Multivariate Short Term Load Forecasting Strategy: Application to Anomalous Days of ISO New England Data

Innocent Sizo Duma[1] and Bhekisipho Twala[2]
[1]*Department of Electrical Power Engineering, Faculty of Engineering and the Built Environment, Durban University of Technology, P.O. Box 1334, Durban, 4000, South Africa*
[2]*Faculty of Engineering and the Built Environment, Durban University of Technology, P.O. Box 1334, Durban, 4000, South Africa*

Keywords: Short Term Load Forecasting, Multivariate Denoising using Wavelet and Principal Component Analysis, Bayesian Optimization Algorithm, Long Short-Term Memory Neural Networks, Feedforward Neural Networks, Random Forest.

Abstract: In this paper, we consider short-term electricity load forecasting which is for making forecasting within 1 hour to 7 days or a month ahead usually used for the day-to-day operations of the utility industry, such as scheduling the generation and transmission of electric energy. This is a three step process: (1) Data preprocessing which include feature extraction, (2) Modeling and (3) Model Evaluation. Electrical load time series are non stationary and notoriously very noisy because of variety of factors that affect the electrical markets. As a data preprocessing step to remove the white noise on the multivariate predictor variables (which include historical load, weather, and holidays) we perform a multivariate denoising using wavelets and principal component analysis (MWPCA). In the modeling step we propose three multivariate Bayesian Optimization (BO) based Random Forest (RF), Feedforward Neural Networks (FFNN) and Long Short-term Memory (LSTM) neural network for day ahead hourly load forecast of the anomalous days system load of the ISO New England grid. For model evaluation we used three evaluation metrics, the Mean Absolute Percent Error (MAPE), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE). All the trained models achieved a superior results on the chosen model evaluation metrics most notably achieving a MAPE of less than 1% on the data under study. And the FFNN model outperformed both the RF and LSTM models.

1 INTRODUCTION

Load forecasting is a central and integral process in the planning and operation of electric utilities. It involves the accurate prediction of both the magnitudes and geographical locations of electric load over the different periods (usually hours) of the planning horizon. The basic quantity of interest in load forecasting is typically the hourly total system load. However , load forecasting is also concerned with the prediction of hourly, daily, weekly and monthly values of the system load, peak system load and the system energy. Load forecasting can be classified in terms of the planning horizon's duration: up to 1 hour for very short-term load forecasting (VSTLF), 24 hours to one week for short-term load forecasting (STLF), more than one week to few months for medium-term load forecasting (MTLF), and 1±10 years for long-term load forecasting (LTLF).

Accurate load forecasting holds a great saving potential for electric utility corporations, these savings are realised when load forecasting is used to control operations and decisions such as dispatch, unit commitment, fuel allocation and off-line network analysis. The accuracy of load forecasts has a significant effect on power system operations, as economy of operations and control of power systems may be quite sensitive to forecasting errors. It is observed that both positive and negative forecasting errors resulted in increased operating costs. Hobbs (Hobbs et al., 1999) quantified the dollar value of improved Short Term Load Forecasting for a typical utility and observed that a 1% reduction in the average forecast error for a $10,000$MW utility can save up to \$1.6 million annually and that was 22 years ago.

Previous work has been carried out on Short term load forecasting for Anomalous Days of ISO New England Grid Data by Raza (Raza et al., 2020) who proposed an ensemble forecast framework with a sys-

Duma, I. and Twala, B.
Multivariate Short Term Load Forecasting Strategy: Application to Anomalous Days of ISO New England Data.
DOI: 10.5220/0010652600003063
In *Proceedings of the 13th International Joint Conference on Computational Intelligence (IJCCI 2021)*, pages 293-301
ISBN: 978-989-758-534-0; ISSN: 2184-2825

tematic combination of three predictors, namely Elman Neural Network (ELM), Feedforward Neural Network (FFNN) and Radial Basis Function (RBF) Neural Network. They trained these predictor models using Global Particle Swarm Optimization (GPSO) to improve their training capability in the ensemble framework. The outputs of individual predictors were combined using trim aggregation technique by removing forecating anomalies. Their predictor variables which include weather, seasonality and historical load were subjected to univariate wavelet denoising to remove fluctuations and spikes. Their proposed model showed a significant improvement in prediction accuracy compared to autoregressive integrated moving average (ARIMA) and back-propagation neural networks (BPNN).

As Raza (Raza et al., 2020) put it succinctly: " For load forecasting of a normal day, a day with a predictable load profile, the training data have enough correlated training samples to train the model. However, an anomalous day load forecasting has a much smaller number of patterns for effective training of the model. Therefore, the anomalous day forecasting model is more complex and difficult to design for higher forecast accuracy. Generally, the prediction accuracy of models for anomalous days is lower due to multiple factors such as uncertainty in demand, meteorological variables, unpredictable sociological events and intermittency of renewable energy resources etc." It is these challenges that encourages us to put foward our methodology.

Nti (Nti et al., 2020) undertook a systematic and critical review of about seventy-seven (77) relevant previous works reported in academic journals over nine years $2010 - 2020$ in electricity load forecasting. Specifically, attention was given to the following themes: (i) The forecasting algorithm used and their fitting ability in this field, (ii) the theories and factors affecting electricity consumption and the origin of research work, (iii) the relevant accuracy and error metrics applied in electricity load forecasting, and (iv) the forecasting period. Their results revealed that 90% out of the top nine models used in electricity load forecasting where artificial intelligence based, with Artificial Neural Networks (ANN) representing 28%. They also observed that ANN models were primarily used for short-term electricity load forecasting where electrical energy consumption are complicated.

Son (Son and Kim, 2020) proposed a LSTM model that can accurately forecast monthly residential electricity demand and compared its performance to four benchmark models: Support Vector Regression, Artificial Neural Network (ANN), Autoregressive Integrated Moving Average (ARIMA) and Multiple Linear Regression. The LSTM model showed a superior performance for MAPE by achieving the lowest MAPE value, of less or equal to 1%.

A comparative analysis of five commonly used short-term load forecasting techniques, i.e. Auto-Regressive Integrated Moving Average (ARIMA), Multiple Linear Regression (MLR), Recursive Partitioning Regression Trees with Bootstrap Aggregating (RPART+BAGGING), Conditional Inference Trees with Bootstrap Aggregating (CTREE+BAGGING), and Random Forest (RF) was performed by Kapoor (Kapoor and Sharma, 2018). On comparison of MAPE of all techniques, they concluded that the error associated with RF was least and this approach produced more accurate results. A comparative study by Kandananond (Kandananond, 2011), in which three methodologies, ARIMA, ANN and Multiple Linear Regression (MLR) were deployed to load forecasting in Thailand. The results showed that the ANN model reduced the MAPE to 0.996%, while those of ARIMA and MLR were 2.80981% and 3.2604527% respectively.

Rana (Rana and Koprinska, 2016) presented an approach for very short-term load forecasting called Advanced Wavelet Neural Networks (AWNN) which used a shift invariant advanced wavelet packet transform for load decomposition, Mutual Information for feature selection and a multi-layer perceptron Neural Network trained with Levenberg-Marquardt algorithm for prediction. They evaluated the performance of their AWNN model using two different datasets for two years: Australian 5-min data and Spanish 60-min data. They choose the different geographic location and time resolution to better access the robustness of their AWNN model. Using the two evaluation metrics MAE and MAPE their AWNN models outperformed a number of methods used for comparison: three ARIMA methods, three Holt-Winters exponential smoothing methods, an industry model, several naïve baselines, and also single non-wavelet Neural Networks, Linear Regression, and Model Tree Rule.

Wang (Wang et al., 2014) proposed a novel approach for short-term load forecasting by applying univariate wavelet denoising in a combined model that is a hybrid of the seasonal autorgressive integrated moving average (SARIMA) model and a back propagation neural networks (BPNN). Electricity load data from New South Wales, Australia was used to evaluate the performance of their proposed approach. They compare their combined model with the SARIMA, and BPNN and the results showed that their proposed model can effectively improve the forecasting accuracy.

Munem (Munem et al., 2020) proposed a multivariate Bayesian Optimization based Long short-term memory (LSTM) neural network to forecast the residential electric power load for the upcoming hour. Their model surpasses convolutional neural network (CNN), artificial neural network (ANN) and support vector machine (SVM) using MAE, RMSE and MSE (Mean Squared Error) as performance evaluation metrics. They also observed that though their model performed conspicuously it can be improved by adding feature selection technique with the model.

Overview. The main objective of this study is to achieve a MAPE of less than 1% on day ahead hourly load forecast of the Anomalous Days of ISO New England Grid Data. To this period no monogram in short term load forecasting has combined Multivariate Denoising Using Wavelet and Principal Component Analysis with Bayesian Optimization for model hypeparameter tuning and this work seeks to fill in that gap.

Section 2 briefly discusses Multivariate Denoising Using Wavelet and Principal Component Analysis, Bayesian Optimization for Hyperparameter Tuning, the three models used in this study Random Forest (RF), Feedforward Neural Networks (FFNN), and the Long Short Term Memory (LSTM) Neural Network.

Section 3 outlines the methodology followed in this study.

Section 4 outlines the experiments and findings of this study with regards to the objective specified above.

Section 5 concludes the paper and outlines future directions.

2 BACKGROUND

In this section, we briefly describe Multivariate Denoising Using Wavelet and Principal Component Analysis, Bayesian Optimization for model hypeparameter tuning, Random Forest, Feedforward Neural Network (FFNN) and Long Short-Term Memory (LSTM) Neural Network.

2.1 Multivariate Wavelet Denoising using Principal Component Analysis

The Multivariate Denoising Using Wavelet and Principal Component Analysis Scheme of Aminghafari (Aminghafari et al., 2006) is performed by the MATLAB R2020b function WMULDEN and outlined in Algorithm 1. The wavelet decomposes the time series producing approximation coefficients which describe the overall shape of the signal by capturing low frequency information and detail coefficients that describe finer local changes by capturing high frequency information. Among detail coefficients, low intensity ones correspond to the noisy part of the signal. Principal component analysis is then performed on the approximation coeffients to keep only the most important features of the signal.

Algorithm 1: Multivariate Denoising Using Wavelet and Principal Component Analysis (WPCA).

• **Parameters:** $J \in \mathbb{N}$.

1: Apply the level J wavelet decomposition of each column of X, where X is an $n \times p$ predictor matrix. This step produces $J+1$ matrices $D_1, \ldots, D_J$ containing the detail coefficients at level 1 to J of the p signals and the approximation coefficients A_J of the p signals. Matrices D_j, $1 \leq j \leq J$, and A_J are, respectively, of size $\frac{n}{2^j} \times p$ and $\frac{n}{2^J} \times p$.

2: State $\hat{\Sigma}_\varepsilon$ the estimator of Σ_ε, the noise covariance matrix, equivalent to the Minimum Covariance Determinant estimator (MCD) as $\hat{\Sigma}_\varepsilon = MCD(D_1)$ and then compute the Singular Value Decomposition (SVD) of $\hat{\Sigma}_\varepsilon$ providing an orthogonal matrix V such that $\hat{\Sigma}_\varepsilon = V\Lambda V^T$ where $\Lambda = \text{diagonal}(\lambda_i, 1 \leq i \leq p)$. Apply to each detail after change of basis (namely D_jV, $1 \leq j \leq J$), the p univariate thresholding strategies using the threshold $t_i = \sqrt{2\lambda_i \log(n)}$ for the ith column of D_jV.

3: Apply the PCA of the matrix A_J and then choose the convenient number p_{J+1} of principal components.

4: Rebuild the denoised matrix $\breve{X}$, from the simplified detail and approximation matrices, by changing of basis using V^T and inverting the wavelet transform.

2.2 Bayesian Optimization Algorithm

Machine learning modeling involves training a model M to minimize some predefined loss function $L(\mathbf{X}^{(val)}; M)$ on a given validation data set $\mathbf{X}^{(val)}$. Most common loss functions are the root mean squared error (RMSE) and mean squared error (MSE). The model M is constructed by a learning algorithm A using a training data set $\mathbf{X}^{(tr)}$. The learning algorithm A may itself be parameterized by a set of hyperparameters λ, for example $M = A(\mathbf{X}^{(tr)}; \lambda)$. The goal of hyperparameter search is to find a set of hyperparameters λ^* that yield an optimal model M^* (to

be used make forecast on an out of sample data set $\mathbf{X}^{(test)}$) which minimizes $L(\mathbf{X}^{(val)};M)$. Formally this is described as follows:

$$\lambda^* = \arg\min_{\lambda} L(\mathbf{X}^{(val)}; A(\mathbf{X}^{(tr)};\lambda)) \quad (1)$$

$$= \arg\min_{\lambda} F(\lambda; A, \mathbf{X}^{(tr)}, \mathbf{X}^{(val)}, L) \quad (2)$$

The objective function F takes a tuple of hyperparameters λ and returns the loss. The data sets $\mathbf{X}^{(tr)}$ and $\mathbf{X}^{(val)}$ are given and the learning algorithm A and loss function L are chosen (Claesen and Moor, 2015).

Bayesian optimization find λ^* by estimating F as a form of Gaussian Process. Bayesian optimization incorporates prior belief about F and updates the prior with samples drawn from F to get a posterior that better approximates F. Bayesian optimization also uses an acquisition function that directs sampling to areas where an improvement over the current best observation is likely. Let's define λ_i as the ith sample, and $F(\lambda_i)$ as the observation of the objective function at λ_i. As we accumulate observations $D_{1:t} = \{(\lambda_i, F(\lambda_i)), i = 1,2,\ldots,t\}$, the prior distribution is combined with the likelihood function $P(D_{1:t}|F)$ to obtain the posterior distribution:

$$P(F|D_{1:t}) \propto P(D_{1:t}|F)P(F) \quad (3)$$

The prior distribution $P(F)$ represents our belief about the space of possible objective functions and the posterior distribution captures our updated beliefs about the unknown objective function.

Algorithm 2: Bayesian Optimization Algorithm.

• **Select:** T, u

1: **for** $t = 1,2,\ldots,T$ **do**

2: Find λ_t by optimizing the acquisition function u over function F:

$$\lambda_t = \arg\max_{\lambda} u(x|D_{1:t-1})$$

3: Sample the objective function:

$$y_t = F(\lambda_t)$$

4: Augment the data:

$$D_{1:t} = \{D_{1:t-1}, (\lambda_t, y_t)\}$$

and update the posterior of function F.

5: **end for**

There are two main parts of Algorithm 2: Maximizing the acquisition function (step 2) and updating the posterior distribution (steps 3 and 4). The MATLAB R2020b function BAYESOPT is used to maximize the acquisition function and as for updating the posterior distribution BAYESOPT uses another MATLAB R2020b function FITRGP which fit a Gaussian process model to the data.

2.3 Random Forest (RF)

Random forests are indeed a generalization of bagging. Instead of considering all of the predictors at each split of the tree, only a random sample of "*NumPTS*"[1] predictors can be chosen each time. The main advantage of random forests with respect to bagging can be noticed in the case of correlated predictors, predictions from the bagged trees will be highly correlated so that bagging will not reduce the variance so much, whereas random forests overcome this problem by forcing each split to consider only a subset of the predictors. In the case of random forest, the efficiency of the method depends on a suitable selection of the number of trees n and the number of predictors *NumPTS* tested at each split. The out-of-bag (OOB) error can be used for searching a suitable n as well as a suitable *NumPTS*. As with bagging, random forests will not overfit if we increase n, so the goal is to choose a value that is sufficiently large.

2.4 Feedforward Neural Network (FFNN)

The FFNN architecture used in this study consists of three layers known as the input layer, hidden layer and output layer. The activation function used in the hidden and output layers is hyperbolic tangent sigmoid transfer function (*tansig*) and linear transfer function (*purelin*) respectively. During the training process, the input data (in the input layer) will be trained and weighted by the neurons in the hidden layer. Then, the estimated output from the training process will be compared with the desired target (in the output layer). The comparison is further evaluated based on the mean squared error (MSE). The training process is repeated by adjusting the weights and bias inside the neurons until the estimated output and the desired target is matched with minimum MSE. The main advantage of FFNN is the ability to implicitly learn and model the complex relationships between inputs and outputs.

[1] *NumPTS* = Number of Predictors to Sample

2.5 Long Short-Term Memory (LSTM) Neural Network

LSTM is a specific recurrent neural network (RNN) architecture that was designed to model temporal sequences and their long-range dependencies more accurately than conventional RNNs. The LSTM contains special units called memory blocks, in the recurrent hidden layer. The memory blocks contain memory cells with self-connections storing the temporal state of the network in addition to special multiplicative units called gates to control the flow of information (Sak et al., 2014). The LSTM memory cells are arranged sequentially to create a stable memory sequence which eliminates the vanishing gradient problem. LSTM have several hyperparameters that needs to be fine tuned, such as the number of hidden units in the LSTM layer, so as to learn the data well. In addition an Optimizer that updates the weights and biases of the LSTM Neural Network has hyperparameters such initial learning rate that needs tuning as well. According to Greff (Greff et al., 2017) the learning rate and network size are the most crucial tunable LSTM hyperparameters. For a detail discussion of LSTM Neural Network for load forecasting refer to Munem (Munem et al., 2020) and He (He et al., 2019)

3 METHODOLOGY

In this section we describe the Model evaluation metrics used in this study, Data, RF, FFNN, and LSTM training:

3.1 Model Evaluation Metrics used in This Study

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(y_j - \hat{y}_j\right)^2} \tag{4}$$

$$MAPE = 100 * \frac{1}{N}\sum_{j=1}^{N}\left|\frac{y_j - \hat{y}_j}{y_j}\right| \tag{5}$$

$$MAE = \frac{1}{N}\sum_{j=1}^{N}\left|y_j - \hat{y}_j\right| \tag{6}$$

- $RMSE$: Root Mean Square Error
- $MAPE$: Mean Absolute Percent Error
- MAE: Mean Absolute Error
- N: Total number of values
- y_j: Actual observed value to compare the forecast with
- $\hat{y_j}$: Forecast value, that is, the model output

3.2 Data

Real recorded hourly data of ISO New England grid from 2004 to 2009 is used in this study. The raw data files can be obtained directly from ISO New England (www.iso-ne.com). The power generating and distribution system of the entire New England Area is managed and operated by the ISO New England. New England is a region comprising six states of USA: Connecticut, Maine, Rhode Island, Vermont, Massachusetts and New Hampshire. A better description and analysis of the same data set used in this study is given in Raza (Raza et al., 2020). The predictor variable are:

- Dry bulb temperature
- Dew point temperature
- Hour of day
- Day of the week
- Holiday/weekend indicator (0 or 1)
- Previous 24-hr average load
- 24-hr lagged load
- 168-hr (previous week) lagged load

Denoising of the Predictor Variables. The denoising procedure is oulined in Algorithm 1:

- Wavelet Decomposition Parameters: The first step in Wavelet Denoising Scheme is the selection of the proper wavelet type, here we lected Symlets 4 and the next step is to select the level of decomposition, which is the number of times that the original signal is decomposed by the wavelet transform. Aminghafari (Aminghafari et al., 2006) proposed a decomposition maximum level of 8 and in this study we selected 5 levels. Both the selected wavelet and level are the default values in the MATLAB R2020b function WMULDEN. The choice being that with more decomposition we may remove more noise, however it is more likely that we may also remove valuable time series fluctuations.
- Denoising Parameters: The next step is to select a thresholding method. The intuition is that small wavelet coefficients are combined with noise, while large wavelet coefficients contain more useful signal than noise. Hence to remove those components with small coefficients and reduce the

influence of components with large coefficients we apply soft fixed form thresholding (default in WMULDEN). Fixed form thresholding is given by

$$\sqrt{2 \times \log(\text{length}(X'))}$$

where X' is $n \times 1$

- Principal Components Parameters: The Kaiser's rule (Aminghafari et al., 2006) define the way to select principal components for approximation at the chosen level in step 1 of Algorithm 1 in the wavelet domain and for final PCA after wavelet reconstruction by keeping the components associated with eigenvalues greater than the mean of all eigenvalues.

3.3 RF Training

The data set was divided as follows:

- Training - From 1 January 2004 to 31 December 2008
- Testing - From 1 January 2009 to 31 December 2009

It is noted that the performance of random forest is closely related to the strength of each tree and the inter-tree correlation. As *NumPTS* goes up, the strength of each tree can be improved, but the inter-tree correlation increases and the random forest error rate goes up. As *NumPTS* goes down, both inter-tree correlation and the strength of individual trees go down. So, the optimal value of *NumPTS* must be chosen carefully to make the trees as uncorrelated as possible. In addition, since the final regression result of algorithm is produced by all decision trees, the number of decision trees in the random forest will also affect the accuracy of the model.
Usually, the performance of single tree in the forest is low, so if the number of decision trees is too small, the whole random forest model will have bad performance. Hence, the hyperparameters *NumPTS* and *n* must be chosen carefully. Next, hyperparameters of random forest model are tuned by Bayesian optimization. We choose the number of decision trees in the random forest *n* and the size of the predictor variables subset *NumPTS* as hyperparameters. The ranges of value for hyperparameters are *n* in the range $[1,300]$ and *NumPTS*[2] in the range $[1,7]$, respectively. The model OOB error is chosen as the objective function.

3.4 FFNN Training

The data set was divided as follows:

- Training - From 1 January 2004 to 31 December 2006
- Validation - From 1 January 2007 to 31 December 2008
- Testing - From 1 January 2009 to 31 December 2009

The effectiveness of FFNN is influenced by algorithm and FFNN parameters such as momentum constant, learning rate and the number of neurons in hidden layers. In this study we utilize the TRAINBR (Bayesian regularization backpropagation) MATLAB R2020b function to train the network. Bayesian regularization takes place within the Levenberg-Marquardt (LM) algorithm. The LM algorithm offers a tradeoff between the benefits of the Gauss-Newton method and the steepest descent method. The LM algorithm updates the network weights using the Hessian matrix approximation:

$$w_{k+1} = w_k - \left[J^T + \mu I\right]^{-1} I^T e \tag{7}$$

where J is the Jacobian matrix (first-order derivatives of the errors), I is the identity unit matrix, e is the vector of the network errors, and μ the *Marquardt adjustment parameter*. The *Marquardt adjustment parameter* controls the algorithm. If $\mu = 0$ the algorithm behaves as in the Gauss-Newton's method. For high values of μ the algorithm uses the steepest descent method. We tune the hyperparameters *number of neurons* in the hidden layer as well as the *Marquardt adjustment parameter* using Bayesian Optimization to minimize the Mean Squared Error (MSE) of the validation set as the objective function. The ranges of value for hyperparameters are *number of neurons* in the range $[1,50]$ and *Marquardt adjustment parameter* in the range $\left[1 \times 10^{-3}, 1 \times 10^{10}\right]$, respectively.

3.5 LSTM Training

The data set was divided as follows:

- Training - From 1 January 2004 to 31 December 2006
- Validation - From 1 January 2007 to 31 December 2008
- Testing - From 1 January 2009 to 31 December 2009

Deep learning models are typically trained by a stochastic gradient descent optimizer. There are many variations of stochastic gradient descent; in this study, we utilize the Adam optimizer. The Adam parameter (weights and biases) update is calculated on the following equation:

[2]Number of predictors is 8

$$\theta_{n+1} = \theta_n - \frac{\alpha m_n}{\sqrt{v_n} + \varepsilon} \tag{8}$$

where

$$m_n = \beta_1 m_{n-1} (1 - \beta_1) \nabla E(\theta_n) \tag{9}$$

and

$$v_n = \beta_2 v_{n-1} + (1 - \beta_2) [\nabla E(\theta_n)]^2 \tag{10}$$

where

- n is the following steps of iterative process of training
- α is the learning rate
- θ the vector of trained parameters
- $E(\theta)$ is the loss function
- β_1 is the Gradient Decay Factor
- β_2 is the Squared Gradient Decay Factor

The learning rate tells the optimizer how far to move the weights in the direction opposite of the gradient for a mini-batch. If the learning rate is low, then training is more reliable, but optimization will take a lot of time because steps towards the minimum of the loss function are tiny. If the learning rate is high, then training may not converge or even diverge. Weight changes can be so big that the optimizer overshoots the minimum and makes the loss worse.

There are multiple ways to select a good starting point for the learning rate and in this study we let Bayesian Optimization select the best initial learn rate. In this study, we utilize one LSTM layer. And in addition to the initial learn rate, we also optimize the number of hidden units in the LSTM layer, using Bayesian Optimization. The number of hidden units corresponds to the amount of information remembered between time steps (the hidden state). The hidden state can contain information from all previous time steps, regardless of the sequence length. If the number of hidden units is too large, then the layer might overfit to the training data, to overcome this problem of overfitting we include a dropoutlayer after the LSTM layer. The range of values for the two hyperparameters are learning rate in the range $[1 \times 10^{-3}, 1]$ and number of hidden units in the range $[1, 200]$, respectively.

4 EXPERIMENTS RESULTS AND ANALYSIS

4.1 Models Hyperparameters Values Selected by Bayesian Optimization Algorithm

Table 1: RF Model.

Hyperparameters	Selected Hyperparameters Values
n (Tree Size)	200
NumPTS (No. of points to sample)	1

Table 2: FFNN Model.

Hyperparameters	Selected Hyperparameters Values
number of neurons in the hidden layer	25
Marquardt adjustment parameter	3753.8

Table 3: LSTM Model.

Hyperparameters	Selected Hyperparameters Values
No. of hidden units in the LSTM layer	186
Initial learn rate	0.0046407

4.2 Anomalous Days Load Prediction

The selected anomalous days are Christmas Day, (Friday, December 25, 2009), Memorial Day (Monday, May 25, 2009), Easter Day (Sunday, April 12, 2009), and Labor Day (Monday, September 7, 2009).

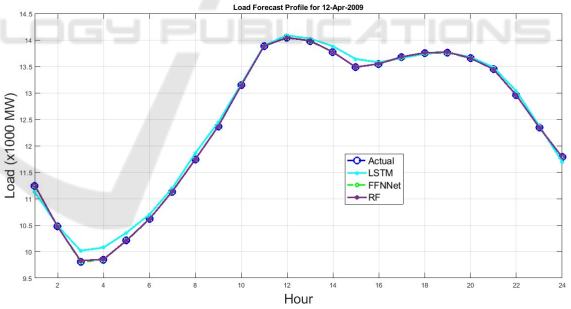

Figure 1: Day ahead hourly load forecast of Easter Day.

Table 4: MAPE, MAE and RMSE for Easter Day 2009.

Model	Out of Sample		
	MAPE (%)	MAE (MWh)	RMSE (MWh)
LSTM	0.65	74.66	97.07
FFNN	6.24×10^{-7}	7.37×10^{-5}	8.68×10^{-5}
RF	0.045	5.20	7.03

The x-axis of the graph represents the hours of the anomalous day and the y-axis represents the load demand in megawatt. From the graphs and outlined in the tables of Model evaluation metrics, the Feedforward Neural Network (FFNN) is a better fit to the data followed by the Random Forest (RF) and last is the

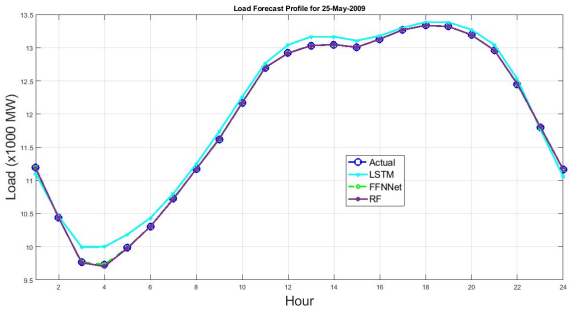

Figure 2: Day ahead hourly load forecast of Memorial Day 2009.

Table 5: MAPE, MAE and RMSE for Memorial Day 2009.

Model	Out of Sample		
	MAPE (%)	MAE (MWh)	RMSE (MWh)
LSTM	0.89	101.27	117.19
FFNN	6.00×10^{-7}	6.52×10^{-5}	9.23×10^{-5}
RF	0.042	4.62	7.48

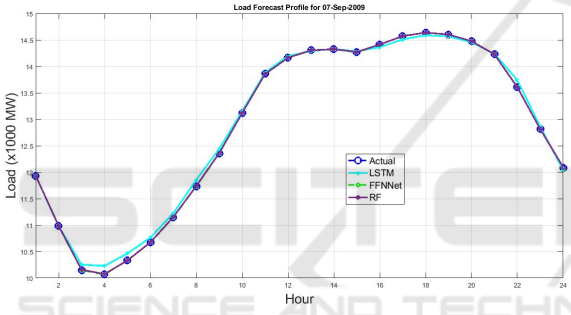

Figure 3: Day ahead hourly load forecast of Labour Day 2009.

Table 6: MAPE, MAE and RMSE for Labour Day 2009.

Model	Out of Sample		
	MAPE (%)	MAE (MWh)	RMSE (MWh)
LSTM	0.48	56.42	72.78
FFNN	4.42×10^{-7}	5.51×10^{-5}	6.10×10^{-5}
RF	0.032	3.86	4.94

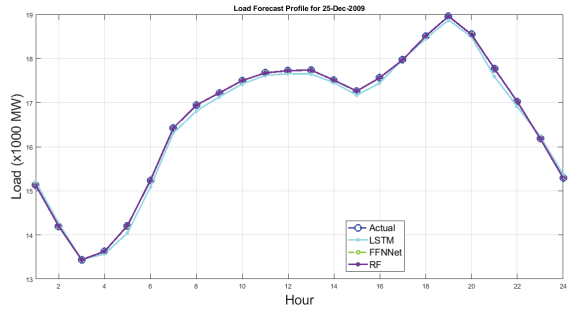

Figure 4: Day ahead hourly load forecast of Christmas Day 2009.

Table 7: MAPE, MAE and RMSE for Christmas Day 2009.

Model	Out of Sample		
	MAPE (%)	MAE (MWh)	RMSE (MWh)
LSTM	0.52	85.27	95.61
FFNN	3.73×10^{-7}	6.17×10^{-5}	7.04×10^{-5}
RF	0.014	2.33	2.68

Long Short Term Memory (LSTM) Neural Network. The MAPE, MAE and RMSE from Table 4 to Table 7 show that the FFNN has achieved the lowest errors and hence the highest accuracy in predicting the load profile of the selected anomalous days. The predicted load demand for the FFNN model is very close to the actual load. The performance of the FFNN model implies that with proper data preprocessing and excellent hyperparameter optimization shows that the FFNN is capable of approximating any measurable function to any desired degree of accuracy. The problem of overfitting was avoided during training by the use of MATLAB R2020b function TRAINBR which uses Bayesian regularization and enhances generalization as well. As the second best performing model the Random Forest is however the one most prefered the reason being their flexibility which is suitable for linear and non-linear relationships, they take into account interactions among predictors at different levels, no assumption on the data are needed, they provide very accurate predictions as evident in this study and their interpretability. As the least performing model, the LSTM model however can be improved by adding several LSTM layers, optimizing hyperparameters such as Gradient Decay Factor and the Squared Gradient Decay Factor of the Adam Optimizer

5 CONCLUSIONS

The paper presented three multivariate Bayesian Optimization (BO) based Random Forest (RF), Feedforward Neural Networks (FFNN) and Long Short-term Memory (LSTM) neural network for day ahead hourly load forecast of the anomalous days system load of the ISO New England grid. The predictor variables were subjected to Multivariate Denoising using Wavelet and Principal Component Analysis. The methodology followed shows that the Feedforward Neural Networks achieved superior results. Also, the main objective of this study was attained which was to achieve a MAPE of less than 1% on the day ahead hourly load forecast of the Anomalous Days of ISO New England Grid Data. The major challenge going forward is to apply our methodology to different grid data sets.

ACKNOWLEDGEMENTS

I would like to acknowledge the Faculty of Engineering and the Built Environment at the Durban University of Technology for their financial support.

REFERENCES

Aminghafari, M., Cheze, N., and Poggi, J.-M. (2006). Multivariate de-noising using wavelets and principal component analysis. In *Computational Statistics and Data Analysis, 50, pp. 2381–2398*. ELSEVIER.

Claesen, M. and Moor, B. D. (2015). Hyperparameter search in machine learning. In *arXiv preprint arXiv:1502.02127*. CORNELL UNIVERSITY.

Greff, K., Shrivastava, R. K., Koutnik, J., Steunebrink, B. R., and Schmidhuber, J. (2017). Lstm: A search space odyssey. In *IEEE Transactions on Neural Networks and Learning Systems, vol. 28, issue 10, pp. 2222–2232*. IEEEXplore.

He, F., Zhou, J., Feng, Z., Liu, G., and Yang, Y. (2019). A hybrid short-term load forecasting model based on variational mode decomposition and long short-term memory networks considering relevant factors with bayesian optimization algorithm. In *Applied Energy, 237, pp. 103–116*. SCIENCEDIRECT.

Hobbs, B. F., Jitprapaikulsarn, S., Konda, S., Chankong, V., Loparo, K. A., and Maratukulam, D. J. (1999). Analysis of the value for unit commitment of improved load forecasts. In *IEEE Transactions on Power Systems, vol. 14, no. 4, pp. 1342–1348*. IEEEXplore.

Kandananond, K. (2011). Forecating electricity demand in thailand with an artificial neural network approach. In *Energies, 4(8), 1246–1257*. MDPI.

Kapoor, A. and Sharma, A. (2018). A comparison of short-term load forecasting techniques. In *2018 IEEE Innovative Smart Grid Technologies - Asia (ISGT Asia), Singapore, pp. 1189–1194*. UIASSIST.ORG.

Munem, M., Bashar, T. M. R., Roni, M. H., Shahriar, M., Shawkat, T. B., and Rahaman, H. (2020). Electric power load forecasting based on multivariate long short-term memory neural network using bayesian optimization. In *2020 IEEE Electric Power and Energy Conference (EPEC), Edmonton, AB, Canada, pp. 1–6*. IEEEXplore.

Nti, I. K., Teimeh, M., Nyarko, O., and Adekoya, A. F. (2020). Electricity load forecasting: A systematic review. In *Journal of Electrical Systems and Information Technology 7, 13*. Springer.

Rana, M. and Koprinska, I. (2016). Forecasting electricity load with advanced wavelet neural networks. In *Neurocomputing, 182, pp. 118–132*. ELSEVIER.

Raza, M. Q., Mithulananthan, N., Li, J., and Lee, K. Y. (2020). Multivariate ensemble forecast framework for demand prediction of anomalous days. In *IEEE Transactions on Sustainable Energy, vol. 11, no. 1, pp. 27–36*. IEEEXplore.

Sak, H., Senior, A., and Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *In Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*. RESEARCHGOOGLE.

Son, H. and Kim, C. (2020). A deep learning approach to forecasting monthly demand for residential-sector electricity. In *Sustainability, 12, 3103*. MDPI.

Wang, J., Wang, J., Li, Y., Zhu, S., and Zhao, J. (2014). Techniques of applying wavelet de-noising into a combined model for short-term load forecasting. In *Electrical Power and Energy Systems, 62, pp. 816–824*. ELSEVIER.