as one total number. One of this paper's contributions is to overcome this research gap by employing the proposed model to predict the daily peak load.
- Formulate the problem of finding the best network architecture as a single-objective multi-parameters discrete optimization problem and use the genetic algorithm (GA) to solve it.
- Adam optimizer is used to optimize the process of model parameters training. A Dropout and early stopping techniques are applied to prevent overfitting training.
- Examine tens of deep learning models with many hyperparameters tuning and compare their results against the proposed model to indicate the greater accuracy of the proposed model.
- Compare the proposed model results with two classical forecasting approaches, ARIMA and SVR based on real-life data to prove the ability of the proposed model to give accurate results of LTLF in real life.
- Models' configurations, hyperparameters hypotheses, and results are recorded in a database for in-depth analyses.

TABLE 1. Overview of LTLF related work.

Ref.	Models type	Data set source	Training period	Forecasting period
[28]	1. ANN 2. RNN	Japan	1975 - 1995	Annual peak load 2010 & 2020
[33]	ANN	Greek	1992– 2004	Annual energy consumption 2005-2015
[29]	ANN + biogeography optimizer	India	1980– 2012	Annual energy consumption 2013-2025
[34]	Fuzzy logic	Mubi, Nigeria	Not Mentioned	Monthly peak load 2013 - 2014
[35]	ARIMA + NAR ANN	Algeria (Sonelgaz)	2000-2010	Monthly peak load 2011 - 2012
[36]	ANN (BPNN, BFNN)	Jordon	2000 - 2014	Annual peak load 2015 - 2029
[30]	ANN	Turkey	1981 - 2007	Annual energy consumption 2008 - 2014
[31]	neuro-fuzzy	Taiwan	1945 - 1976	Annual energy consumption 1977 - 2008
[R1]	GRNN		2000 - 2008	Annual peak load 2009-2019
[R4]	Hybrid method	Iran national grid	1991-2016	Yearly peak load 2017 - 2026
[R5]	Feature-fusion kernel-based Gaussian process	Global Energy Forecasting Competition data set.	five years of historical electric load	one-month-ahead probabilistic forecast

Next, in this paper, Section II presents an overview of the main deep learning architectures and techniques used in the proposed model. Section III discusses the proposed model. Section IV discusses how to use the Genetic algorithm to get the best network architecture configuration of the proposed model. Section 0 shows the experiments used to evaluate the performance of the proposed model. The paper is concluded in section VI.

II. DEEP LEARNING ARCHITECTURES AND TECHNIQUES

As stated before, DL is a new type of ML methods which can be considered as the heart of the AI big picture. DL is

based on the classical idea of neural networks but with many changes in methods, architectures, and scales. In recent years, deep learning contributes to providing the best solutions to many problems in image recognition, speech recognition, and natural language processing (NLP). This section focuses on the layers, and techniques that will be found in the proposed model. Also, challenges in DL context are summarized at the end of this section:

A. RECURRENT NEURAL NETWORKS (RNN) LAYERS

The recurrent neural network (RNN) achieved impressive results in sequence generation problems such as NLP. In RNN, there are feedbacks in some manner of neurons outputs to their inputs or the previous layers. This feedback aims to add the memory concept to the neural network. In other words, the training process not only depends on the current inputs that are fed to the input layer but also depends on the previous values of inputs. In the RNN context, a past state on the RNN unit, called the hidden state (h), is calculated according to its type, but overall, it depends on the historical outputs and is used to get the next output. For example, figure 1 shows the different state for the basic RNN unit, which is called vanilla RNN unit, the following equations calculate the final output, where U, W, V are equation weights:

$$h_t = \tanh\left(Wx_t + Uh_{t-1} + b_a\right) \quad (1)$$

$$y_t = \text{softmax}\left(Va_t + b_y\right) \quad (2)$$

Vanilla RNN suffers from many drawbacks such as vanishing gradients, so researchers have developed more sophisticated types of RNNs to deal with these drawbacks. Examples of these types are Long short-term memory (LSTM) [37], Gated recurrent units (GRUs) [38], and Bidirectional recurrent unit [39]. Next, a short overview of LSTM is presented because it is used in our proposed network and is considered the most important type of RNNs.

Long short-term memory: LSTM was designed to overcome two main shortages in vanilla RNN. The first one is long-term dependency in RNNs, which is not recommended for many applications. In simple words, the current state of the RNN cell is affected by the previous states even if the timestamp of previous states is very far from the current timestamp. This is not logical in many applications such as auto-translations and speech recognition. The second one is the vanishing gradient and exploding gradient. Learning methods in RNN depend on changing the weights by receiving an update proportional to the partial derivative of the loss (error) function with respect to the current weight in each iteration of training. The problem is this gradient will be vanishingly small, effectively preventing the weight from changing its value and as a result, the learning process is stopped. LSTM overcomes these problems by introducing the gates concept. In LSTM, there are three types of gates: forget gate, input gate, and output gate. The forget gate uses the sigmoid function applied to the current input and the hidden state to decide the contribution of each of them in the next

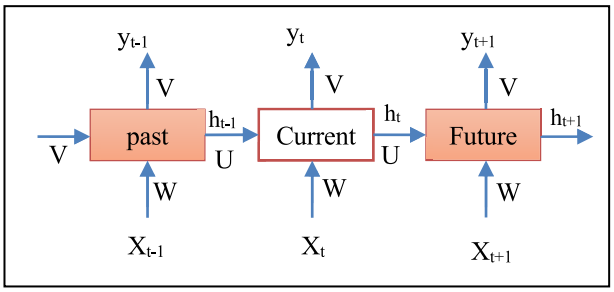

FIGURE 1. States of vanilla RNN unit in work.

state of the LSTM cell as shown in equation (3). So, it plays a key role in solving the Long-term dependency problem. The input gate uses a combination of used sigmoid and tach functions applied to the input and hidden state to compute the current state of the cell, as shown in equations (4,5). The output of the previous two gates is used by the output gate to calculate the cell's next hidden state as shown in equations (6,7).

Figure 2 shows the dataflow and relation between different gates entire the LSTM cell. As noticed, LSTM cell has many parameters and need much more calculations compared to ordinary artificial neuron.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (3)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (4)$$

$$c_t = f_t c_{t-1} + i_t tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (5)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_0) \quad (6)$$

$$h_t = o_t tanh(c_t) \quad (7)$$

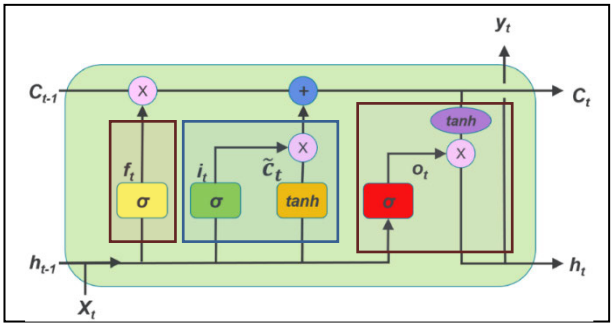

FIGURE 2. The long short-term memory (LSTM) cell [40].

B. REGULARIZATION

One of the traditional problems in any neural network is the overfitting problem. Overfitting is the problem of the dissimilarity of the performance of the model, where it performs very well with the training data, and its performance decreases or fails dramatically in case of using test data [41]. This problem increases proportionally as the network increases, such as in the case of deep learning. There are many techniques and recommendations are used to overcome this problem; regularization techniques are in the heart of them. L2 and L1,

Dropout [42], Data augmentation [43], Early stopping [44] are examples of regularization techniques. The proposed model use Dropout and early stopping techniques to improve its results.

The idea behind Dropout is that randomization is one of the success secrets of neural networks, but after a large number of training iterations for deep and large neural network, the network is settled which lead to the overfitting problem. So, in dropout, a percentage of neurons are canceled randomly in every iteration. This percentage is a hyperparameter that should be tuned when using Dropout. This process of randomly removing neurons produces a network randomly structured and many studies prove the excellent results of this technique in solving the overfitting problem.

Early stopping tries to solve the overfitting problem with a differing point of view. The overfitting occurs when the model is trained more than it should on the training data, so it overfits on them. Early stopping monitors the performance of the model and stops the training process when it detects that the model starts to overfit and its performance enhancement is almost stopped.

III. THE STACKED LONG SHORT-TERM MEMORY NETWORK

Using RNN in many time series problems was not achieving the best results [14], which disappoints many researchers of using RNN in this type of problem. In this paper, multiple layers of RNN are combined to make the network deeper. The idea behind this architecture was inspired by [45] in which the authors applied phoneme recognition on the TIMIT database [46] and got excellent results compares to other techniques for the same dataset. According to [47], as more RNN networks being deeper by stacking multiple recurrent hidden states on top of each other as it potentially allows the hidden state at each layer to work at different timescales.

A model based on the Stacked Long Short-Term Memory Network (SLSTMN) is proposed which is tested as will be shown next and its performance compared with many other models. As shown in figure 3, this model consists of a number of stacked LSTM layers followed by a number of dense layers. As a state-of-art issue, SLSTMN uses two techniques for regularization: dropout and early stop. This model is optimized using Adam optimizer [48] to update network weights during the training process. This optimizer has achieved outstanding results compared to other stochastic optimizers, as will be shown later in this paper. In [49], the author introduces a gentle overview of this optimizer, and he presents the idea behind this optimizer and its mathematical background. SLSTMN has many hyperparameters that need to be tuned. Next, in this paper, the way of tuning these parameters to get the best configuration setting for SLSTMN is discussed.

The data needs to be normalized before starting the training process to improve the efficiency of the forecasting method and to avoid the occurrence of overflow during the calculation process. Equation (8) shows how to perform min-max

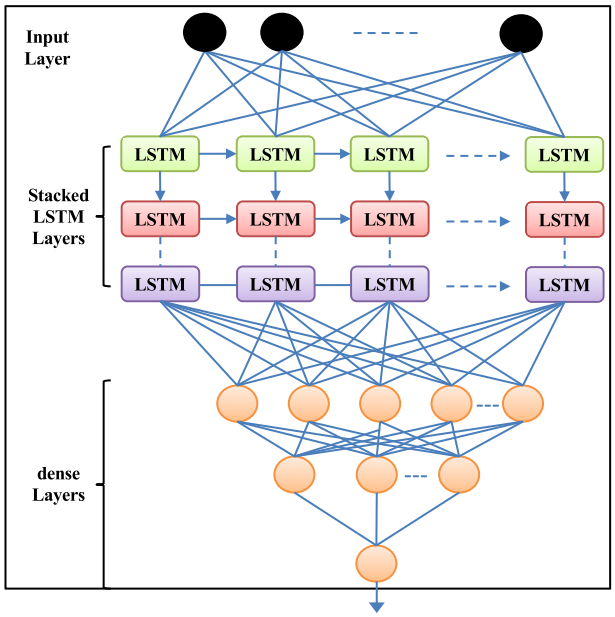

FIGURE 3. Stacked long short-term memory network (SLSTMN).

normalization between [−1,1] for any data column X, where X is the original value and X' is the normalized value.

$$X' = -1 + \frac{2 * (X - mix\,(X))}{max\,(X) - min\,(X)} \tag{8}$$

Figure 4 presents an overview of the training and evaluation process of the proposed model. This process consists of three main stages:

1) Data Preprocessing: the dataset which is used for LTLF is loaded. Then, it normalized, as mentioned before. After that, the normalized data re-framed as supervised learning problems. The final step is to split the available data into two sets: the training set and the testing set.
2) Model fitting and training: the model is created and trained for several epochs.
3) Model Evaluation: use the trained model to predict the output using the test set as input and compare the predicted values with the actual values to evaluate the accuracy of the model.

IV. PROPOSED MODEL ARCHITECTURE OPTIMIZATION

In figure 3, the architecture of SLSTMN is shown. The number of staked LSTM layers and many other parameters should be determined. The architecture parameters are formulated as a single-objective multi-parameters discrete optimization problem. Then, the genetic algorithm (GA) is used to solve this problem. GA is a global search procedure that searches from one population of points to another. It belongs to a class of probabilistic methods called ‘‘evolutionary algorithms’’ based on the principle’s selection and mutation.

In this case, the GA optimizer aims to get the minimum value of mean absolute percentage error (MAPE) for the evaluated model. Table 2 shows the value space of the model

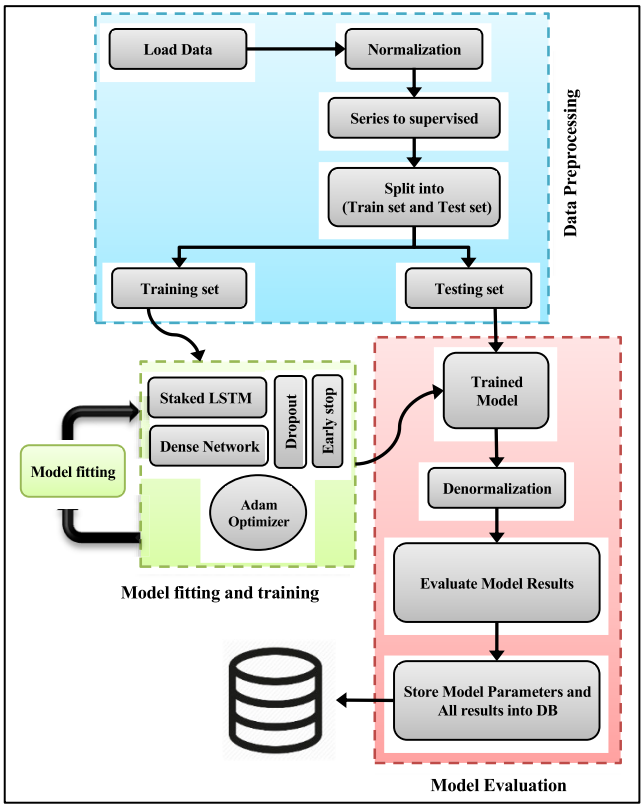

FIGURE 4. SLSTMN model training and evaluation process.

TABLE 2. Model parameters.

parameters	Description	Value space
SL	Number of the stacked LSTM layers	[1 – 10]
SN	Number of neurons in each LSTM layer	[8, 32, 64, 256, 512]
D1-D2	Number of neurons in each dense layer	[0,8, 32, 64, 256, 512]
Dout	The value of the dropout in each layer	[0, 0.2 ,0.5 ,0.8]
opt	The optimization algorithm that is used to tunning the model hypermarkets during the training process	Adam, SGD, Adadelta, RMSprop, Adagrad, and Nadam
batch	The batch size	[32, 64, 128, 256, 512, 1024]

parameters. All the parameters have discrete values. Each parameter is considered as a gene in the GA chromosome as presented in figure 5. Table 3 summarizes all the data that are needed to initialize the GA optimizer to find the best solution. To implement this optimizer, a modified version of PyGAD[2] v2.5.0 is used. This package is designed to solve continuous optimization problems using GA. This package is modified to solve discrete optimization problems.

Algorithm 1 summarizes the main steps to the SLSTMN model architecture optimization process which can be used to optimize any other model architecture by some changes.

[2]https://github.com/ahmedfgad/GeneticAlgorithmPython

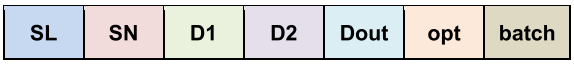

FIGURE 5. GA chromosome.

TABLE 3. GA data.

GA parameters	value
Number of genes	7
fitness function	$MAPE(SL, SN, D1, D2, Dout, opt, batch)$
objective function	$min(MAPE(SL, SN, D1, D2, Dout, opt, batch)$
Number of generations	100
Number of solutions in the population	50
Select parents for crossover	steady-state selection technique
Crossover	Single point
Mutation	10% Random mutation (within the values of the predefined value space)

It's worth saying that GA is not used to train the internal hyperparameters such as weights and biases of the model. This role is performed by one of modern gradient descent optimization algorithms such as the Adam algorithm.

V. PERFORMANCE EVALUATION

A. DATASET DESCRIPTION AND FORECASTING SCENARIO

Load forecasting is a time series problem that was reformulated to be a supervised learning problem.[3] The problem is starts with training the model by the training set and evaluate the model performance using the test set. In DL, the dataset size plays the main role. As much as training data is large, the model performance is enhanced, and many learning problems are avoided. So, it is a must to train the model using the big dataset to benefit from the DL paradigm. Australia is one of the leading counties in the field of data availability [50]. Therefore, a dataset from The Australian Energy Market Operator (AEMO) [51] is employed which is responsible for operating Australia's largest gas and electricity markets and power systems. On their website, AEMO provides the Net System Load profile (NSLP) for each of the distribution network areas such as Victoria, NSW, ACT, SA, QLD.

In this paper, the load data of South Australia State (SA) in 12 years from 2005 to 2016 is considered. The load data provided on a half-hour scale. For LTLF, the daily maximum load value is considered. In addition to load data, the weather data in SA from 2005 to 2016 is collected.

Figure 6 shows the daily load in SA in 2016 which represents a portion of the dataset. All the required weather data is available on the Bureau of Meteorology website [52]. The Bureau of Meteorology is Australia's national weather, climate, and water agency. The daily maximum and minimum temperature and solar exposure from the North Adelaide weather station are collected. Adelaide is the capital city

[3]https://machinelearningmastery.com/convert-time-series-supervised-learning-problem-python/

Algorithm 1: Model Architecture Optimization Using GA

Initialization:

1) set parameters:

N = 100 //number of generations

NS = 50 // Number of solutions in the population

NG = 7 // number of gens according to Table 2

Pm = 10% // mutation probability

Pc = 10% // crossover probability

2) set genes values ranges according to Table 2

3) IP = Generate initial population randomly

4) fitness function: MAPE(solution)

Output: ***Solution sol***

for i = 1 **to** N

PL=select(IP)

// get the new population after crossover and mutation

Offspring = Crossover(PL)

new_P=mutation(Offspring)

foreach sol **in** new_P:

// create SLSTMN using values in sol

Model = createModel(sol)

Final_model=train_test(Model)

// get MAPE for the test set as sol evolution

z=fitness_evalutaion(Final_model)

add z to new_P evalution list (EL)

end

best_sols = best values of EL

// update initial population using best solution founded

update (IP, best_sols)

end

// get the best solution from IP

sol = best_fitness_value (IP)

return sol

of the state of South Australia. Adelaide is home to more than 75% of the South Australian population. Also, a list of public holidays on SA in the study period is collected.

In this paper, the tested scenario is to forecast the maximum daily load of 2016 in SA based on the data available for the years from 2005 to 2015 (11 years). So, the training set represents the data of around (11 × 365 = 4015) days, the output of the models is the expected values of the daily load of 2016, which compared with the real values recorded in 2016 that represent the test set for our models.

B. EVALUATION METRICS

Two main evaluation metrics are considered to measure the forecasting process success: mean absolute percentage error (MAPE) and root-mean-square error (RMSE). The following equations define these values:

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|A_i - P_i|}{A_i} * 100 \quad (9)$$

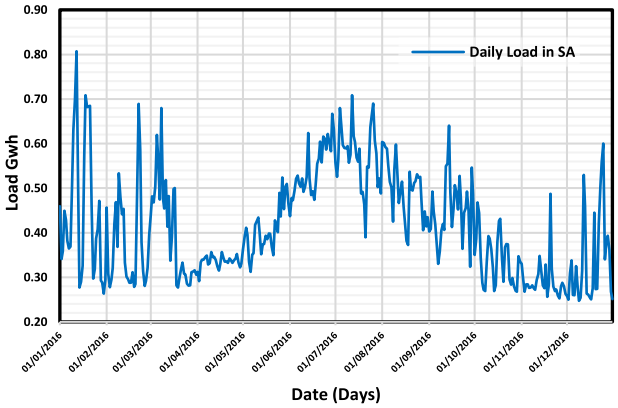

FIGURE 6. Daily load of SA in 2016.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(A_i - P_i)^2}{N}} \quad (10)$$

where A_i, P_i and N is the actual value, the forecasted value, and the testing dataset size respectively? The LTLF problem is solved offline, so the execution time is not an evaluation metric, unlike the case of short-term load forecasting. The most important issue of the power system operator is to get the most accurate forecasting results.

C. EVALUATED MODELS

In this research, tens of models are examined with many hyperparameters hypotheses. Besides, six forecasting models are compared. The first two models are non-deep learning models. The other four models are DL models. Next, the architecture of each model will be presented. The last model is noted as *(Model 4)* based on the proposed *SLSTMN* architecture. The architecture of each DL model is optimized using GA to get the best configuration for each model. The input features which are tested include maximum daily load, daily maximum temperature, daily minimum temperature, the weekday, the month, and holidays. The different input feature collections are summarized in table 4.

TABLE 4. Input features collections.

Collection	Input features
C1	daily maximum load.
C2	daily maximum load, daily maximum temperature, daily minimum temperature, the weekday, the month, and holidays.
C3	daily maximum load, daily maximum temperature.
C4	daily maximum load, the weekday, the month, and holidays.
C5	daily maximum load, holidays.

1) ARIMA MODEL

One of the classical time series forecasting models is the Autoregressive integrated moving average (ARIMA). Two variations of ARIMA will be examined: seasonal ARIMA (SARIMA) and non-seasonal ARIMA. The seasonality value in SARIMA is assigned to 30 (one month).

More details about ARIMA and SARIMA can be found in [53]–[56].

2) SVR MODEL

This model is a support vector regression (SVR) model [57], [58]. SVR is a widely used approach for power load forecasting [17], [20], [59]. It is vital to choose the best SVR kernel and the values of its parameters [60]. There are many choices for SVR kernel such as (linear, poly, RBF,..). As stated in [60], "The RBF kernel is a reasonable first choice." So, RBF is chosen as the SVR kernel. There are two main parameters in the RBF kernel (γ and C). By using the cross-validation and grid-search methods, the best values for γ and C can be obtained.

3) MODEL 1 (ANN)

As shown in figure 7, model 1 consists of a number of dense layers after each layer except the last one a Dropout layer is added. The number of neurons in the last (output) dense layers is fixed as one. Similar to the optimization process presented in section IV. GA is used to optimize the following hyperparameters: the number of dense layers, the number of neurons in each dense layer, the dropout value, batch size, the number of lagged days. The number of trainable parameters varies and reaches 20,501 in some trials.

4) MODEL 2 (CNN + ANN)

As shown in figure 8, model 3 consists of one convolutional layer followed by two dense layers after each layer except the last one, a Dropout layer is added. The convolutional layer has 64 filters, and the kernel size equals 2. The number of neurons in the last(output) dense layers is fixed as one. The hyperparameters that are optimized using GA are filter and kernel size in CNN layer, the number of dense layers, the number of neurons in each dense layer, the dropout value, batch size, the number of lagged days. The number of trainable parameters varies and reaches 89,993 in some trials.

5) MODEL 3 (GRU RNN +ANN)

As shown in figure 9, model 3 consists of a number of GRU layers followed by a number of dense layers after each layer except the last one a Dropout layer is added. The hyperparameters hypotheses that are tested in this model are: The hyperparameters that are optimized using GA are: the number of GRU layers, the number of dense layers, the number of neurons in each dense layer, the dropout value, batch size, the number of lagged days. The number of trainable parameters varies and reaches 122,001 in some trials.

6) MODEL 4(SLSTMN)

This model based on the proposed *SLSTMN* architecture that is shown in figure 3. The model architecture is optimized as mentioned in section IV. The number of trainable parameters varies and reaches 210,177 in some trials.

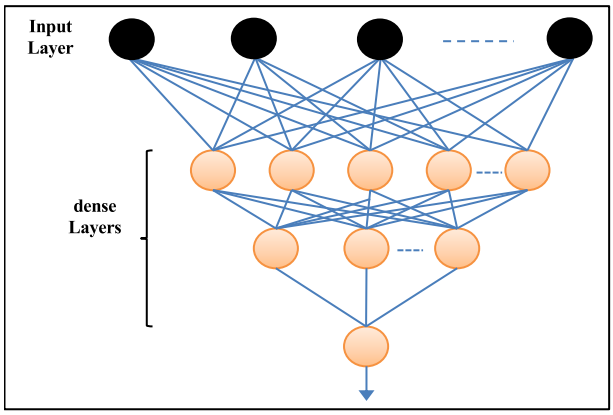

FIGURE 7. Model 1 architecture.

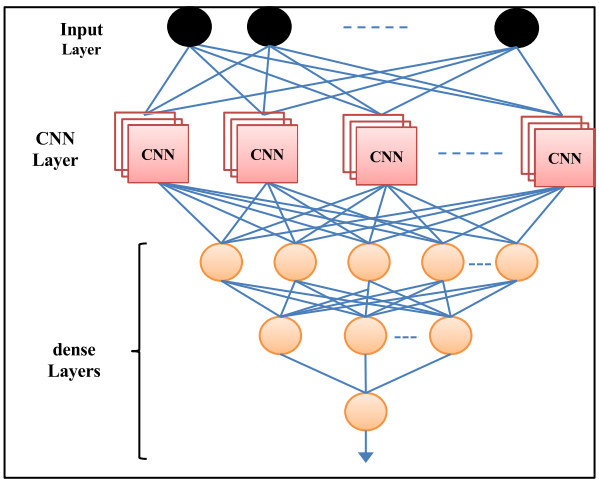

FIGURE 8. Model 2 architecture.

D. SIMULATION AND ANALYSIS OF MODELS FORECASTING RESULTS

The ARIMA model[4] and SVR Model[5] are implemented by using the Azure Machine Learning service (Azure ML). Azure ML is a cloud-based predictive data analysis provider [61]. Azure ML provides many predestined data analysis models, data processing modules, and many other modules that may be needed to create an ML model. These models and modules are integrated into Azure ML Studio [62].

The deep learning models in this research are implemented using python Anaconda [63] and Keras API [11] with Google's TensorFlow [10] backend engine and are tested on a machine that has the following specifications: CPU: Core i5 Processor 2.5 GHz /16GB RAM /500GB SSD, GPU: NVIDIA GeForce GTX1050 4GB, compute capability 6.1.

[4]The Azure ML implementation of the ARIMA model using R script can be accessed by the link: https://gallery.cortanaintelligence.com/Experiment/Classical-Time-Series-Forecasting-LTLF

[5]The Azure ML implementation of the SVR model using python script can be accessed by the link: https://gallery.cortanaintelligence.com/Experiment/LTLF-SA-Data-SVM. The best values of (γ and C) parameters are (0.015625, 0.125).

TABLE 5. Top 15 experimental results (out of more than 1126 experiments).

Exp. ID	n_days	features	epochs	batch	neurons	Dropout	RMSE (MW)	MAPE (%)	Model No.	Training time (ms)
20200819040116	1	1	16	256	50	0.8	10.296	1.044	4	137
20200819035634	1	1	19	256	50	0.5	9.9241	1.048	4	139
20200819035205	1	1	16	256	50	0.2	10.1579	1.0713	4	137
20200819040253	1	1	17	256	100	0.8	10.5965	1.1302	4	136
20200819035807	1	1	15	256	100	0.5	10.7724	1.3959	4	135
20200819035332	1	1	18	256	100	0.2	11.4245	1.4646	4	138
20200819035038	1	1	21	256	10	0.2	10.6281	1.5333	4	140
20200819041447	2	1	43	256	50	0.5	12.2568	1.5544	4	144
20200821125215	1	1	21	256	50	0.2	11.1351	1.6489	2	15
20200819235910	1	1	14	512	10	0.5	11.0724	1.6717	1	60
20200819042224	2	1	79	256	50	0.8	13.4732	1.693	3	220
20200819205921	2	1	14	256	100	0.8	13.0226	1.7467	1	61
20200819205550	1	1	14	256	100	0.8	11.0169	1.7672	1	60
20200820000411	2	1	80	512	10	0.2	14.1632	1.7906	1	102
20200819205811	2	1	15	256	100	0.5	13.5765	1.8265	1	65

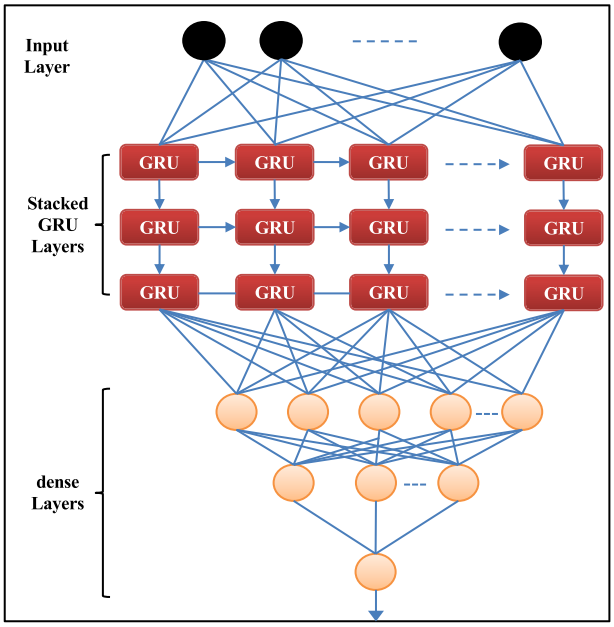

FIGURE 9. Model 3 architecture.

Configuring TensorFlow to use the GPU processor decreases the execution time dramatically, which helps us to perform hundreds of trials to test many models and tuning their hyperparameters. Next, our trials will be summarized. For each model, the input features collections and the hyperparameters hypotheses are tested and the results for each trail (experiment) are stored in a database for analysis purposes. The following data is available for each experiment: the values of hyperparameters, no. of training epochs, RMSE, MAPE, optimizer type, the predicted values, the actual values, and training loss history after each epoch. This data is used to study the training process of each model and enhance it by using many techniques for regulation, optimization to avoid the overfitting problem.

Table 5 shows the best 15 results of the deep learning models, the best model, and the best hyperparameters values. The proposed model (SLSTMN) produces the best results in general. Table 6 compares the best result of each tested model with respect to RMSE and MAPE. The results show the advance of SLSTMN where the RMSE equals 10.296 MW, and MAPE equals 1.044 % which represents an excellent result in the field of LTLF.

TABLE 6. Comparison of the best result for each of the tested models.

Model	Exp. ID	n_features	n_days	RMSE (MW)	MAPE (%)
ARIMA Model	Azure ML	1	1	135.39	33.28
SVR Model	Azure ML	1	1	66.90186	10.91
Model 1	20200819235910	1	1	11.0724	1.6717
Model 2	20200821125215	1	1	11.1351	1.6489
Model 3	20200819042224	2	1	13.4732	1.693
Model 4 (SLSTMN)	**20200819040116**	**1**	**1**	**10.296**	**1.044**

Table 7 shows the comparison between the tested models and the other published models used for LTLF. The data that was used in these models was different, so, the value of MAPE is considered as the standard metric of success of any forecasting model. This comparison shows the superiority of the SLSTMN model over other models.

For more details about the best model, figure 10 presents training set loss versus test set loss during the learning process of model 4 (SLSTMN). This model needs only 16 epochs to reach the best loss and as shown the proposed model avoids the overfitting problem by achieving close results when applying to the training set as well as the test set. The small number of epochs needed for model learning is an excellent remark especially with very complex architectures such as SLSTMN to avoid an extremely slow learning process. Figure 11 shows the close predictions of the maximum daily load in 2016 against the actual recoded values when using the proposed model.

TABLE 7. Comparison between SLSTMN and other published models.

Model	MAPE (%)
Best model in [28]	3
Best model in [33]	3.52
Best model in [29]	1.45
Best model in [34]	6.9
Best model in [35]	2.35
Best model in [36]	2.7
Best model in [31]	1.37
Model 6 (SLSTMN)	**1.044**

In the proposed model, the "Adam" optimizer is used for network training. The outstanding performance of this optimizer is verified, when it has been compared to the other modern gradient descent optimization algorithms such as (SGD, Adadelta, RMSprop, Adagrad, and Nadam). Figure 12 presents the comparison of the best MAPE value for each optimizer when applying to SLSTMN.

E. DISCUSSION

As clear from the above results, the achieved forecasting results of the proposed model which compared with different models prove the superiority of the proposed model over other models. Table 8 indicates the improvement of MAPE and RMSE of the proposed model over other models.

TABLE 8. Improvements of the proposed model over other models.

Model	Improvements (%)	
	MAPE	**RMSE**
Model 4 (SLSTMN)	---	---
ARIMA Model	96.86	92.40
SVR Model	90.43	84.61
Model 1	37.55	7.01
Model 2	36.69	7.54
Model 3	38.33	23.58

Table 8 prove that the proposed model gives an accurate and precise forecasting accuracy for the LTLF problem compared to other models. Besides, the proposed model reduced the MAPE significantly compared to other published models as shown in Table 7.

Fig. 10 indicates that the proposed model needs a small number of epochs for learning which avoids an extremely slow learning process. Also, the results of Fig. 11 show that the predicted maximum daily load obtained by the proposed model is very close to the actual maximum daily load values which prove the efficiency of the proposed model in predicting the maximum daily load in contrast with other models in the literature which predict annual peak load as one total number.

The superiority of the proposed model comes from obtaining the optimal configuration of the SLSTMN using GA and employing the "Adam" optimizer for network training which gives superior performance compared to other modern gradient descent optimization algorithms as shown in Fig. 12.

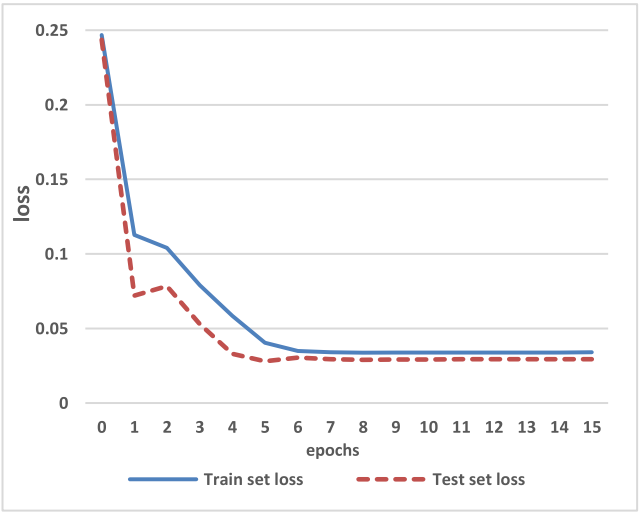

FIGURE 10. Training set loss vs Test set loss during the learning process (Exp. ID 20200819040116).

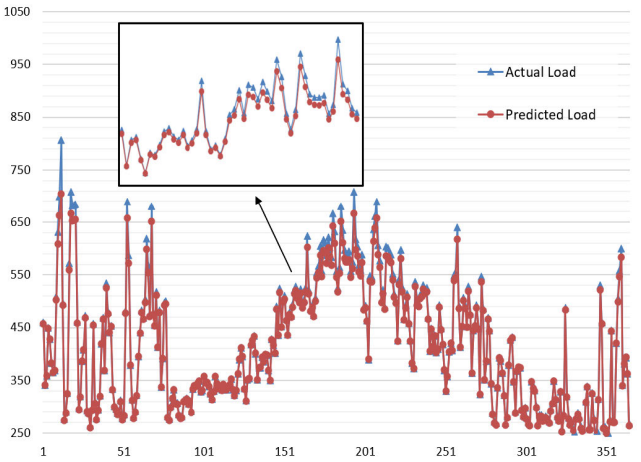

FIGURE 11. Actual vs. predicted values of maximum daily load in 2016 (Exp. ID 20200819040116).

The training time of the proposed model for different experiments is shown in Table 5. As shown the training time is in milliseconds which proves that the proposed model can be trained in a very short time. These results indicate that the proposed model is also efficient when computational time is concerned. However, the LTLF problem is solved offline, the proposed model can be trained and obtain the solution to the LTLF problem in a very short time.

To confirm the validity of the proposed model, coefficient of determination (R^2) is employed which can be defined as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (A_i - P_i)^2}{\sum_{i=1}^{N} \left(A_i - \bar{A}_i\right)^2} \tag{11}$$

The best possible value of R^2 is 1 and it can be negative. Table 9 shows the coefficient of determination (R^2) of the proposed model for different experiments. As shown in Table 9, the values of R2 of the proposed model for different experiments are very close to 1. This validation confirms

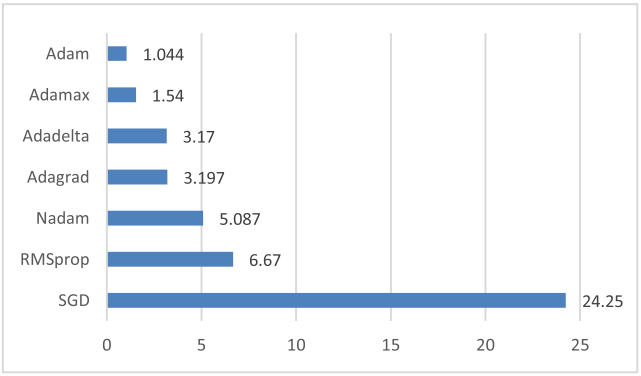

FIGURE 12. Best achieved MAPE (%) for each optimizer.

that the proposed model gets precise prediction results, which is clearly applicable to the LTLF problem.

TABLE 9. Coefficient of determination (R2) of the proposed model for different experiments.

Exp. ID	R^2
20200819040116	0.997032
20200819035634	0.996958
20200819035205	0.997876
20200819040253	0.997998
20200819035807	0.997533
20200819035332	0.997816
20200819035038	0.99694
20200819041447	0.992094
20200821125215	0.998338
20200819235910	0.997419
20200819042224	0.990571
20200819205921	0.992547
20200819205550	0.997784
20200820000411	0.98937
20200819205811	0.991407

VI. CONCLUSION

A model based on the Stacked Long Short-Term Memory network (SLSTMN) is proposed. Traditionally, LTLF models are designed to predict the annual peak load or annual energy consumption as one total number. In this paper, the proposed model is designed to fill this gap by forecasting the daily load.

The proposed model is tested and applied to forecast the electrical load of the South Australia state (SA). Its results are compared against non-deep learning models and tens of deep learning models with hundreds of hyperparameters tuning. GA is used to optimize the architecture of the proposed model network. SLSTMN records the lowest MAPE value (almost 1%). Therefore, SLSTMN provides more accurate predictions than the tested models. With this MAPE value, the proposed model performs better than the models used in the related work in the field of LTLF. All the trails and models' configurations, hyperparameters hypotheses, and results are recorded in a database for in-depth analyses.

The future work includes the applications of the proposed model for a chaotic time period (in the energy market) such as during the COVD-19 pandemic (2019-2020) then gives projections for years after 2021.

ACKNOWLEDGMENT

The authors would like to acknowledge the financial support received from Taif University Researchers Supporting Project Number (TURSP-2020/86), Taif University, Taif, Saudi Arabia.

REFERENCES

[1] H. M. Al-Hamadi and S. A. Soliman, "Long-term/mid-term electric load forecasting based on short-term correlation and annual growth," *Electr. Power Syst. Res.*, vol. 74, no. 3, pp. 353–361, Jun. 2005.
[2] S. R. Khuntia, J. L. Rueda, and Mart A. M. M. van der Meijden, "Forecasting the load of electrical power systems in mid-and long-term horizons: A review," *IET Gener., Transmiss. Distrib.*, vol. 10, no. 16, pp. 3971–3977, Dec. 2016.
[3] K. B. Lindberg, P. Seljom, H. Madsen, D. Fischer, and M. Korpás, "Long-term electricity load forecasting: Current and future trends," *Utilities Policy*, vol. 58, pp. 102–119, Jun. 2019.
[4] E. E. Elattar, N. A. Sabiha, M. Alsharef, M. K. Metwaly, A. M. Abd-Elhady, and I. B. M. Taha, "Short term electric load forecasting using hybrid algorithm for smart cities," *Int. J. Speech Technol.*, vol. 50, no. 10, pp. 3379–3399, Oct. 2020.
[5] S. J. Norvig, *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ, USA: Prentice-Hall, 2009.
[6] T. Dillon, C. Wu, and E. Chang, "Cloud computing: Issues and challenges," in *Proc. 24th IEEE Int. Conf. Adv. Inf. Netw. Appl.*, 2010, pp. 27–33.
[7] B. Marr, *Big Data: Using Smart Big Data, Analytics and Metrics to Make Better Decisions and Improve Performance*. Hoboken, NJ, USA: Wiley, 2015.
[8] G. Baltazar. (2018). *CPU vs GPU in Machine Learning*. [Online]. Available: https://www.datascience.com/blog/cpu-gpu-machine-learning
[9] F. Chollet, *Deep Learning with Python*. Shelter Island, NJ, USA: Manning, 2018.
[10] A. A. M. Abadi, "TensorFlow: Large-scale machine learning on heterogeneous systems," Google's Mach. Intell. Res. Org., Tech. Rep., 2015. [Online]. Available: https://arxiv.org/abs/1603.04467
[11] F. Chollet. (2015). *Keras*. [Online]. Available: https://keras.io
[12] A. Paszke, S. Chintala, S. Chanan, G. Yang, E. DeVito, Z. Lin, Z. Desmaison, A. Antiga, and L. Lerer, "Automatic differentiation in PyTorch," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, 2017, pp. 1–77.
[13] A. Narayan and K. W. Hipel, "Long short term memory networks for short-term electric load forecasting," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Banff, AB, Canada, Oct. 2017, pp. 2573–2578.
[14] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, *Recurrent Neural Networks for Short-Term Load Forecasting: An Overview and Comparative Analysis*, 1st ed. Basel, Switzerland: Springer, 2017.
[15] L. Han, Y. Peng, Y. Li, B. Yong, Q. Zhou, and L. Shu, "Enhanced deep networks for short-term and medium-term load forecasting," *IEEE Access*, vol. 7, pp. 4045–4055, 2019.
[16] Z. Yu, Z. Niu, W. Tang, and Q. Wu, "Deep learning for daily peak load Forecasting–A novel gated recurrent neural network combining dynamic time warping," *IEEE Access*, vol. 7, pp. 17184–17194, 2019.
[17] E. E. Elattar, J. Goulermas, and Q. H. Wu, "Electric load forecasting based on locally weighted support vector regression," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 40, no. 4, pp. 438–447, Jul. 2010.
[18] W. Kong, Z. Y. Dong, D. J. Hill, F. Luo, and Y. Xu, "Short-term residential load forecasting based on resident behaviour learning," *IEEE Trans. Power Syst.*, vol. 33, no. 1, pp. 1087–1088, Jan. 2018.
[19] B. Li, J. Zhang, Y. He, and Y. Wang, "Short-term load-forecasting method based on wavelet decomposition with second-order gray neural network model combined with ADF test," *IEEE Access*, vol. 5, pp. 16324–16331, 2017.
[20] H. Jiang, Y. Zhang, E. Muljadi, J. J. Zhang, and D. W. Gao, "A short-term and high-resolution distribution system load forecasting approach using support vector regression with hybrid parameters optimization," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3341–3350, Jul. 2018.
[21] W.-J. Lee and J. Hong, "A hybrid dynamic and fuzzy time series model for mid-term power load forecasting," *Int. J. Electr. Power Energy Syst.*, vol. 64, pp. 1057–1062, Jan. 2015.

[22] C.-J. Huang and P.-H. Kuo, "Multiple-input deep convolutional neural network model for short-term photovoltaic power forecasting," *IEEE Access*, vol. 7, pp. 74822–74834, 2019.
[23] P.-H. Kuo and C.-J. Huang, "A high precision artificial neural networks model for short-term energy load forecasting," *Energies*, vol. 11, no. 1, p. 213, Jan. 2018.
[24] Y. Shen, Y. Ma, S. Deng, C.-J. Huang, and P.-H. Kuo, "An ensemble model based on deep learning and data preprocessing for short-term electrical load forecasting," *Sustainability*, vol. 13, no. 4, p. 1694, Feb. 2021.
[25] Z. Wen, L. Xie, Q. Fan, and H. Feng, "Long term electric load forecasting based on TS-type recurrent fuzzy neural network model," *Electr. Power Syst. Res.*, vol. 179, Feb. 2020, Art. no. 106106.
[26] D. L. Y. Guan, S. Xue, and Y. Xi, "Feature-fusion-kernel-based Gaussian process model for probabilistic long-term load forecasting," *Neurocomputing*, vol. 426, pp. 174–184, Feb. 2021.
[27] M.-R. Kazemzadeh, A. Amjadian, and T. Amraee, "A hybrid data mining driven algorithm for long term electric peak load and energy demand forecasting," *Energy*, vol. 204, Aug. 2020, Art. no. 117948.
[28] B. Kermanshahi and H. Iwamiya, "Up to year 2020 load forecasting using neural nets," *Int. J. Electr. Power Energy Syst.*, vol. 24, no. 9, pp. 789–797, Nov. 2002.
[29] J. Kumaran and G. Ravi, "Long-term sector-wise electrical energy forecasting using artificial neural network and biogeography-based optimization," *Electr. Power Compon. Syst.*, vol. 43, no. 11, pp. 1225–1235, Jul. 2015.
[30] M. Çunka and A. A. Altun, "Long term electricity demand forecasting in turkey using artificial neural networks," *Energy Sour., B, Econ., Planning, Policy*, vol. 5, no. 3, pp. 279–289, 2010.
[31] T. Chen and Y.-C. Wang, "Long-term load forecasting by a collaborative fuzzy-neural approach," *Int. J. Electr. Power Energy Syst.*, vol. 43, no. 1, pp. 454–464, Dec. 2012.
[32] W. Aribowo, S. Muslim, and I. Basuki, "Generalized regression neural network for long-term electricity load forecasting," in *Proc. Int. Conf. Smart Technol. Appl. (ICoSTA)*, Feb. 2020, pp. 1–5.
[33] L. Ekonomou, "Greek long-term energy consumption prediction using artificial neural networks," *Energy*, vol. 35, no. 2, pp. 512–517, Feb. 2010.
[34] D. Ali, M. Yohanna, M. I. Puwu, and B. M. Garkida, "Long-term load forecast modelling using a fuzzy logic approach," *Pacific Sci. Rev. A, Natural Sci. Eng.*, vol. 18, no. 2, pp. 123–127, Jul. 2016.
[35] R. M. Nezzar, N. Farah, M. T. Khadir, and L. Chouireb, "Mid-long term load forecasting using multi-model artificial neural networks," *Int. J. Electr. Eng. Informat.*, vol. 8, no. 2, pp. 389–401, Jun. 2016.
[36] M. EA Feilat *et al.*, "Long-term load forecasting using neural network approach for Jordan's power system," *Eng. Press*, vol. 1, no. 2, pp. 43–50, 2017, doi: 10.28964/EngPress-1-108.
[37] S. Hochreiter and J. J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[38] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN Encoder–Decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1–8.
[39] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Oct. 1997.
[40] *Long Short-Term Memory*. Accessed: Feb. 20, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Long_short-term_memory
[41] D. M. Hawkins, "The problem of overfitting," *J. Chem. Inf. Comput. Sci.*, vol. 44, no. 1, pp. 1–12, Jan. 2004.
[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 4, pp. 1929–1958, May 2014,
[43] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," 2017, *arXiv:1712.04621*. [Online]. Available: http://arxiv.org/abs/1712.04621
[44] Y. Yao, L. Rosasco, and A. Caponnetto, "On early stopping in gradient descent learning," *Constructive Approximation*, vol. 26, no. 2, pp. 289–315, Aug. 2007.
[45] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Oct. 2013, pp. 6645–6649.
[46] K. Darpa-Isto, *The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus (TIMIT)*. Gaithersburg, MD, USA: National Institute of Standards and Technology, 1990.
[47] C. G. Razvan Pascanu, K. Cho, and Y. Bengio, "How to Construct Deep Recurrent Neural Networks," *CoRR*, vol. abs/1312.6026, pp. 1–9, Oct. 2014.
[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[49] V. Bushaev. *Adam—Latest Trends in Deep Learning Optimization*. Accessed: Feb. 20, 2021. [Online]. Available: https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c
[50] M. A. Cabinet, "Data integration partnership for Australia," Austral. Energy Market Operator, Melbourne, VIC, Australia, Tech. Rep., 2017.
[51] AEMO. (2019). *The Australian Energy Market Operator*. [Online]. Available: https://www.aemo.com.au/
[52] TBO Meteorology. (2019). *Australia's National Weather, Climate and Water Agency*. [Online]. Available: http://www.bom.gov.au/
[53] G. A. R. Hyndman, *Forecasting: Principles and Practice*, vol. 8. Melbourne, VIC, Australia: OTexts, 2018.
[54] Y. Kareem and A. Majeed, "Monthly peak-load demand forecasting for sulaimany governorate using SARIMA," in *Proc. IEEE/PES Transmiss. Distrib. Conf. Expo., Latin Amer.*, 2006, pp. 1–8.
[55] S. Gn, "Short-term load forecasting using ARIMA model for Karnataka state electrical load," *Int. J. Eng. Res. Develop.*, vol. 13, pp. 75–79, Oct. 2017.
[56] B. Kleiner, "Time series analysis: Forecasting and control," *Technometrics*, vol. 19, no. 3, pp. 343–344, Aug. 1977.
[57] V. N. Vapnik, *Statistical Learning Theory*. Hoboken, NJ, USA: Wiley, 1998.
[58] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statist. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004.
[59] A. Kavousi-Fard, H. Samet, and F. Marzbani, "A new hybrid modified firefly algorithm and support vector regression model for accurate short term load forecasting," *Expert Syst. Appl.*, vol. 41, no. 13, pp. 6047–6056, Oct. 2014.
[60] C. Hsu, C. Chang, and C. Lin, "A practical guide to support vector classification," Dept. Comput. Sci. Inf. Eng., Nat. Taiwan Univ., Taipei, Taiwan, Tech. Rep., 2003. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[61] *Azure Machine Learning Service*. Accessed: Sep. 7, 2019. [Online]. Available: https://azure.microsoft.com/en-in/services/machine-learning-service/
[62] *Azure Machine Learning Studio*. Accessed: Sep. 7, 2019. [Online]. Available: https://azure.microsoft.com/en-us/services/machine-learning-studio/
[63] I. Anaconda. (2018). *Anaconda Software Distribution*. [Online]. Available: https://www.anaconda.com

TAMER AHMED FARRAG was born in 1981. He received the B.Sc., M.Sc., and Ph.D. degrees from the Department of Computers and Systems Engineering, Mansoura University, Egypt, in 2002, 2006, and 2012, respectively. He works as an Assistant Professor with the Department of Communications and Computers, MISR Higher Institution of Engineering and Technology. His research interests include artificial intelligence, optimization, programming languages, and computing systems.

EHAB E. ELATTAR (Senior Member, IEEE) was born in 1976. He received the B.Sc. (Hons.) and M.Sc. degrees in electrical engineering from the Department of Electrical Engineering, Menoufia University, Egypt, in 1999 and 2003, respectively, and the Ph.D. degree from the Department of Electrical Engineering and Electronics, University of Liverpool, U.K., in 2010. From 2010 to 2016, he was a Lecturer with the Department of Electrical Engineering, Menoufia University, where he promoted to an Associate Professor, in 2016. He joined the Department of Electrical Engineering, College of Engineering, Taif University, Saudi Arabia. His research interests include power systems analysis and operation, artificial intelligence, and modern optimization methods and their applications to power systems operation and integrating of renewable energy sources into power systems.