A Comparative Analysis Between LSTM and LSTM Ensembled Strategy for Short-term Load Forecasting

This paper was downloaded from TechRxiv (<https://www.techrxiv.org>).

LICENSE

CC BY 4.0

SUBMISSION DATE / POSTED DATE

13-10-2021 / 19-10-2021

CITATION

Fattah, Fathun; Mojumder, Pritom; Fuad, Azmol Ahmed; Ahmad, Mohiuddin; hossain, Eklas (2021): A Comparative Analysis Between LSTM and LSTM Ensembled Strategy for Short-term Load Forecasting. TechRxiv. Preprint.

https://www.techrxiv.org/articles/preprint/A_Comparative_Analysis_Between_LSTM_and_LSTM_Ensembled_Short-term_Load_Forecasting/16807213

DOI

[10.36227/techrxiv.16807213.v1](https://doi.org/10.36227/techrxiv.16807213.v1)

A Comparative Analysis Between LSTM and LSTM Ensembled Strategy for Short-term Load Forecasting

Fathun Karim¹, *Member, IEEE*, Pritom Mojumder², Azmol Ahmed Fuad³, Mohiuddin Ahmad⁴, *Member, IEEE*, Eklas Hossain⁵, *Senior Member, IEEE*

Abstract—This paper aims at assessing the potential of Long-short term memory (LSTM) network and LSTM ensembled network into carrying out short term load forecasts and observe their relative performances in terms of their forecast precision. The experiment shows that, LSTM ensembled networks work substantially better as compared to LSTM only networks. An improvement of 0.04% for hour-ahead forecasts and 0.14% improvement for the day ahead forecasts had been observed for LSTM ensembled approach. For an hour ahead forecast both networks generated the least errors which were found to be hovering below the decimal point. But it moved up to the single digit domain, more precisely around a 2.377% increase for the ensembled approach and a 2.47% increase for LSTM only approach in case of day-ahead forecasts. Applying either LSTM or LSTM ensembled techniques as laid out in this paper can save utilities hundreds of thousands of dollars as compared to most of the other contemporary neural network techniques, but the significant error reduction potential of LSTM ensembled network can play an even better role for utilities since it carries around the potential to save 224000 dollars for day ahead forecasts and also 64000 dollars for hour ahead forecasts.

Index Terms—STLF, LSTM network, Ensemble learning, Bagging, Hourly Load, Daily forecasting

I. INTRODUCTION

LOAD Forecasting is an essential component for utility companies to manage the production and distribution of electric power in the most convenient and cost-efficient way possible. A better forecast model can save the utilities a lot of money and also help the consumers get an interruption-free low-cost energy supply. Short term load forecasting (STLF) is essential for maintaining appropriate coordination between the basic generation scheduling functions, providing insights on the security condition of the whole system at any particular point, and also for presenting the system dispatcher with appropriate and timely information with regards to the load behavior [1]. The fact that load pattern between short intervals appears to be pretty erratic is what makes a highly accurate forecast quite difficult to achieve. But the benefits of a precise forecast is worth all the effort, considering the economic and security advantages, which are one of the driving motivations behind the large volume of researches being quite regularly produced in this field.

Short-term load forecasting had been approached through several techniques. Statistical models had largely been used for this purpose, for example, Multiple Linear Regression, Stochastic Time Series, General Exponential Smoothing, State

Space Method, etc. [2] with some handful of exceptions, for example, knowledge-based algorithmic approach [3] and ALFA approach [4]. Among other statistical models most popularly seen in the literature are Autoregressive Moving Average (ARMA) [5], Autoregressive Integrated Moving Average (ARIMA) [6], [7], General Exponential Smoothing [8], Holt-Winters smoothing approach [9], [10], Periodic AR modelling [10], Kalman Filtering [11], [12], Box Jenkins Method [13], [14], Autoregressive Moving Average with Exogenous Inputs (ARMAX) [15], [16]. Despite such a wide pool of statistical alternatives, linear regressions methods were seen to be the most captivating choice because of their relative simplicity and low trade-offs with a reasonable performance [17], [18], [19]. But one problem with linear and time-series models is that they perform poorly while capturing the nonlinear nature of the load due to a variety of exogenous factors.

To tackle the challenge of reconciling the nonlinear nature of the load, different methods were experimented with to discern better forecasting techniques. Fuzzy logic systems and their different variants were tested since they had the ability to cover up the drawbacks of rigid mathematical models which are bound within a stringent rule, thereby allowing them the capability to accommodate sporadic fluctuations, unforeseen loads, mechanical uncertainties, transmission losses, voltage instability, etc. Fuzzy approaches were observed to perform with a relatively lower, single-digit error compared to the conventional approaches [20], [21], [22]. But fuzzy error fluctuations had been observed to behave quite irregularly when mapped throughout a whole day [22]. With the advent of machine learning technologies, a new domain appeared for carrying out reliable load forecasts. Among such techniques are support vector machine [23], [24], [25], [26], [27], extreme learning machine [23], [28], support vector regression [29], random forest models [30], extreme gradient boosting (XGBoost) [31]. While support vector models were seen to perform relatively better as compared to other conventional models [32], one drawback associated with such technique had been its inability to perform with similar efficiency on large scale datasets, because SVM complexity is directly dependent upon the size of the dataset. E. aguiler et al. have shown the superiority of XGBoost over SVR and RF methods in carrying out more precise forecasts [31]. But XGBoost methods should be utilized for datasets with an abundance of features and less noise, otherwise, it may cause the model to underperform. Random Forest models are computationally time-consuming

and do not outperform XGBoost models concerning the accuracy of the forecast, as shown by E. aguiler et al. [31]. Neural Networks, since the beginning of their inception, have always lured researchers into accommodating their potential into the field of load forecasting due to it having better learning potential and with time many novel variants emerging with time with even better learning capacity, the prospects become even more promising. Hippert et. Al has reviewed a wide set of neural network models that had been implemented for short-term load forecasting purposes, and they identified some common recurring issues in such models, such as overfitting and overparameterizing [33]. ANN was first seen to be in use in the early days of machine learning [34]. Since then, with further developments in the field of neural networks, many different variants and approaches were experimented with such as radial basis function neural networks [35], wavelet neural network approaches [36], neural networks with genetic algorithms executed with backpropagation [37], neural networks with particular swarm optimization [38], evolutionary ELM [39] etc. Many hybrid approaches were explored as well to augment the efficiency of the machine learning and neural network models. Mamun et al. have done an extensive review of all available literature on the hybrid ML approach towards STLTF [40]. Their work exhibited the superiority of hybrid SVM and ANN models for better accuracy compared to other hybrid models. From the ANN hybrids (ANN-FA, ANN-FOA, ANN-GA, ANN-CT, ANN-NFIS, ANN-WT, ANN-AIS, ANN-PSO), they had found out ANN-NFIS hybrid having the capability to provide the highest accuracy. But it suffered from gradient descent problem, longer runtime, and lacked real-time response. From SVM hybrids (SVM-FA, SVM-GA, SVM-FOA, SVM-PSO, SVM-ABC, SVM-SA), they had observed SVM-ABC be able to provide better forecasts results with the least error, but the problem associated with this technique was its slow convergence speed. Mayur et al. have introduced a different SVM hybrid (SVM-GOA) which was not included in the assessment of Mamun et. al [41]. It has shown its relative superiority over other hybrid methods, for example, SVM-GA and SVM-PSO.

Following the breakthrough research on deep learning by Hinton and Lecun [42], researchers dived into the world of deep learning neural networks to tinker with these novel techniques and check their viability for STLTF purposes. From the many methods of Deep learning networks that are available, one popular technique was to apply convolutional neural networks [43], [44], [45], but CNN was mostly seen to be used as a hybrid model with other methods, for example, long short-term memory networks [46], [47], [48], [49]. Bendaoud et al. has observed CNN models to perform better than other ML trends such as ANN, SVM, RF, GBRT (Gradient Boosting Regression Trees) for short-term forecasts. Feed Forward deep neural network [50], Deep neural network with Restricted Boltzman Machines (RBM) pretraining [51], [52], RBM with Elman Neural Network [53], etc. have been tried as well. Feed Forward networks and RBM pretraining methods were found to have augmented the efficiency of the deep neural networks. But these models become harder to train with an increasing number of layers, hence the restriction

of layer numbers hobbles the efficiency of the models. Deep recurrent neural networks were found to be more effective than feedforward networks [50] and hence they have been widely studied throughout the literature [54]. RNN was found useful for learning highly nonlinear relationships and also to learn shared uncertainties. But it had an issue of overfitting and layer constraints, which were later attempted to mitigate by a novel pooling method using PDRNN, as proposed by Heng Shi et. Al. [55]. The shortcomings of deep learning models were further addressed by incorporating residual neural networks instead of stacking multiple hidden layers between the input and output [56]. RNN techniques also saw decent improvement when coupled with input attention mechanisms and hidden connection mechanisms [57].

Using RNN for forecasting had a different concern which is the vanishing gradient problem. A little tweak into the RNN architecture, through the addition of a memory unit called LSTM, was introduced to address this problem. Incorporating LSTM units have been shown to significantly increase the gap length for RNN backpropagation without having to fear the long-term gradients vanishing away, since LSTM units allow long-term gradients to propagate unchanged. LSTM networks have not been left unexplored in the literature [58], [59], [60]. W. kong has shown LSTM networks to perform better than other neural network models [60]. Like other standard statistical and machine learning models, LSTM models have also been coupled with various other models and were experimented with for STLTF purposes. Some of those hybrid models include CNN-LSTM [46], [48], MLR-LSTM [61], EMD-LSTM [62], and Resnet-LSTM [63], all of which have proven to significantly improve the error factor in the forecasts of standalone LSTM models.

It is fairly easy to achieve a MAPE value within 10% with any kind of neural network approach, but on the other hand, dropping the error further below is found to be equally challenging [33]. Being able to drop the prediction error even as little as 1% has a significant impact on the economic aspect of the whole power system. It has been found out that for a 10,000 MW utility, a 1% drop in forecasting error can save the company up to 1.6 million dollars annually [64]. This paper aims at testing LSTM networks and LSTM Ensembled networks for STLTF purposes and assess their relative performance with respect to each another. Although LSTM networks had been previously tested throughout literature and in some cases its superiority over other neural networks having been established, but so far the effect of LSTM ensembled networks had not been established yet, a gap which this paper aims to fill. We try to test if LSTM ensembled network can improve the precision of forecast compared to LSTM only network and if so, then to quantify that effect. The rest of the paper has been organized as follows, Section II discusses some of the fundamentals of the networks, section III discusses the methodology, section IV discusses the results and discussion and section V concludes the paper.

II. LSTM & LSTM NETWORK ENSEMBLE TECHNIQUE

A. LSTM

Long Short-Term Memory is a variant of recurrent neural network proposed by Hochreiter & Schmidhuber [65]. It tackles the complications arising from the long-term dependencies of RNN.

The chain structure of LSTM contains four neural networks and different cells which are basically memory blocks. The networks used have a single input layer, a single hidden layer, and one output layer. The network uses cells for storage of information while the gates are used to perform memory manipulations.

As stated by Gers et.al. [66] the memory block can be referred to be the basic unit of the hidden layer of an LSTM network. At the core, these memory cells have a recurrently self-connected linear unit called “Constant Error Carousel” or CEC. The activation of these CEC is called cell state. When activation is around zero the irrelevant noise and inputs are blocked from entering the cell keeping the cell state pristine.

As each of these memory cells is self-looping they contain temporal info in their cell states. The information flow through the network is done by manipulating the cell state (i.e., writing, erasing, and reading).

Three gates are used to achieve cell manipulation respectively. The entire operation of the LSTM can be elucidated using few mathematical equations.

i) Forget Gate: Forget gate is utilized for removal of the information that is no longer useful in the cell state. This gate is described by the following equation.

$$forget_{gate} = \sigma(in_{[t]}W_{forgetint} + out_{[t-1]}W_{forgetcell} + b_{forget}) \quad (1)$$

Where $forget_{gate}$, $in_{[t]}$, $out_{[t-1]}$ denotes the forget gate activation, input state and previous output state vector respectively. $W_{forgetint}$ and $W_{forgetcell}$ are the weight matrices for forget gate to intermediate state and forget gate to cell vector respectively. The term b denotes individual bias vector for each of the gates. σ denotes the logistic sigmoid function.

ii) Input Gate: Input gate is applied for the addition of necessary information to the cell. The input gates operation can be denoted using the following equation:

$$input_{gate} = \sigma(in_{[t]}W_{inputint} + out_{[t-1]}W_{inputcell} + b_{in}) \quad (2)$$

Where σ , $in_{[t]}$, $out_{[t-1]}$ and b bear their previously stated meanings. The $input_{gate}$ denotes the activation state vector of the input gate. $W_{inputint}$ and $W_{inputcell}$ are the weight matrices for input gate to intermediate state and input gate to cell vector individually.

iii) Output Gate: The task of the output gate is extracting useful information from the current cell state and this information is presented in the form of the output. The output gate equation is denoted by the following equation,

$$output_{gate} = \sigma(in_{[t]}W_{outputint} + out_{[t-1]}W_{outputcell} + b_{output}) \quad (3)$$

Where the term $output_{gate}$ denotes the output gate activation vector. $W_{outputint}$ and $W_{outputcell}$ are the weight matrices for output gate to intermediate state and output gate to cell vector chronologically. Apart from the equations describing the working of the three gates the LSTM network memory cell has 3 more equations which are given below:

$$up = \tanh(in_{[t]}W_{upint} + out_{[t-1]}W_{upcell} + b_{up}) \quad (4)$$

$$int_{[t]} = int_{[t-1]}forget_{gate} + input_{gate}up \quad (5)$$

$$out_{[t]} = output_{gate}\tanh(up) \quad (6)$$

Where equations 4 through 6 represent the update signal, value of the state at time step t and output of the cell respectively. W_{upint} and W_{upcell} are the weight matrices for update to intermediate state and update to cell vectors.

B. LSTM Network Ensemble Technique

The fabrication of artificial neural networks ensemble technique is achieved by training multiple models and combining them to produce a better output. Due to its very nature of ensembling multiple outputs, it is more natural for an ensemble of models to perform better than any individual model because of the various errors of the models getting “averaged out.”

In an ensemble learning method, time forecasting is improved by combining multiple forecasts (predictions) generated through a number of diverse machine learning models. In such a model, a bunch of distinct networks are employed to train a certain dataset with the intention of utilizing all their relative advantages and henceforth minimizing the overall prediction error. One of the benefits for the ensembled approach is that, the parameters do not have to be heavily optimized in accordance with one single network, hence relieving the system from being overparameterized. Dietterich has categorized the benefits for using ensembling technique into three types, statistical, computational and representational [67]. In our particular study data is abundant, our goal is to achieve statistical advantages due to its ability to smooth out data and feature shortbacks. To bypass computational complexities with regards to time and resource and also to bypass the diminishing returns of performance, the number of ensemble networks are kept at an optimal level, usually around 3 or 5 and in some cases around 10 depending on the context and the neural networks which are to be ensembled. LSTM ensembled approach uses a diverse number of LSTM networks which are distinct from each other in terms of the attributes they incorporate, for example, the networks kernel initializers, optimizers, activation functions etc. An ensemble approach is deemed eligible if different networks in the ensemble produce different outputs.

III. METHODOLOGY

Prior to designing the LSTM and LSTM ensembled networks, a public dataset had been collected from ISO North England. The networks were designed following standard

Fig. 1: A typical lstm ensembled network with different activation functions and stacking generalization

procedures before being fed with the dataset. This dataset included the hourly load information of the north american region starting from 1/3/2003 to 31/12/2014.

Five parameters were passed on the LSTM Module from the given dataset. Month, Day, Weekday, Hourly Demand, and Temperature were taken as input parameters. Season data, another feature was added alongside the input parameters. Unlike the rest of the parameters, the season data has been calculated using one-hot encoding, instead of fetching them from other sources. This work predicts both hourly and a day ahead forecast. The total work can be generalized using the following block diagram in Figure 1.

Fig. 2: Generalized Block Diagram of the Work

A. Hourly Forecasting

1) *Data Processing*: For hourly load forecasting, a set of features were accommodated into the sample. One of the features were the inclusion of load data from the previous six hours. The intuition behind this approach was to help the model capture the load consumption trend even more efficiently. Since load consumption patterns were dependent on the timeframe of the day, generating a unit pattern for a day that had a near orderly repetition while traversing throughout the week, hence we took an approach to take the base pattern of a day into account and feed the neural network with the

load trend of the past six hours to help the model capture the course of the load curve of the timeframe that the forecast hour falls into.

Features	Details	Size
M	Month	12
W	Weekday	1
H	Hour	1
S	Season	4
T	Temperature	1
F	Festival	1
prev_hour_x	Previous x^{th} hour demand	6

TABLE I: Details of the features passed onto the network

The dataset was randomly split into three subsets before being fed into the prediction model. The split ratio was allocated as 64% for training data, 16% for validation data, and 20% for test data.

One hot code had been applied for season data, festival data, and weekday data. Weekend data had been incorporated into weekday data, hence additional block was avoided from being taken. These data are aimed at helping the model to assess the time series pattern more precisely, since all of these factors have a significant influence on the nature of load consumption alongside other meteorological factors, for example, temperature. Temperature data had been provided as input without any further manipulation or normalization. The following table shows the naming of the data types, their data legend, and their size.

2) *LSTM Model Architecture*: The dataset was then fed into the LSTM network with samples containing 26 features each. For initial samples, the previous load demand parameter had been set to null. The LSTM network consisted of 6 layers. The first 2 layers were comprised of regular LSTM layers, while the latter 4 were picked to be dense layers. The activation function chosen for the output layer was SELU (Scaled Exponential Linear Unit), for its unique normalizing

and faster convergence property and also because it rids the network from concerns of vanishing and exploding gradients. The following table shows the different stages of the network, their outputs, and some parameters associated with each of these stages.

Layer type	Output shape	Parameter
input_3	[None, 1, 26]	0
lstm_10	[None, 1, 20]	3760
lstm_11	[None, 20]	3280
dense_22	[None, 20]	420
dense_23	[None, 20]	420
dense_24	[None, 20]	420
dense_25	[None, 20]	420
dense_26	[None, 1]	21

TABLE II: LSTM model architecture details in tabulated form

A total of 300 epochs were chosen for training the dataset. The number of epochs was determined by observing the convergence trend of the loss function. Beyond 300 epochs the error was not improving any further which indicated that any further epoch extension would have caused the network to suffer from redundancy.

3) *LSTM Ensemble Model Architecture*: The dataset was run through an LSTM ensembled Network parallelly to assess if ensembled strategies can yield a better forecast modeling for LSTM networks. Generally, ensembled strategies can assist in enhancing the performance of a neural network for prediction modeling. Such strategy is to work with different models ensembled together to work towards generating further output efficiency. Each model comes with its own sets of advantages and disadvantages. The idea behind the ensembling is that different models will learn the input features differently and thus can achieve a better generalization capability for producing a relatively better output. For an ensemble approach, we have used four different LSTM variants with four different activation functions and concatenated them to produce an output. We have used four exponential linear units, ELU, SELU, GELU, and a softplus activation function. Chen k. opined that RELU would have been a preferable activation function to utilize, but since it comes with the drawback of dead RELU problem, hence they have resorted to a modified RELU function, i.e. PRELU, whereas we have used ELU as an alternative. ELU addresses the dead RELU problem along with providing a better noise-robust deactivation state compared to modified RELUs like PRELU or LRELU. SELU too was picked a similar goal used to address dead RELU issues while also providing a self-normalizing feature. SELUs are seen to perform at a faster pace compared to most other activation functions and both vanishing and exploding gradient problems are virtually impossible in this case. GELU was used because it was a fairly new model being used in transformer models like GPT-3 and others, which has been shown to perform relatively better in terms of nonlinearity to other activation functions especially ELU and other RELU variants. GELU also is capable of being differential to all values for the input. And finally, the softplus function was used to make further

smoother approximations. For ELU and SELU variants, we have used glorot_uniform as kernel initializer and for the latter two, GELU and Softplus variants, random_normal had been used. The four different LSTM variants worked in tandem producing individual outputs which were then passed into a bagging process yielding our final output. Bagging or bootstrap aggregating approach was adopted instead of other ensemble algorithms because it can work with minimal complexity without having to compromise its prediction performance [68]. Bagging process takes in the initial forecast outputs and aggregates them to achieve a final prediction. Thus error differences thus get averaged out. After the bagging process, the aggregated output was then channeled through a dense output layer block to receive the final forecast.

B. Day Ahead Forecast

1) *Data Processing*: We further extended our work to check how efficiently both of the models work in terms of day-ahead load forecast. Since STLFL approaches include day-ahead load prediction as well as hour-ahead load prediction hence a fair assessment of the performance of both of the prediction models was deemed convenient for checking the resilience of the system. The prediction networks designed previously for the hour ahead forecast were tweaked to generate day ahead forecast instead of hour ahead forecast. For the hour ahead forecast we had only one output, but through this approach, we would be expecting 24 forecast outputs. Since our output size had increased hence we had to increase the number of features in our input sample to suit the model. For day-ahead forecasts, we had dropped the previous 6 hours demand from the input sample which we previously included for an hour ahead forecasts to capture hourly trends, and instead added the previous seven days demand set into the sample to capture daily trends. We have also included previous month load data into the sample to provide the model with even more trend cues. For initial samples with no previous seven-day demand data, a null value was passed. Season data, festival data, weekend/weekday data had all been derived similarly to the hour ahead forecast model. Month and season data had been hot encoded. The total size feature size in the sample ended up being 258 after completion of data processing. The following table shows the user data legend and their amount for the day ahead forecast portion of the work.

The total feature size in the sample ended up being 258 after the completion of data processing. The dataset was divided into three fragments. 64% for training, 16% for testing, and 20% for validation. The split was done randomly to avoid bias.

2) *LSTM model*: The dataset was then passed into the LSTM model without any further processing. The LSTM network contained 6 hidden layers. The first two layers were kept as regular LSTM layers and the subsequent layers were kept as Dense Layers. The output was a dense layer having an activation function SELU. A total of 1200 epoch was adopted for training due to the model reaching convergence at that epoch level. Taking further epochs would have caused redundancy. The following table shows the system details in a tabulated form mentioning the layer names, their associated

Features	Details	Size
M	Month	12
W	Weekday	1
S	Season	4
F	Festival	1
dh_x	Demand of h_th hour of the day	24
th_x	Temperature of the h_th hour of the day	24
prev_Dn_dh_x	Demand of the h_th hour of the previous n_th day	168
prevM_dh_x	demand of the h_th hour of the previous month	24

TABLE III: Features and their corresponding details for day ahead forecasting

output shapes, and the related parameters to each of these layers.

Layer type	Output shape	Parameter
input_1	[(None, 1, 258)]	0
lstm	[(None, 1, 250)]	367200
lstm_1	[(None, 200)]	320800
dense	[(None, 200)]	40200
dense_1	[(None, 200)]	40200
dense_2	[(None,200)]	40200
dense_3	[(None, 200)]	40200
dropout	[(None,200)]	0
dense_4	[(None, 24)]	4824

TABLE IV: Table showing the different layers, associated outputs, and parameters in a day-ahead forecast using LSTM.

3) *LSTM Ensembled Model*: For the ensembled approach, we proceeded similarly as before, taking four different LSTM variants and concatenating them to produce an output. The LSTM variants were created with four different activation functions, ELU, SELU, GELU, and Softmax just as it had been done for an hour ahead forecast. The kernel initializers had been adopted similarly too, which is glorot_uniform initializer for ELU and SELU, and random_normal for GELU and Softmax. Similar to the LSTM approach, 1200 epochs were chosen for the ensembled approach as well, to avoid epoch redundancy

IV. RESULTS AND DISCUSSIONS

A. Hour ahead forecast

The LSTM model took approximately 57.72 minutes to be trained with the availed dataset. The loss curve was observed to converge at around 300 epochs. The correspondence between the training and validation loss curve were seen to be fairly high. The model generated a MAPE value of 0.6130 for an hour ahead forecast after complete convergence of its loss functions.

For the LSTM Ensemble method, the total training period for the dataset was found to be around 87.87 minutes, which

Fig. 3: Training Loss Comparison for LSTM and LSTM Ensembled Network

Fig. 4: Validation loss comparison for LSTM and LSTM Ensembled Network

is slightly longer than the LSTM method due to four different networks working in tandem. The epoch were kept at 300 for this case too, since extending epochs were causing no significant improvement. The convergence and loss correspondence pattern found here did not vary much from the LSTM approach. The model produced an average MAPE of 0.5723 which is 0.04% lower than the LSTM only method.

Fig. 3 and Fig. 4 show the comparison between training and loss validation between LSTM and LSTM ensemble approaches.

Both the LSTM method and the LSTM Ensembled method yielded quite similar results with similar loss patterns. No radical difference was observed between the outputs of both of the models. Although, from a comparative point of view, the LSTM Ensembled approach appeared to have a relatively smoother and faster error convergence, as can be viewed from the loss figures..

A certain timeframe had been taken as a test sample to assess the performance of the models. The objective was to observe the contrast between the forecast performances of both of the models. With this purpose in mind, the actual load curve and the forecasted load curve had been mapped and juxtaposed for that exact timeframe in order to visualize the resilience of the models. The curves displayed in Fig. 5 and Fig. 6 exhibit a significant accuracy level for both of the models.

LSTM ensembled approach had been observed to contain

Fig. 5: Actual vs Predicted load for July 2, 2014 with LSTM approach

Fig. 7: 95% prediction interval for a day in January 2014 for hour ahead forecast.

Fig. 6: Actual vs Predicted load for July 2, 2014 with LSTM ensemble approach

Fig. 8: Comparison of training loss between LSTM and LSTM ensemble approach for day ahead forecast

greater correspondence between actual and predicted load. Both models contained slight deviations, especially in transition regions, and sudden bends. The deviations were mostly observed in the peak hour region, but for off-peak hours the predicted and the actual curves were virtually indistinguishable.

From the observation, the consistency was observed to be maintained fairly regularly without any outlier in sight.

Now that we had already corroborated the relatively better efficacy of the ensemble approach as compared to the LSTM only approach, we further carried out probabilistic forecasting for this particular strategy to check its probabilistic nature. The probabilistic curve was determined within the 95% uncertainty in the forecast. Bootstrapping method was used for determining the probabilistic forecast curve (shown in Fig. 7).

The probabilistic curve showed a higher degree of confidence as can be visualized from the stretch of the shade. It reassures the forecast's reliability for containing a little degree of uncertainty throughout the forecasting procedure.

B. Day ahead forecast

For the day-ahead forecast, similar procedures were followed as was done with the hour ahead forecast. The model was tweaked in places to produce day ahead forecasts instead of an hour ahead ones. The model was observed to have trained faster compared to hour ahead forecasts due to its reduced

sample feeds. Total time spent to train the model for the LSTM only method had been 8.37 minutes which is around 50 minutes less than for the hour ahead forecast training period. The inclusion of a higher number of features that had been crafted for this model caused the loss curve to converge at a higher epoch than the hour ahead scenario.

The graph shows that convergence was found to be at 1200 epochs and no further improvement was observed beyond this level. The loss curves had shown similar characteristics just as they did for the hour ahead scenario. The model generated a Test MAPE of 3.0818 for a day ahead forecast after epoch completion.

For LSTM Ensemble the test MAPE was found to be 2.9440 after epoch completion. The ensemble model took longer to be trained as expected due to its multiple networks working in tandem but compared to the hour ahead forecast model, it performed a lot faster. It took in a total of 15.26. minutes to train the LSTM ensemble model for day ahead forecast which is around 72 minutes less than the hour ahead forecasting needed. Fig. 8 and Fig. 9 highlight the contrast between the LSTM and LSTM ensemble approach with respect to training and validation losses for a day ahead forecast. The correspondence between the loss curves with respect to epochs were fairly satisfactory to be reassured about its efficacy.

Similarly, to the hour ahead forecast, LSTM ensemble network was found to be more efficient compared to the day-

Fig. 9: Comparison of validation loss between LSTM and LSTM ensemble approach for day ahead forecast

Fig. 12: 95% Prediction interval for a day in January 2014 for a day ahead forecast

Fig. 10: Actual load and Predicted load for a day in July 2, 2014 with LSTM approach

ahead forecast, with a 0.1378% error reduction. Figure 10 and Figure 11 show the condition of the correspondence of the predicted load and the actual load for a day ahead forecast.

The efficiency of both of the models seemed to be relatively poorer to day-ahead forecast as compared to an hour-ahead forecast. The aberration of forecasted points was larger as compared to the aberration nature in the hour ahead forecasts. This can be attributed to the higher MAPE produced by the day-ahead forecast. But the value was not alarmingly high as to

Fig. 11: Actual load and Predicted load for a day in July 2, 2014 with LSTM Ensembled approach

disregard its utility. A probabilistic curve had been determined to visualize the uncertainty of this model. Figure 12 shows the forecasted curve with a 95% confidence interval for the LSTM Ensemble approach. Like in hour ahead forecast, lstm only method was skipped to avoid redundancy.

From the observations of the probabilistic curve, the confidence of the curve can be inferred to be pretty high, judging from the range of uncertainty. The probabilistic curve for both of the model prove higher degree of reliability of such models.

Furthermore, we corroborated the fact that lstm ensembled model performs relatively better than other conventional models for example, feedforward neural network model, convolutional neural network model and recurrent neural model, and even lstm only model by comparing their relative performance with respect to MAPE for both day ahead and hour ahead forecast. The bar for the lstm ensembled has been found to be lowest for both set of forecasts while the bar for the recurrent network were found to be comparatively higher. This finding comes into accord with other studies which too have found higher performing ability for lstm model over other neural network models, but the significance of this study is that it establishes the fact that lstm ensembled models perform even better than regular lstm standalone models, which proves that it performs subsequently better than other conventional neural network models as well. Although the improvement seems minute in terms of numbers but considering the economic ramifications it seems quite noteworthy enough. The improvement that had been found out through this study has the potential to save thousands of dollars for utility companies. As per hobbs, a 1% reduction in error can save utilities 1.6 million dollars, following which we can conclude that lstm ensemble method, which has a 0.04% error reduction than lstm only method, can save utilities 64000 dollars for hour ahead forecasts and 224000 dollars for day ahead forecasts due to a 0.14% error reduction. Figure 12 highlights a chart of the comparative picture of the performance of conventional neural network models with respect to their generated MAPE value. From the chart in figure 13, it is even more evident that instead of lstm network if we compare the lstm ensembled method with the other conventional neural network models, it seems to carry the potential to save even more for large scale utilities, since

Fig. 13: MAPE comparison chart for LSTM Ensembled, LSTM, FFN, CNN-1D and RNN network

the error differences are even much higher comparatively. For instance, compared to the RNN model, the LSTM ensembled model can save 246000 dollars for hour ahead and 2.424 million dollars in case of day ahead forecasts.

V. CONCLUSION

From our findings, we can safely conclude that LSTM ensembled strategies perform relatively better compared to LSTM only models for both hour ahead and day ahead forecasts. Although the day ahead forecast operated with a higher error range compared to the hour ahead forecasts, nonetheless the range of error did not exceed beyond a concerning threshold which may appear to nullify its utility. Such a finding has the potential to assist utilities and researchers in making more appropriate decision regarding model selection, model tweaking at the expense of relatively low resources and with a relatively higher degree of reliability for short term load forecasting. It also sheds some valuable insight into how ensemble learning fares better than single network learning for time series data. The output of this particular study reaffirms and quantifies that position.

The model might have performed a bit better if more meteorological parameters could have been funneled into it alongside the temperature data. Humidity, for example, plays a very significant role in terms of load usage. In warm climates, Higher humidity drives up load demands and low humidity drives it down. Such parameters could not be utilized in this study due to it being unavailable in the availed dataset. Same with wind speed, precipitation and other similar meteorological data, the unavailability of which handicapped us from doing any further experimentations with additional parameters and also hobbled us from modifying our models any further. Such modifications might have had the potential to increase the precision of the networks even further, presenting us with an even more apt scenario.

REFERENCES

- [1] G. Gross and F. Galiana, "Short-term load forecasting," *Proceedings of the IEEE*, vol. 75, no. 12, pp. 1558–1573, 1987.
- [2] I. Moghram and S. Rahman, "Analysis and evaluation of five short-term load forecasting techniques," *IEEE Transactions on Power Systems*, vol. 4, no. 4, pp. 1484–1491, 1989.
- [3] S. Rahman and R. Bhatnagar, "An expert system based algorithm for short term load forecast," *IEEE Transactions on Power Systems*, vol. 3, no. 2, pp. 392–399, 1988.
- [4] K. Jabbour, J. Riveros, D. Landsbergen, and W. Meyer, "Alfa: automated load forecasting assistant," *IEEE Transactions on Power Systems*, vol. 3, no. 3, pp. 908–914, 1988.
- [5] S.-J. Huang and K.-R. Shih, "Short-term load forecasting via arma model identification including non-gaussian process considerations," *IEEE Transactions on Power Systems*, vol. 18, no. 2, pp. 673–679, 2003.
- [6] C.-M. Lee and C.-N. Ko, "Short-term load forecasting using lifting scheme and arima models," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5902–5911, 2011.
- [7] G. Juberias, R. Yunta, J. G. Moreno, and C. Mendiivil, "A new arima model for hourly load forecasting," in *1999 IEEE Transmission and Distribution Conference (Cat. No. 99CH36333)*, vol. 1. IEEE, 1999, pp. 314–319.
- [8] W. Christiaanse, "Short-term load forecasting using general exponential smoothing," *IEEE Transactions on Power Apparatus and Systems*, no. 2, pp. 900–911, 1971.
- [9] M. S. Al-Hafid and G. Hussein Al-maamary, "Short term electrical load forecasting using holt-winters method," *Al-Rafidain Engineering Journal (AREJ)*, vol. 20, no. 6, pp. 15–22, 2012.
- [10] J. W. Taylor and P. E. McSharry, "Short-term load forecasting methods: An evaluation based on european data," *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 2213–2219, 2007.
- [11] J. Park, Y. Park, and K. Lee, "Composite modeling for adaptive short-term load forecasting," *IEEE Transactions on Power Systems*, vol. 6, no. 2, pp. 450–457, 1991.
- [12] H. Al-Hamadi and S. Soliman, "Short-term electric load forecasting based on kalman filtering algorithm with moving window weather and load model," *Electric power systems research*, vol. 68, no. 1, pp. 47–59, 2004.
- [13] M. T. Hagan and S. M. Behr, "The time series approach to short term load forecasting," *IEEE transactions on power systems*, vol. 2, no. 3, pp. 785–791, 1987.
- [14] N. Amjadi, "Short-term hourly load forecasting using time-series modeling with peak load estimation capability," *IEEE Transactions on power systems*, vol. 16, no. 3, pp. 498–505, 2001.
- [15] H.-T. Yang and C.-M. Huang, "A new short-term load forecasting approach using self-organizing fuzzy armax models," *IEEE Transactions on Power Systems*, vol. 13, no. 1, pp. 217–225, 1998.
- [16] H.-T. Yang, C.-M. Huang, and C.-L. Huang, "Identification of armax model for short term load forecasting: An evolutionary programming approach," in *Proceedings of Power Industry Computer Applications Conference*. IEEE, 1995, pp. 325–330.
- [17] A. D. Papalexopoulos and T. C. Hesterberg, "A regression-based approach to short-term system load forecasting," *IEEE Transactions on power systems*, vol. 5, no. 4, pp. 1535–1547, 1990.
- [18] T. Haida and S. Muto, "Regression based peak load forecasting using a transformation technique," *IEEE Transactions on Power Systems*, vol. 9, no. 4, pp. 1788–1794, 1994.
- [19] R. Ramanathan, R. Engle, C. W. Granger, F. Vahid-Araghi, and C. Brace, "Short-run forecasts of electricity loads and peaks," *International journal of forecasting*, vol. 13, no. 2, pp. 161–174, 1997.
- [20] S. C. Pandian, K. Duraiswamy, C. C. A. Rajan, and N. Kanagaraj, "Fuzzy approach for short term load forecasting," *Electric power systems research*, vol. 76, no. 6-7, pp. 541–548, 2006.
- [21] G. Černe, D. Dovžan, and I. Škrjanc, "Short-term load forecasting by separating daily profiles and using a single fuzzy model across the entire domain," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 9, pp. 7406–7415, 2018.
- [22] R. Mamlook, O. Badran, and E. Abdulhadi, "A fuzzy inference model for short-term load forecasting," *Energy Policy*, vol. 37, no. 4, pp. 1239–1248, 2009.
- [23] W. Ahmad, N. Ayub, T. Ali, M. Irfan, M. Awais, M. Shiraz, and A. Glowacz, "Towards short term electricity load forecasting using improved support vector machine and extreme learning machine," *Energies*, vol. 13, no. 11, p. 2907, 2020.
- [24] X. Zhang, J. Wang, and K. Zhang, "Short-term electric load forecasting based on singular spectrum analysis and support vector machine optimized by cuckoo search algorithm," *Electric Power Systems Research*, vol. 146, pp. 270–285, 2017.
- [25] W.-C. Hong, "Electric load forecasting by support vector model," *Applied Mathematical Modelling*, vol. 33, no. 5, pp. 2444–2454, 2009.
- [26] D. Niu, Y. Wang, and D. D. Wu, "Power load forecasting using support vector machine and ant colony optimization," *Expert Systems with Applications*, vol. 37, no. 3, pp. 2531–2539, 2010.

- [27] M. Rejc and M. Pantos, "Short-term transmission-loss forecast for the slovenian transmission power system based on a fuzzy-logic decision approach," *IEEE Transactions on Power Systems*, vol. 26, no. 3, pp. 1511–1521, 2011.
- [28] S. Li, L. Goel, and P. Wang, "An ensemble approach for short-term load forecasting by extreme learning machine," *Applied Energy*, vol. 170, pp. 22–29, 2016.
- [29] E. Ceperic, V. Ceperic, and A. Baric, "A strategy for short-term load forecasting by support vector regression machines," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4356–4364, 2013.
- [30] G. Dudek, "Short-term load forecasting using random forests," in *Intelligent Systems' 2014*. Springer, 2015, pp. 821–828.
- [31] E. Aguilar Madrid and N. Antonio, "Short-term electricity load forecasting with machine learning," *Information*, vol. 12, no. 2, p. 50, 2021.
- [32] J. Luo, T. Hong, and S.-C. Fang, "Benchmarking robustness of load forecasting models under data integrity attacks," *International Journal of Forecasting*, vol. 34, no. 1, pp. 89–104, 2018.
- [33] H. S. Hippert, C. E. Pedreira, and R. C. Souza, "Neural networks for short-term load forecasting: A review and evaluation," *IEEE Transactions on power systems*, vol. 16, no. 1, pp. 44–55, 2001.
- [34] K. Lee, Y. Cha, and J. Park, "Short-term load forecasting using an artificial neural network," *IEEE transactions on power systems*, vol. 7, no. 1, pp. 124–132, 1992.
- [35] S. R. Salkuti, "Short-term electrical load forecasting using radial basis function neural networks considering weather factors," *Electrical Engineering*, vol. 100, no. 3, pp. 1985–1995, 2018.
- [36] C. Guan, P. B. Luh, L. D. Michel, Y. Wang, and P. B. Friedland, "Very short-term load forecasting: wavelet neural networks with data pre-filtering," *IEEE Transactions on Power Systems*, vol. 28, no. 1, pp. 30–41, 2012.
- [37] P. Ray, S. K. Panda, and D. P. Mishra, "Short-term load forecasting using genetic algorithm," in *Computational Intelligence in Data Mining*. Springer, 2019, pp. 863–872.
- [38] Z. Bashir and M. El-Hawary, "Applying wavelets to short-term load forecasting using pso-based neural networks," *IEEE transactions on power systems*, vol. 24, no. 1, pp. 20–27, 2009.
- [39] S. Li, P. Wang, and L. Goel, "Short-term load forecasting by wavelet transform and evolutionary extreme learning machine," *Electric Power Systems Research*, vol. 122, pp. 96–103, 2015.
- [40] A. Al Mamun, M. Sohel, N. Mohammad, M. S. H. Sunny, D. R. Dipta, and E. Hossain, "A comprehensive review of the load forecasting techniques using single and hybrid predictive models," *IEEE Access*, vol. 8, pp. 134 911–134 939, 2020.
- [41] M. Barman, N. D. Choudhury, and S. Sutradhar, "A regional hybrid goa-svm model based on similar day approach for short-term load forecasting in assam, india," *Energy*, vol. 145, pp. 710–720, 2018.
- [42] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [43] Z. Deng, B. Wang, Y. Xu, T. Xu, C. Liu, and Z. Zhu, "Multi-scale convolutional neural network with time-cognition for multi-step short-term load forecasting," *IEEE Access*, vol. 7, pp. 88 058–88 071, 2019.
- [44] N. M. M. Bendaoud and N. Farah, "Using deep learning for short-term load forecasting," *Neural computing and applications*, vol. 32, no. 18, pp. 15 029–15 041, 2020.
- [45] S. M. J. Jalali, S. Ahmadian, A. Khosravi, M. Shafie-khah, S. Nahavandi, and J. P. Catalao, "A novel evolutionary-based deep convolutional neural network model for intelligent load forecasting," *IEEE Transactions on Industrial Informatics*, 2021.
- [46] M. Alhussein, K. Aurangzeb, and S. I. Haider, "Hybrid cnn-lstm model for short-term individual household load forecasting," *IEEE Access*, vol. 8, pp. 180 544–180 557, 2020.
- [47] X. Guo, Q. Zhao, D. Zheng, Y. Ning, and Y. Gao, "A short-term load forecasting model of multi-scale cnn-lstm hybrid neural network considering the real-time electricity price," *Energy Reports*, vol. 6, pp. 1046–1053, 2020.
- [48] B. Farsi, M. Amayri, N. Bouguila, and U. Eicker, "On short-term load forecasting using machine learning techniques and a novel parallel deep lstm-cnn approach," *IEEE Access*, vol. 9, pp. 31 191–31 212, 2021.
- [49] C. Tian, J. Ma, C. Zhang, and P. Zhan, "A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network," *Energies*, vol. 11, no. 12, p. 3493, 2018.
- [50] G. M. U. Din and A. K. Marnerides, "Short term power load forecasting using deep neural networks," in *2017 International conference on computing, networking and communications (ICNC)*. IEEE, 2017, pp. 594–598.
- [51] S. Ryu, J. Noh, and H. Kim, "Deep neural network based demand side short term load forecasting," *Energies*, vol. 10, no. 1, p. 3, 2017.
- [52] X. Kong, C. Li, F. Zheng, and C. Wang, "Improved deep belief network for short-term load forecasting considering demand-side management," *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1531–1538, 2019.
- [53] X. Zhang, R. Wang, T. Zhang, Y. Liu, and Y. Zha, "Short-term load forecasting using a novel deep learning framework," *Energies*, vol. 11, no. 6, p. 1554, 2018.
- [54] B. Zhang, J.-L. Wu, and P.-C. Chang, "A multiple time series-based recurrent neural network for short-term load forecasting," *Soft Computing*, vol. 22, no. 12, pp. 4099–4112, 2018.
- [55] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting—a novel pooling deep rnn," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 5271–5280, 2017.
- [56] M. Zhang, Z. Yu, and Z. Xu, "Short-term load forecasting using recurrent neural networks with input attention mechanism and hidden connection mechanism," *IEEE Access*, vol. 8, pp. 186 514–186 529, 2020.
- [57] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on lstm recurrent neural network," *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 841–851, 2017.
- [58] B.-S. Kwon, R.-J. Park, and K.-B. Song, "Short-term load forecasting based on deep neural networks using lstm layer," *Journal of Electrical Engineering & Technology*, vol. 15, no. 4, pp. 1501–1509, 2020.
- [59] W. Kong, Z. Y. Dong, D. J. Hill, F. Luo, and Y. Xu, "Short-term residential load forecasting based on resident behaviour learning," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 1087–1088, 2017.
- [60] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on lstm recurrent neural network," *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 841–851, 2017.
- [61] J. Li, D. Deng, J. Zhao, D. Cai, W. Hu, M. Zhang, and Q. Huang, "A novel hybrid short-term load forecasting method of smart grid using mlr and lstm neural network," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2443–2452, 2021.
- [62] H. Zheng, J. Yuan, and L. Chen, "Short-term load forecasting using emd-lstm neural networks with a xgboost algorithm for feature importance evaluation," *Energies*, vol. 10, no. 8, p. 1168, 2017.
- [63] H. Choi, S. Ryu, and H. Kim, "Short-term load forecasting based on resnet and lstm," in *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2018, pp. 1–6.
- [64] B. F. Hobbs, S. Jitprapaikulsaarn, S. Konda, V. Chankong, K. A. Loparo, and D. J. Maratukulam, "Analysis of the value for unit commitment of improved load forecasts," *IEEE Transactions on Power Systems*, vol. 14, no. 4, pp. 1342–1348, 1999.
- [65] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [66] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.
- [67] T. G. Dietterich, "Ensemble methods in machine learning," in *International workshop on multiple classifier systems*. Springer, 2000, pp. 1–15.
- [68] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.