Applied Computer Science, vol. 15, no. 4, pp. 75–92
doi:10.23743/acs-2019-31

Submitted: 2019-10-29
Revised: 2019-11-25
Accepted: 2019-12-01

Short-term Load Forecasting, Deep Learning Architectures, RNN, LSTM, CNN, SAE

Saheed ADEWUYI[*], *Segun AINA*[**], *Moses UZUNUIGBE*[***], *Aderonke LAWAL*[**], *Adeniran OLUWARANTI*[**]

AN OVERVIEW OF DEEP LEARNING TECHNIQUES FOR SHORT-TERM ELECTRICITY LOAD FORECASTING

Abstract

This paper presents an overview of some Deep Learning (DL) techniques applicable to forecasting electricity consumptions, especially in the short-term horizon. The paper introduced key parts of four DL architectures including the RNN, LSTM, CNN and SAE, which are recently adopted in implementing Short-term (electricity) Load Forecasting problems. It further presented a model approach for solving such problems. The eventual implication of the study is to present an insightful direction about concepts of the DL methods for forecasting electricity loads in the short-term period, especially to a potential researcher in quest of solving similar problems.

1. INTRODUCTION

The power systems structure is characterised by complex infrastructures that are necessary for the sourcing and delivery of electricity to end-users. In order to deliver electricity to end-users, the power Generation Company (GenCo) will transport power through networks of power transmission lines, which is controlled by the Transmission Companies (TransCo). The Distribution Companies (DisCo), also known as the Utilities, receive power from the TranCo and ensure its safe delivery to consumers.

[*] Osun State University, Department of Information and Communication Technology, Osogbo, Osun State, Nigeria, saheed.adewuyi@uniosun.edu.ng
[**] Obafemi Awolowo University, Department of Computer Science and Engineering, Ile-Ife, Osun State, Nigeria, s.aina@oauife.edu.ng,
[***] Transmission Company of Nigeria, 132/33 kV, Ajebandele, Ile Ife, Osun State, Nigeria, moses_uzunuigbe@yahoo.com

The utilities have the responsibility to meet the electricity demand of their customers. Electricity demand is the load on the electrical system which the system must satisfactorily bear and service for customers. This load or demand increases as population increases. To manage this, Utilities need to carry out load forecasts of electricity ahead of need. Electricity load forecasting is germane to the GenCo, DisCo, and TransCo stakeholders, especially in a deregulated economy. The electricity market deregulation and unbundling of the power industry has engendered this even more. Each of the resulting companies that is, the GenCo, TransCo and DisCo have responsibilities to meet the demand of her customers ahead of electricity supply. Therefore, electricity load forecasting is their essential routine.

Load forecasting in power systems is the prediction of users' demands on the grid prior to actual consumption. Load forecasting will, therefore, help the power players across classes to manage the power system's load effectively and efficiently. With load forecasting, the Utilities will especially make essential decisions critical to its operation and planning. This includes purchasing decision and power generation decision. Also, the decisions can be one of the following: Load switching; infrastructure development; capacity planning; maintenance schedules; energy demand; production adjustment; and contract evaluation (Ghullam & Angelos, 2017; Kuo & Huang, 2018; Seunghoung, Hongseok & Jaekoo, 2017).

Therefore, tackling the problem of electricity consumption forecasting using deep learning techniques involve simplifying it based on Fig. 1. Fig. 1 shows that load forecasting problem can be simplified based on five categories: Model, Horizon, Aim, Variables and Area (Luis *et. al.*, 2012, 2013, 2014). Load forecasting is simplified based on the type of the model to develop, which results in linear or non-linear model categorisation. Fig. 1 further classifies the problem into four categories based on the horizon (Feinberg & Genethliou, 2005; Luis *et. al.*, 2012). This includes Very Short Term Load Forecasting (VSTLF) which falls within seconds or minutes, Short Term Load Forecasting (STLF); spanning a week from an hour, Medium-Term Load Forecasting (MTLF); which covers a period from a week up to a few months and Long Term Load Forecasting (LTLF); which is usually more than a year (Luis *et. al.*, 2012, 2013, 2014). Furthermore, the electricity load forecasting is simplified based on the aim of the forecast that is, the values to be predicted, which can be a single value or multiple values (Luis *et. al.*, 2014). When considering certain factors influencing load consumption, which are ingredients of forecast estimation, it is essential to include time, weather and customer class (Feinberg & Genethliou, 2005; Swalin, 2019). These factors are classified as variables in Fig. 1. They, therefore, include variables such as load data and calendar data; or a combination of load data, calendar data and weather parameters; or another combination of load data, calendar data and other data variables which may be demographic, economic and social in nature, usually prevalent in the residential class of electricity end-users' class.

The Fig. 1 also categorises area of forecast estimation as including country, region/city and community/microgrids. In any situations, load forecasting can be implemented for one of these identified categories.

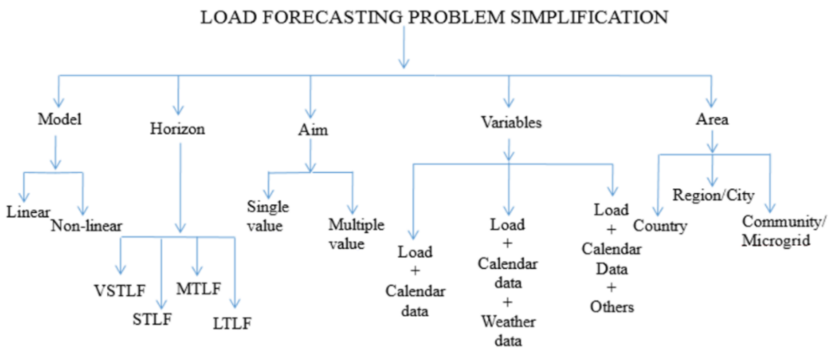

Fig. 1. Load forecasting problem categorisation

In this paper, the scope of the study is limited to four Deep Learning (DL) algorithms already established in the literature (Bengio, 2009; Brownlee, 2018; Chengdong, Zixiang, Dongbin, Jianqiang & Guiqing, 2017; Deng, 2013; Ghullam & Angelos, 2017; Hamedmoghadam, Joorabloo & Jalili,2018; Hussein & Hussein, 2017, Kuo & Huang, 2018; Nor, Rahaini & Siti, 2018; Swalin, 2019) for modelling electricity load forecasting. The techniques are the standard Recurrent Neural Network (RNN), Long Short Term Memory (LSTM) network, Convolutional Neural Network (CNN), and Stacked Autoencoder (SAE). Also, the concepts of deep learning were introduced. These cover the motivating influences for the technology; its classification, as well as a few terminologies of deep architectures. Also, the problems with primitive methods for modelling sequential electricity load were highlighted. As a glimpse, we present a concise approach to developing load forecasting problems. This will be to define a robust or scalable model, preprocess data and carry out performance evaluation of the model.

The rest of the paper is arranged as follow. Section 2 presents background knowledge and an overview of modelling electricity load forecasting in the short-term. Section 3 extends the discussion on deep architectures relevant for STLF problems. Here, the four DL techniques are discussed in detail. In Section 4, an approach towards implementing the STLF problem is presented. Finally, in Section 5, conclusions from the work are drawn.

2. BACKGROUND

In order to establish a background into the study of electricity load forecasting, there is a need to introduce the concept from the perspective of how a model can be developed for it. We categorized the approaches into two: the primitive or classical approach and the DL approach. These two are briefly discussed in the subsections that follow.

2.1. Primitive approach

Classical approaches have been researched and discussed in academia and are applied in the industry with varying success. The techniques are well reported. Linear methods like Auto-Regressive Integrated Moving Average (ARIMA) model have been widely chosen because they are easily understood and well effective on some problems (Brownlee, 2018). This classical approach has three variants including itself, ARIMA and two others namely: Auto-Regressive Moving Average (ARMA) and Autoregressive (AR) models (Nor, Rahaini & Siti, 2018). ARIMA models are regression type of models that adopt lagged values of the dependent variable and or random disturbance term as explanatory variable (Sarabjit & Rupinderjit, 2013). The explanatory variables have in-built dependence relationship (Sarabjit & Rupinderjit, 2013). The model is an integration of two autoregression and moving average models. A Seasonal ARIMA can be hybridised with a Back Propagation Algorithm Neural Network to achieve a more accurate load forecasting (Yi, Jie, Yanhua & Caihong, 2013). It will, however, be interesting to highlight that these traditional methods have suffered from various limitations:

1. Complete data: This means it sees data as wholesome and cannot manage issues of missing or corrupt data automatically.
2. Linear relationships: This implies that it addresses only linearities and leaves out complexities in the data distributions.
3. Fixed temporal dependence: This implies that the relationship between observations at each time-step as well as the number of lagged features in the input must be scrutinised and explicitly stated.
4. Univariate data: Usually, real-life problems are characterised by multiple variables as input, but classical approaches are mostly able to handle univariates.
5. One-step forecast: Practical problems will require multiple-step forecasts than the single-step forecast characterized by primitive models.

The problems highlighted, therefore, leave users with the requirement to hand-engineer features which are expensive to create (Gamboa, 2017). At the wake of these issues, Deep learning (DL) techniques emerged. Their inventions have tremendously advanced the sphere of artificial intelligence capabilities to solving vast human problems. In fact, DL has helped in providing dependable solutions to those problems sustained by primitive methods while handling more sophisticated problems.

2.2. Concept of deep learning

Deep learning is a machine learning technique that learns features and tasks directly from data (Brownlee, 2018). Data can be images, text, or sound. Any of these data types can be used as input to deep learning models for problem-solving. Deep learning also refers to a class of machine learning techniques, where many layers of information-processing stages in hierarchical architectures are exploited for pattern classification and representation or feature learning (Deng, 2013). Further to this, deep learning involves neural networks that are able to naturally learn arbitrary complex mappings from inputs to outputs. It also has support for multiple inputs and outputs. Interestingly, some of these features offer a great promise for electricity load forecasting (Brownlee, 2018), particularly on problems characterised by complex/nonlinear dependencies, or multivalent inputs, and multi-step forecasting. These features and other neural network capabilities offer great promises, such as the automatic feature learning characteristic of convolutional neural networks and the natural support for sequential data in recurrent neural networks (Brownlee, 2018). Sequential data are datasets whose features are constrained by time, making it a little difficult. Electricity load data is an example. Electricity load profile of a customer is measured as a function of time (per hour). This is evident in customers' electricity bills, which is computed in terms of energy used. A unit of energy used is measured in Kilowatt Hour (kWh). Furthermore, electricity loads, unlike other machine learning problems solved by classification of labels or even regression analysis of quantities, add time complexity. This inherently makes them have certain temporal dependencies among data features (Brownlee, 2018). The temporal dependencies, therefore, introduce difficulties in handling data for the purpose of model's fitting and evaluation. Conversely, the temporal structure characterising the electricity load can equally enhance modelling by providing added structures such as trends and seasonality, which when leveraged improve model performance on problems (Brownlee, 2018).

2.2.1. Motivation for deep learning

Although deep learning techniques have been in use for some time, in recent times it gained a lot of popularity due to certain developments. These, according to (Deng, 2013) are: First, due to DL techniques' increased accuracy at performing several human-related tasks. This is as a result of recent advances in machine learning and signal/image processing. Second, as a result of the increased chip processing ability, such as the Graphics Processing Units, engendered by a high reduction in the cost of computing hardware. Third, there are larger volumes of labelled data available.

The above highlighted motivating points that are key to DL algorithms' application on a particular problem. The same motivations are the reasons for applying DL techniques on electricity loads forecasting problems.

2.2.2. Classifying deep learning architectures

Most deep learning architectures use neural network-based methods. This is why some deep learning techniques such as Deep Belief Networks (DBN) are interchangeably referred to as Deep Neural Network (DNN) in literature (Deng & Yu, 2013). The term deep in deep learning refers to the number of hidden layers in the neural network. The hidden layer count can be unlimited. In order to classify deep learning architecture, a three-way classification scheme is summarized from various work done by researchers and industry experts. These classification schemes are grouped into generative, discriminative, and hybrid algorithms (Deng, 2013; Deng & Yu, 2013).

The generative deep architectures are learning architectures intended to characterize high-order correlation properties of the observed or visible data for pattern analysis or synthesis purposes, and/or characterize the joint statistical distributions of the visible data and their associated classes. They are unsupervised learning algorithms. Examples include autoencoders, Boltzmann machine and sum-product network (Deng, 2013). In relation to this, we can assert that some electricity load forecasting problems belong to this category. This is because the problem has been tackled with a few DL algorithms, such as the auto-encoder, in this class. This is documented in (Hussein & Hussein, 2017).

The discriminative type of deep architecture focuses on direct provision of discriminative power for pattern classification instances, often achieved by characterising the posterior distribution of classes conditioned on the visible data. These architectures are supervised learning in nature (Deng & Yu, 2013). Examples of such models include some learning algorithms like stacked networks, recurrent neural network and convolutional neural network. We also found in (Hussein & Hussein, 2017) that electricity load forecasting problem is a member of this class.

The hybrid deep architectures are either comprising or making use of both generative and discriminative model components. This architecture type has the goal of discrimination, assisted at times by the outcomes of generative deep networks. This can be accomplished by better optimization and or regularization of the deep networks in discriminative models. The goal can also be accomplished when discriminative criteria for supervised learning are used to estimate the parameters in any of the deep generative or unsupervised deep networks in generative models (Deng, 2013; Deng & Yu, 2013). Examples of such architectures include DNN-DBN model, DNN-Conditional Random Field (CRF) among several others. Similarly, electricity load data have also been solved by hybridised algorithms. The work of (Hussein & Hussein, 2017) is good case. The study combined DNN with SAEs and CNN with LSTM.

2.2.3. Application of deep learning techniques to STLF problem

Deep learning architectures have been applied to acoustics, images and signal processing studies, with tremendous successes (Merkel, Povinelli & Brown, 2017). For this reason, its applicability to electricity load forecasting problems is also a possibility because load profiles are characterised by some non-linear factors. For instance, in (Hussein & Hussein, 2017), some deep learning techniques were analysed. The DL architecture analysed include the Feed Forward Neural Network (FFNN), that is characterised by influx of input signal from the input layer to the output layer, one layer at a time, without looping back. Other model architectures also analysed in (Hussein & Hussein, 2017) are the Recurrent Neural Network (RNN), which allows data to flow in any direction, and the Convolutional Neural Network (CNN), which is applied to computer vision problems and acoustic modelling as well. Similarly, the other architectures are the Stacked Autoencoders (SAE) and Long Short Term Memory (LSTM). Seunghyoung *et al.* (2017), the need to investigate important aspect of Demand-Side Management (DMS) was studied on forecasting electricity loads. The study forecasts individual customer's daily load using deep neural network based approach. Ghullam and Angelos (2017), developed Feed-Forward DNN and Recurrent DNN models to predict short term electricity loads. The study analysed time and frequency as features influencing electricity load demand. Wan (2014), presented Restricted Boltmann Machine (RBM) as deep learning pre-training method for STLF problem. Kuo and Huang (2018), studied an introduction of accurate deep neural network algorithm for short-term load forecasting (STLF). Rahul *et al*. (2018), developed a novel approach for long-term load forecasting; although, this was with the aim to forecasting electricity loads at hourly horizon. In Hussein (2018), an investigation into application of DNN to forecasting electricity loads was done for a DisCo. The study also proposed a multi-layered DNN's system for the problem. The next section discusses more in perspective some of these deep learning techniques used for predicting electricity loads especially in the short term horizon.

3. SPECIFIC TECHNIQUES AND METHODS FOR ELECTRICITY LOAD FORECASTING

In this section we introduce four of the deep learning techniques with their structures, which have been adopted in the literature for solving short term electricity load forecasting problems.

3.1. Recurrent neural network

Recurrent Neural Network (RNN) is a deep learning technique with a long history, but only become popular as result today is traceable to the work published by Schmidhuber and Sepp (1997) and a few other researchers. An RNN can be understood as copies of a single network but each one transferring its signal to another as in Fig. 2. So, to recognise the need for an RNN, patterns in signals must be observed to change with time, just as in a typical electricity loads data. In such scenarios the best model is an RNN or its advanced variant, the LSTM.

This deep learning model has a simple structure with a built-in feedback loop, see Fig. 2; which allows the network to transfer electricity loads from previous time-step to next time-step. This capability thereby results in a situation referred to as persistent flow of information, recognised as RNNs' memory capabilities. The architecture of an RNN consists of units interacting in discrete time via weighted and directed connections with weights wij, linking unit j to i, with i being the first unit and j the last unit of the network (Hussein & Hussein, 2017).

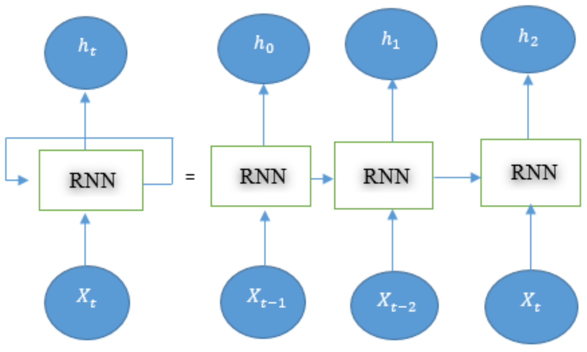

Fig. 2. Standard RNN structure

Furthermore, every unit has an activation function $\widehat{m}(t)$ which is adjusted at every time-step, $t = (1,2,3,..)$ for each electricity load exposed to it. An activation, $\widehat{m}^i$ of unit i is updated by computing its network input sum N^i where:

$$N^i = \sum_j w_{ij}\widehat{m}(t-1) \tag{1}$$

and squashing it with a differentiable function like sigmoid function σ results in:

$$\widehat{m}(t) = \sigma(N^i(\mathrm{t})) \tag{2}$$

3.2. Convolutional neural network

A Convolutional Neural Network (CNN) is one of the most popular algorithms for deep learning with images and videos. Like other algorithms, a CNN is composed of an input layer, an output layer, and many hidden layers sandwiched. The CNN provides better accuracy in highly non-linear problems. The CNN uses the idea of weight sharing whose sets are treated as kernels (Merkel, Povinelli, & Brown, 2017). Fig. 3. is a one dimensional convolution and pooling layer. After the convolution process, the inputs $x1, x2, x3, x4, x5, x6$ (in this case are the electricity load consumed by power users) are transformed to the feature maps $c1, c2, c3, c4$.

Pooling follows, wherein the feature map of convolution layer is sampled and dimension is reduced. The feature dimension before pooling is 4 but after the process the dimension is reduced to 2, as shown in Fig. 3.

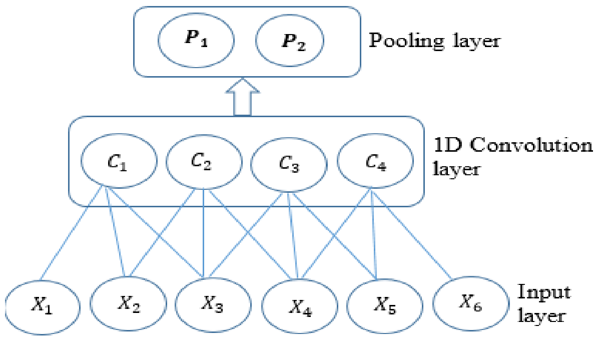

Fig. 3. One dimensional convolution and pooling

The pooling process is a vital procedure of the CNN architecture, for extraction of essential convolution features. A feature map is derived by repeated use of a function across sub-regions of the entire image, that is, by convolution of the input image with a linear filter adding a bias term and then applying a non-linear function. So by denoting the mth feature map at a given layer as h^m, for which filters are determined by the weights W^m and bias b_m, the feature map h^m is derived as in Eq. (3), for hyperbolic non-linearities:

$$h_{ij}^{m} = \tanh(W^m * x)_{ij} + b_m \tag{3}$$

In order to obtain a richer representation of the electricity load data, the hidden layer can be stacked, that is structured to compose multiple feature maps as in (Hussein & Hussein, 2017).

3.3. Stacked autoencoders

Stacked Autoencoders (SAEs) are autoencoders characterised by some multiple building blocks to construct its deep structure (Chengdong *et al.*, 2017). The SAEs utilise stacked architecture, with an autoencoder in each layer (Hussein and Hussein, 2018). Autoencoders are neural networks that have the power to encode its input data, such as electricity loads consumption, into a new representation using unsupervised type of learning. They are hidden layer of neurons that are trained to encode raw input data into a new representation and decode them to reconstruct the original input with minimal deformation possible (Hamedmoghadam, Joorabloo & Jalili, 2018). The target, that is the output, is equal to the input of the model (Chengdong *et al.*, 2017; Hussein, 2018).

The following are, therefore, three important things to take note about autoencoder:

1. Autoencoders are data-specific: This means that the load forecasting deep learning technique will only be able to encode those data similar to what had seen before.
2. Autoencoders are lossy: This will mean that the decoded outputs that is, the electricity load reconstruction, in this case, will be degraded and compared to its original inputs.
3. Autoencoders are learned automatically from data examples: This means it is easy to train specialised instances of the algorithm that will perform well on specific type of input. Therefore, no new data feature engineering is required, but data training.

As an illustration however, Fig. 4. is a simple autoencoder with H hidden layer nodes. As a matter of fact, autoencoder has two main parts, the encoder and decoder. The encoding process seeks to exploit and then reveal a hidden representation $\sigma_1(\mathrm{x})$, of typical electricity load profiles, which can be computed as:

$$\sigma_1(\mathrm{x}) = \mathrm{f}(w_1 x + b_1) \tag{4}$$

where: w_1 − is an encoding matrix,
b_1 − is an encoding bias vector,
$f(\cdot)$ − is the activation function.

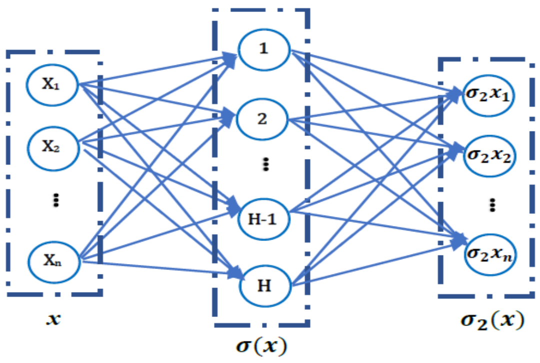

Fig. 4. A simple autoencoder

As a matter of fact, autoencoder has two main parts, the encoder and decoder. The encoding process seeks to exploit and then reveal a hidden representation $\sigma_1(x)$, of typical electricity load profiles, which can be computed as:

$$\sigma_1(x) = f(w_1 x + b_1) \tag{4}$$

where, w_1 is an encoding matrix, $b1$ is an encoding bias vector, and $f(\cdot)$ is the activation function. The activation function can be any of the sigmoid, Rectified Linear unit (ReLu) or tanh functions. Conversely, the decoding process requires that a decoding matrix be determined in order to recover the reconstructed hidden representation $\sigma_1(x)$, back into its original form that is, $\sigma_2(x)$. This therefore leads to the computation of the decoded output as in Eq. (5):

$$\sigma_2(x) = g(w_2 \sigma_1(x) + b_2) \tag{5}$$

where: w_2 – is the decoding matrix,
b_2 – is the decoding bias vector,
$g(\cdot)$ – is the encoding activation function.

It is expected that the error between the input x and the reconstruction $\sigma_2(x)$ is as minimal as possible. To ascertain this, it is imperative to minimise the loss function Eq. (6) below:

$$L(\sigma_2(x)) = \frac{1}{2}\sum_{m=1}^{N} \left\| x^{(m)} - x_2(x^{(m)}) \right\|^2 \tag{6}$$

Moreover, the optimal parameter set of the autoencoder can as well be known by solving the following optimization problem:

$$\Psi = \{w_1, w_2\} = \underset{w_1, w_2}{\overset{arg}{}} \min L\,(x, \sigma_2(x)\,) \tag{7}$$

where: w_1 – is the encoding,
w_2 – is the decoding matrices,
$L(x, \sigma 2(x))$ – is loss function, minimised for optimisation purposes,
Ψ – is a notation defined for the optimisation problem.

In the autoencoder, this optimization problem, Ψ is often solved using one of the variants of the backpropagation algorithms, such as the conjugate gradient method or the steepest descent method (Chengdong *et al.*, 2017).

In summary, the technique affords extraction of useful features essential to forecasting electricity loads.

3.4. Long Short-Term Memory

This is a special kind of RNN. It was developed to overcome the lingering problem of long-term dependency suffered by the standard RNN architecture. This, therefore, provides dependable solutions to a lot of sequence problems. The LSTM networks structure consist of many connected LSTM cells as simply depicted in Fig. 5. The Fig. 5 is a single cell of an LSTM network showing arrows pointing towards the structure and another exiting from it. These will mean that there exists connected LSTM cells before and after it.

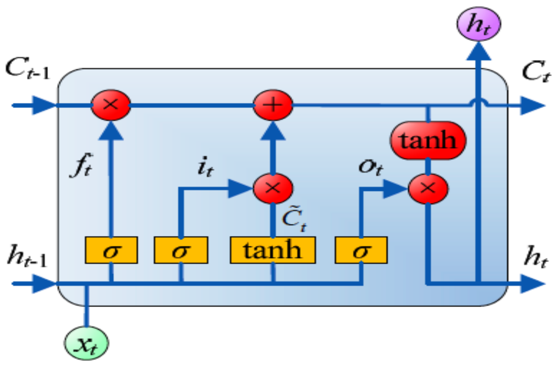

Fig. 5. LSTM Structure (Kuo & Huang, 2018)

The LSTM internal structure is characterised by four interacting neural network layers, also known as the repeating modules. The main idea about the LSTM network operating mechanism is its cell state and the gates layers. The cell state can be termed as a conveyor belt and runs straight down the modules, with only minor interactions, making data to flow seamlessly and uncorrupted down the chain. The LSTM is capable to remove or add information to the cell state.

This is carefully regulated by the gates. The gates are a way to optionally let data input, such as an electricity load features, through the cell state. They are composed of a sigmoid neural net layer and a pointwise multiplication operation as in Fig. 5. The sigmoid layer output is binary, describing how much of each component should pass through the gate or not. A '0' means "let nothing through," while a '1' will mean "let everything through!" An LSTM has three of these (forget, input and output) gates, to protect and control the cell state.

The following are the system of equations that were established in the literature as responsible for the satisfactorily operating performance of the deep learning LSTM technique.

$$f_t = \sigma\left(W_f . [h_{t-1}, x_t] + b_f\right) \tag{8}$$

$$i_t = \sigma(W_i . [h_t - 1, x_t] + b_i) \tag{9}$$

$$\widehat{C_t} = tanh(W_C . [h_t - 1, x_t] + b_c) \tag{10}$$

$$C_t = f_t \times C_{t-1} + i_t \times \widehat{C_t} \tag{11}$$

$$o_t = \sigma(W_o \cdot [h_t - 1, x_t] + b_o) \tag{12}$$

$$h_t = o_t \times tanh(C_t) \tag{13}$$

Eq. (8) decides which of the previous information to be discarded of the cell state. This decision is made via the sigmoid layer called forget gate layer. In Eq. (9) and Eq. (10), decision on new information to be included to the cell state is made. There are two parts. The first is the sigmoid layer termed input gate layer, which decides which value is to be updated and the second is the hyperbolic tangent layer, which creates vector of new candidate values that could be added to the cell to the state. Eq. (11) updates the old cell state, C_{t-1} into the new cell state, C_t. Notice that the previous steps already decided what to do, so update is only done here. Eq. (12) and Eq. (13) decide what is to be generated from the network, which is the prediction outcome. The output is based on the cell state, in filtered state. First, a sigmoid layer is run, which decides what parts of the cell state is to be output. Then, the cell state is passed through a *tanh* layer and is multiplied by the output of the sigmoid gate, so that only the decided parts are predicted. This is the final stage of the LSTM operating mechanism.

From Eqs. (8) to (13), the notations used are defined as follows: x_t, is the load features input at time t; x_{t-1}, is the previous hidden layer computations. The w_f, w_i, w_c, and w_o are respective weight matrices of the forget, input, cell state and output gate layers of the structure, that are regulating data input inflow. The b_f, b_i, b_c, and b_o, are the respective bias vector for each gate layer and c_{t-1}, is the previous cell state while c_t, is the new cell state. Furthermore, whereas $\hat{c}_t$, is a vector of new candidate values for the cell state; σ and $tanh$, are respectively sigmoid and hyperbolic tangent activation functions. The notation o_t, is the output of the sigmoid gate and o_h, is the output of the current hidden layer.

In summary, LSTM is an exciting model for forecasting power loads because of its ability to effectively handle datasets characterised by an order of time such as the electricity consumption data.

3.5. Modelling Approach

In order to forecasting electricity loads in the short- term horizon, there is need to synthesise the problem into its constituents. These implies the defining, compiling, fitting, evaluating and predicting the model. These technical constituents of model development are adoptable by adapting the Fig. 6.

The subsections that follow further detail a typical roadmap for applying electricity load data on any of the deep learning techniques discussed so far, in order to forecast the next hour or day-ahead consumption profile.

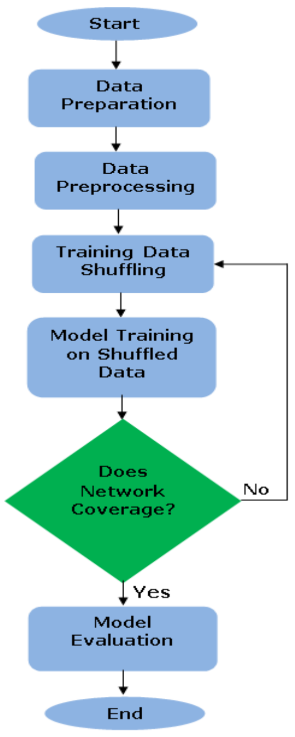

Fig. 6. A simple flowchart for load forecasting

3.5.1. Model training data source

Data is an important aspect of STLF forecasting problem. Therefore, in order to tackle the problem, it is imperative to known the source of the model training dataset. The model data source is how an electricity forecast engine obtains its training set, for the network prediction goal. So, a two-way classification was developed and is delineated in Fig. 7. This is the engineering approach and Artificial Intelligence (AI) or data-driven approach. The engineering method obtains model training data from the context features of the building structure. It also gets the data via the system information of the Heating, Ventilating, and Air Conditioning (HVAC) appliances of that structure and other home appliances, which load forecasting task is estimated. Conversely, the AI or data-driven approach gets training data from the historic electricity consumption data of the study area. So, depending on the load forecasting task in view a researcher would have to make decisive choice regarding model training set.

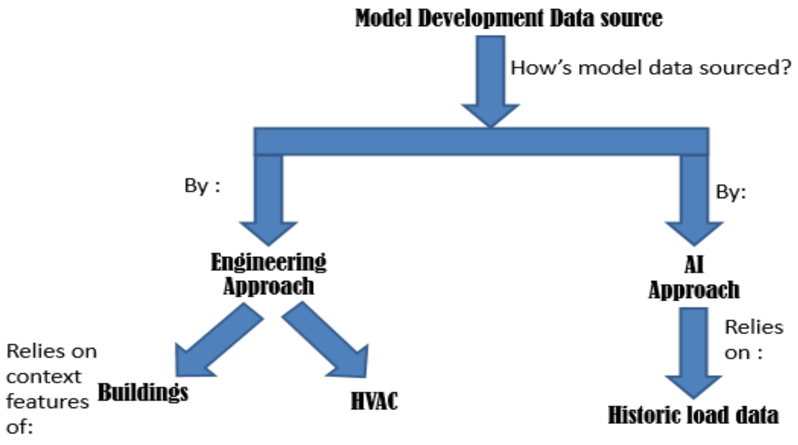
Model Development Data source
How's model data sourced?
By :
By:
Engineering Approach
AI Approach
Relies on
on :
Relies on context features of:
Buildings
HVAC
Historic load data

Fig. 7. How model obtains training data

3.5.2. Data preprocessing

Having obtained the model's training data, the next task will be to carry out appropriate preprocessing procedure on the electricity load forecasting problem. This approach will usually involve data cleansing and preparation. Data cleansing would mean that the data be devoid of any incompleteness or missing value, noise, and inconsistency. Data cleaning routines work to "clean" the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies. A typical framework for handling missing data is presented in (Swalin, 2019). Preparing data would require that it is preprocessed by scaling numeric data and transforming categorical data. Numeric data scaling will improve network stability and modelling performance. This can be achieved through normalization and standardization methods. Similarly, data transformation is converting labels data to quantities, this is achieved in two standard steps namely: integer encoding and one-hot encoding (Brownlee, 2018).

4. EVALUATION METRICS

The evaluation of model's performance of load forecasting model can be assessed by obeying the objective function of the model. The applicability and suitability of a deep learning model on typical electricity load forecasting problem is measured by some evaluation metrics such as the root mean square error and mean absolute percentage error. In general, performance evaluation of a regression type of deep learning model, as in this case, can be measured by one of the following metrics, among others:

4.1. Mean Square Error and Root Mean Square Error

The Root Mean Square Error (RMSE) is a frequently used measure of the differences between samples predicted by a model and the values actually observed. RMSE is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are. RMSE therefore measures how spread out these residuals are. In other words, it speaks about how concentrated the data is around the line of best fit. The RMSE for a training and test sets should be very similar if a good model is built. Formally, the RMSE is given as below:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(Y_{pred} - Y_{true}\right)^2} \qquad (14)$$

Where: $RMSE -$ is evaluation metric of interest,
$N -$ is number of observations,
$Y_{pred} -$ ordinates of the actual loads,
$Y_{true} -$ ordinates of the predicted loads.

4.2. Mean Absolute Error

Mean Absolute Error (MAE) is a term used in determining absolute difference between two variables. Assume, Y_{pred} and Y_{true} are variables of paired observations expressing the same conditions, the MAE is defined as:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|Y_{pred} - Y_{true}\right| \qquad (15)$$

4.3. Mean Absolute Percentage Error

Mean Absolute Percentage Error (MAPE) is yet another approach to evaluating model prediction accuracy. It often expresses accuracy in terms of percentage. This is expressed mathematically as:

$$MAPE = \frac{100\%}{N}\left|\sum_{i=1}^{N}\frac{Y_{pred}-Y_{true}}{Y_{pred}}\right| \tag{16}$$

5. CONCLUSIONS

The foregoing study discusses some of the deep learning techniques that have been applied to electricity load data when modeling typical load forecasting problem. Load forecasting was also categorised as including the type of model, aim of forecast, horizon of forecast, variables of interest in modeling load forecasting problem and the area where forecast is to cover. More so, the interested model data source is categorised as harvestable from an engineering source or AI source. Finally, the DL techniques for modeling a typical electricity load forecasting problem is discussed.

REFERENCES

Bengio, Y. (2009). Learning deep architectures for AI. *Foundation and Trends in Machine Learning*, *2*(1), 1–127.

Brownlee, J. (Ed.) (2018). *Deep learning for time series forecasting: Predicting the future with MLPs, CNNs and LSTMs in Python*. Machine learning mastery.

Chengdong, L., Zixiang, D., Dongbin, Z., Jianqiang, Y., & Guiqing, Z. (2017). Building energy consu-mption prediction: An extreme deep learning approach. *Energies*, *10*(10), 1525–1545.

Deng, L. (2013). A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*, 3(2). doi:10.1017/ATSIP

Deng, L., & Yu, D. (2013). Deep learning: methods and applications. *Foundations and Trends in Signal Processing*, *7*(3-4), 197–387.

Feinberg, E. A., & Genethliou, D. (2005). Load forecasting. In J. H. Chow, F.F. Wu, & J. Momoh (Eds.), *Applied Mathematics for Restructured Electric Power Systems. Power Electronics and Power Systems*. Springer, Boston, MA.

Gamboa, J. (2017). Deep learning for time-series analysis. arXiv: 1701.01887.

Ghullam, M. U., & Angelos, K. M. (2017). Short term power load forecasting using deep neural networks. *ICNC*, *10*(1109), 594–598, 7876196.

Hamedmoghadam, H., Joorabloo, N., & Jalili, M. (2018). *Australia's long-term electricity demand forecasting using deep neural networks*. arXiv: preprint arXiv:1801.02148.

Hussein, A. (2018). *Deep Learning Based Approaches for Imitation Learning* (doctoral dissertation). Robert Gordon University Aberdeen, Scotland.

Hussein, S., & Hussein, P. (2017). Load forecasting using deep neural networks. In *2017 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. IEEE. doi:10.1109/ISGT.2017.8085971

Kuo, P., & Huang, C. (2018). A high-precision artificial neural networks model for short-term energy load management. *Energy*, *11*(1), 213–226.

Luis, H., Carlos, B., Javier, M. A., Lorena, C., Belen, C., Antonio, S., Diane, J. C., David, C., & Jorge, G. (2012). A study of relationship between weather variables and electric power demand inside a smart grid/ smart world. *MDPI Sensors*, *22*(9), 11571–11591.

Luis, H., Carlos, B., Javier, M. A., Lorena, C., Belen, C., Antonio, S., Diane, J. C., David, C., & Jorge, G. (2013). Short-term load forecasting for micro-grids based on artificial neural networks, *MDPI Sensors*, *6*(3), 1385–1408.

Luis, H., Carlos, B., Javier, M. A., Lorena, C., Belen, C., Antonio, S., & Jaime, L. (2014). Artificial neural network for short-term load forecasting in distribution systems, *MDPI*, *7*(3), 1576–1598.

Merkel, G. D., Povinelli, R. J., & Brown, R. H. (2017). Deep neural network regression for short-term load forecasting of natural gas. *Report: Marquette University*.

Nor, H. M., Rahaini, M. S., & Siti, H. H. A. (2018). ARIMA with Regression Model in Modelling electricity load demand, *Journal of Telecommunication, Electronic and Computer Engineering*, *8*(12), 113–116.

Rahul, K. A., Frankle, M., & Madan, M. T. (2018). Long term load forecasting with hourly predictions based on long-short-term-memory networks. In *2018 IEEE Texas Power and Energy Conference (TPEC)*. IEEE. doi:10.1109/TPEC.2018.8312088

Sarabjit, S., & Rupinderjit, S. (2013). ARIMA Based Short Term Load Forecasting for Punjab Region. *IJSR*, *4*(6), 1919–1822.

Schmidhuber, J., & Sepp, H. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.

Seunghoung, R., Hongseok, K., & Jaekoo, N. (2017). Deep neural network based demand side short term load forecasting. *Energies MDPI*, *10*(1), 3–23.

Swalin, A. (2019). How to handle missing data. *Towards Data Science*. Retrieved from https://towardsdatascience.com/how-tohandle-missing-data-8646b18db on 18/01/2019.

Wan, H. (2014). Deep neural network based load forecast. *Computer Modelling and New Technologies, 18*(3), 258–262.

Yi, Y., Jie, W., Yanhua, C., & Caihong L. (2013). *A new strategy for short-term load forecasting*. Hindawi.