Short Term Load Forecasting Using Deep Neural Networks

AMS559: Smart Energy in the Information Age, Spring 2020

Stony Brook University

Gaganraj Maheshwari
112674425
gmaheshwari@cs.stonybrook.edu

Vibhor Shukla
112671826
vshukla@cs.stonybrook.edu

Uzair Bin Tariq
112872382
utariq@cs.stonybrook.edu

Key words: Load Forecasting, Deep Neural Networks, CNN, LSTM
Dated: May 13, 2020

Abstract — Owing to the variable nature of solar, wind, hydro, geothermal, the integration of renewable energy to the existing infrastructure of grid systems in a reliable fashion has always been a non-trivial task. The output of renewable energy sources, for instance, solar depends on clouds and external temperature, whereas, for wind, it depends on the temperature gradient along with planetary movements. A smart grid will model this variability of sources in an efficient manner. Therefore, accurate generation forecasting with a high level of confidence is an important piece to solve this integration puzzle. We will primarily be focusing on forecasting the generation via solar energy. Once we achieve concrete results, we can extend similar methodologies for other renewable resources such as wind, hydro etc.

I. INTRODUCTION

The contemporary usage of energy is based on momentary needs in which energy generation and management need far more attention than currently paid. This is because there is only a limited amount of energy available in the form of burning fossil fuels whilst the demand for energy has risen in an unparalleled fashion since the beginning of the human civilisation. A smart way of energy usage involves using it judiciously so as not to impose pressure on the entire electric grid. One can produce electricity using renewable energy sources like solar, wind, waves etc. which if tapped properly can diminish the scarcity of energy in a tremendously populated world. If each edge consumer installs renewable energy generation capability then one can smartly consume energy from the grid and his personal system and can even contribute the excess to the grid. The prediction of renewable energy sources like solar energy can play a paramount role to enable a smart usage of electricity.

Deep Neural Network architecture comes as an important model to perform load forecasting. Artificial Neural Networks or ANNs are motivated by the human brain system with its information processing and distributed communication nodes. DNNs are a form of ANNs which involve a deep layer structure. Multiple layers are vital to carry out the learning process which is well abstracted by the layers. The Convolutional Neural Networks and the Long Short Term Networks are essential Deep Neural Network models specialising in the sequence prediction.
Some of the terminologies used in this study are defined as follows.

A. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks or CNN is a neural network class for deep neural networks. It employs convolution which is a kind of linear equation. CNN uses convolution in place of matrix multiplication in one of the layers. Convolution is a type of cross-correlation or sliding dot product.

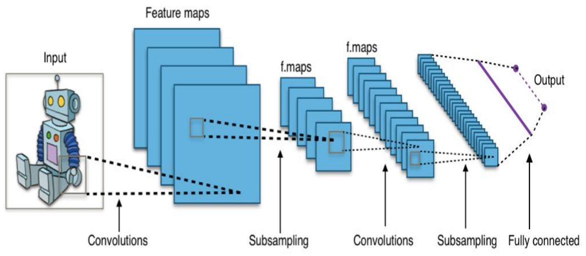


Fig.1 A CNN architecture

As in Fig. 1, A CNN consists of a convolution, subsampling, more convolutions, again subsampling and then the fully connected layers. A feature map is produced by sliding the kernel across the input which is the actual convolution operation. After the convolution layer comes the pooling layer, it samples down the A CNN is based on the working principle of using the human visual cortex for object recognition. A
a series of different filter layers are used to produce unique filter maps.

B. Long Short Term Memory (LSTM)

Long Short Term Memory is an algorithm that is actively used for the sequence prediction problems which are one of the hardest problems to solve in the current industry. The problems are some of the most complex problems such as the voice recognition, translating the languages and in our case, smart energy forecasting

Recurrent Neural Networks have been employed to solve some of these problems and to a greater efficacy. RNNs are good in short context scenarios. But, when it comes to remembering longer contexts and understanding them like a human brain, they are not that efficient. Anything called long back is not remembered by RNNs. This issue can be well solved by the Long Short Term Memory Networks.

LSTMs have the ability to prioritise information. They are able to distinguish which information is *important* and which are *not important.* LSTMs use cell states to categorise information. They carefully make use of what was trending before, the information on the previous day and the factors which can affect information today. The figure 2 portrays the LSTM architecture.

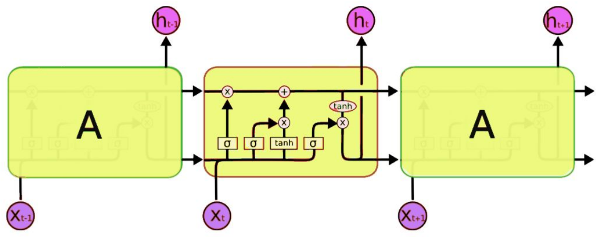


Fig.2 An LSTM architecture

The gates are responsible to do the manipulations to the memory blocks. The memory blocks are called cells which are the yellow shapes in Figure 2. The states are transferred to the next cell states which are cell state and hidden state. We have used the keras library in python to use the LSTM model.

C. CNN LSTM Networks

A combination of the CNN LSTM architecture is fundamentally a Convolutional Neural Network used for feature extraction and the Long Short Term Memory Model used to predict the sequences. CNN LSTMs are used for the problems involving image description, video description,activity recognition and an accurate future energy prediction. The CNN LSTMs are implemented by the Keras library in Python. It involves adding CNN layers in the front, then the LSTM layers having a dense layer in the output.

The individual capabilities of the CNN model and the LSTM model are combined to produce the energy consumption prediction that has the minimum root mean square error in our study.

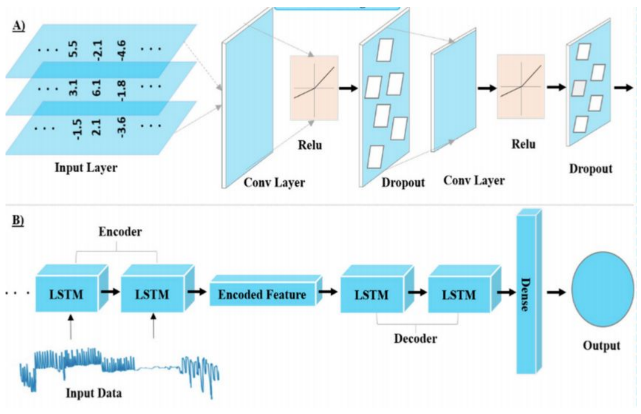


Fig.3 A CNN-LSTM Architecture

As in Figure 3, part A consists of CNN modelling and part B shows the LSTM modelling. The input layer consisting of the time series data is injected in the CNN layer. The resulting output is pushed into the encoder in the LSTM layer which is later densely decoded in the form of the output.

D. Univariate Time Series Forecasting

Univariate time series forecasting is predicting the value of a single variable over time. For instance, if we record the temperature of our body every second, and then predict the temperature in the next second based on the previously recorded values. In this study, the energy load units are taken with time.

E. Multivariate Time Series Forecasting

Multivariate time series means that multiple variables are varying over time. For instance, an accelerometer which is configured on three axes. With each passing time unit, there will be changes in the x, y and z axis altogether. In this study, we are using solar energy, fossil oil shale, geothermal energy load generation, etc. with the time axis.

II. MOTIVATING EXAMPLES

The following reasons are the key motivators to shift towards renewables.

1. *The environmental benefit of integrating renewables*: Due to rapid industrialization and unsustainable ways of generating energy, we are now witnessing the most unprecedented time in human history. This rapid generation and consumption have led to the destruction of ecosystems, change in weather patterns are a few problems to name amongst millions of problems. Shifting to renewables for our electricity generation will help us cut down our carbon footprint by 27 percentage.

2. *Improved public health:* Less carbon footprint will lead to a cleaner environment and hence reduced the risk of airborne illnesses. As per the current statistics: Air Pollution Causes 8.8 Million extra deaths a year.

3. *Jobs and Economic Benefits of integrating renewables:* Projections show that doubling the share of renewables by 2030 would bring a range of positive impacts including an increase in the global gross domestic product (GDP) up to 1.1 percent, improvement of global welfare by 3.7 percent and over 24 million people working in the renewable energy sector, this clearly shows the great economic benefit of shifting to renewables.

Why should solar be one of the primary focus amongst all renewables?
Due to the abundance of sunlight available to us throughout the major part of the year around the globe, solar energy represents the most promising renewable alternative to fossil fuels based energy. Also, we can see from the figure that solar is having the highest penetration.

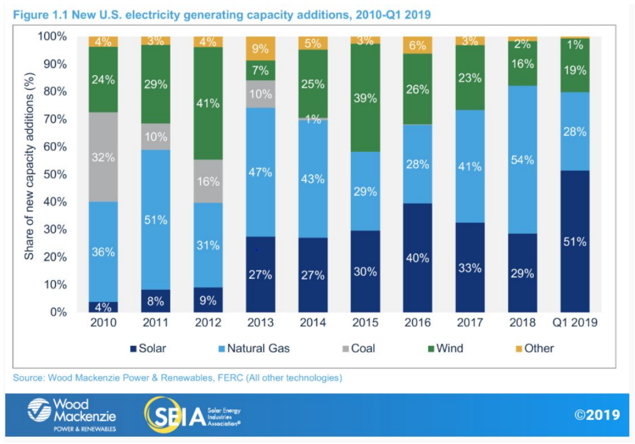


Fig.4 Share of source of energy

Why forecasting solar energy generated a challenging problem?
Though we can model the sun path with physical laws, prediction of the amount of electricity generated is a challenging problem since it depends on a lot of factors like sun position, weather condition, and the characteristics of a photovoltaic panel, curtailment, etc.

III. LITERATURE REVIEW

As part of our quest to understand the topic in detail and to devise our own novel method, we went through a lot of relevant material which had Deep Neural Network techniques like CNN, LSTM, ANN, FCRBM etc. to predict smart energy forecast as their main essence. Key points which we felt core to our research pertaining to each paper are discussed as follows:

1) **Paper:** Deep neural networks for energy load forecasting

This paper explains load forecasting using DNNs, Conditional Restricted Boltzmann Machines (CRBM) and Factored Conditional Restricted Boltzmann Machines (FCRBM). The study has used totally the load data and not the weather data in order to separately evaluate the weightage of the grid data alone in the load forecast measurements. The performance of the CNNs was compared against SVM, ANN, LSTM and FCRBM.

The kernel is a multidimensional array of weights which changes as the algorithm learns through the iterations.
The inputs represented in multidimensional arrays get the convolution. Due to the inputs and kernels being multidimensional, the convolution operation is applied to more than one dimension. Thus, the convolution operation for a two dimensional input can be :

$$s(i, j)=(I*K)(i, j)=\sum\sum I(l, m)K(i+l, j+m)$$

Where K represents a two dimensional kernel, S is the feature map and I is a 2D input.

Each convolutional layer consists of three phases. The first phase produces a feature map. Then, the elements of the feature map are run through a nonlinear activation function. Lastly, a pooling function is used in the 3rd stage to further smoothen the feature map. The pooling operation makes the representation less prone to little variations in the input. Max pooling method is used. In max pooling, the operation returns the maximum value of a predefined rectangular neighborhood. Other pooling techniques such as average pooling, min pooling and weighted average pooling have been used in theory.

The CNN based load forecasting algorithm is based on one-hour data. It was seen that when the specific training testing split was used, the testing error was lower than the training error. The results from CNN were compared to ANN, SVM,LSTM and FCRBM. It was seen that results did not differ much across the many architectures. Hence, it can be deduced that CNN remains to be an important candidate for producing correct load forecasts.

2) **Paper:** Deep Learning based Ensemble Method for Household Energy Demand Forecasting of Smart Home

The main contribution in this paper is that a novel approach has been taken here in which time series analysis is done, neural network models are used on the household dataset and finally the Mahalanobis distance is then used to build models to make future household electricity consumption predictions. This approach gives better results compared to prediction with individual models.

The dataset contains electricity data of one household at a one-minute sampling rate for 4 years between 2006 to 2010. Performing a number of data pre-processing ways on the dataset, in order to convert some aspects of the data like creating a new column sub metering 4, replace missing values. An LSTM block contains: forget gate, input gate, a tanh layer and an output gate. For univariate linear regression, a single variable is used as input feature and it is used to make a prediction of the dependent value using,

$$y = b_0 + b_1 * x$$

In multivariate linear regression multiple feature values are used to train a regression model and get a prediction of the dependent value using,

$$y = b_0 + b_1 * x_1 + \ldots + b_n * x_n$$

For the ARIMA model, it is visible that the model outputs prediction values of energy demand with a high degree of accuracy. The RNN LSTM model is evaluated. The error difference between actual and predicted value made the model updated itself. The LSTM model is great in capturing the overall pattern of the data. Post univariate regression, we get a best-fit line that has been able to fit the data according to covering most data points in the distribution. For multivariate linear regression, a prediction model which predicts the global active power but seeing multiple input features is also considered. Ultimately, to reduce the forecast errors, they obtained prediction of household energy demand using the Mahalanobis distance. It is seen that the values have been predicted very accurately.

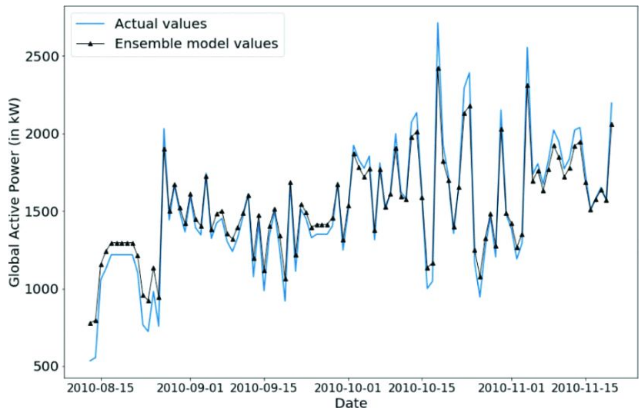


Fig.5 The ensemble model

The superior performance can be understood by the Mahalanobis distance based ensemble method which punishes the poor performance models and takes into account the better performing and more accurate models for the prediction.

3) **Paper:** Load forecasting using deep neural networks

There has been tremendous research done in the area of short-term load forecasting (STLF) . In this paper we compare deep architectures and traditional methods when applied to our STLF problem and we also provide a comprehensive analysis of numerous deep learning models. We then show how these methods can be used to assist in the pricing of electrical rates.
Stacked Autoencoders (SA) utilize a stacked architecture, where there is an autoencoder in each layer. An autoencoder is an unsupervised learning algorithm that is trained to reconstruct the input in the output layer

Recurrent Neural Networks (RNN) are another architecture. RNNs are neural networks containing feedback loops thus allowing the persistence of information.

Our dataset consists of hourly samples over the period of a year and consists of 18 features. The dataset was broken into 3 parts for training, validation and testing of size 65%, 15%, 20% respectively. The readings were recorded at hourly intervals throughout the year. The rest of the features (which do not contain electrical load readings) are the day of the week, hour of the day, if it is a weekend, if it is a holiday, temperature and humidity. Several baseline algorithms were chosen. They include the Weighted Moving Average (WMA)

For our Deep Neural Networks we used1: Deep Neural Network without pretraining (DNN-W), DNN with pretraining using Stacked Autoencoders (DNN-SA), Recurrent Neural

Networks (RNN), RNNs and Long Short Term Memory (RNN-LSTM).

From the table we see that MLR performs the worst, with a MAPE of 24.25%, which would indicate that the problem is not linear.. However, the RT algorithm outperforms the rest of the methods by a noticeable margin. This shows that the problem can be split into some discrete segments which would accurately forecast the load. This can be confirmed by the load. It is clear that depending on the time of day, there is significant overlap of the value of the load between days.

It is seen that most of the DNN algorithms have their lowest MAPE during the week.

4) **Paper:** Review on the Research and Practice of Deep Learning and Reinforcement Learning in Smart Grids

Artificial intelligence will be one of the driving factors in the integration of renewables and the widespread adoption of the smart grid. Deep learning and reinforcement learning are some of the most promising results in many domains. Many of these algorithms are used in multiple domains of smart grid with some of them being:

1. load/demand forecasting
2. In microgrid architecture to help with decision-making capabilities for renewable energy generation and future demand.
3. Demand response i.e. how flexible is the demand side with respect to its energy consumption.
4. Defect/Fault Detection of Electrical Equipment
5. Cybersecurity
6. Renewable energy generation prediction

With our focus on part f, there has been recent research on it. Deep learning(DL) models like LSTM have been applied to predict the generation of wind power. DL has the qualities of being flexible, self-adaptive learning abilities, and the relaxation of the need for physical and phenomenological assumptions, the expectation for the prediction accuracy to be enhanced by combining DL methods with more data sources.

5) **Paper:** Forecast of Solar Energy Production – A DeepLearning Approach

One of the major factors impacting the generation of solar energy is the movement of clouds. This paper tries to propose a novel algorithm to accurately predict the movement of the cloud and its impact on solar energy generation. Paper explores both the state-of-the-art VCNN (Volumetric Convolution Neural Network) and RNN(Recurrent Neural Network). The paper first draws the attention to drawbacks of point-wise solar forecast when it is partially cloudy sky. Firstly, the spatial inaccuracy of the weather forecast, i.e., the exact location and size of the cloud;Secondly, the temporal inaccuracy of the weather forecast, i.e.,the moving pattern of clouds. To overcome this a system named Deep Thunder is used to generate high resolution,high fidelity customized weather forecast engines. The task of modelling weather has been analogized against video classification and action recognition where the learning task is applied on sequences of images constituting as a 4D tensor input data. For the paper 4 dimensions are: weather features (17 features), time, height and width. Using the following methodologies the inventors were able to achieve MAPE of the following extent:

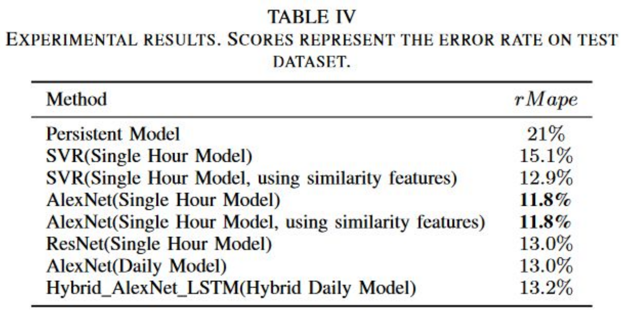


TABLE IV

EXPERIMENTAL RESULTS. SCORES REPRESENT THE ERROR RATE ON TEST DATASET.

Method	$rMape$
Persistent Model	21%
SVR(Single Hour Model)	15.1%
SVR(Single Hour Model, using similarity features)	12.9%
AlexNet(Single Hour Model)	**11.8%**
AlexNet(Single Hour Model, using similarity features)	**11.8%**
ResNet(Single Hour Model)	13.0%
AlexNet(Daily Model)	13.0%
Hybrid_AlexNet_LSTM(Hybrid Daily Model)	13.2%

Fig.6 Architecture and their Mape

The paper successfully shows that the model of AlexNet usually employed in the image processing domain can be employed to model solar energy generation with significant improvement of rMape statistic.

6) **Paper:** Building Energy Load Forecasting using Deep Neural Networks

This paper presents a novel energy load forecasting methodology based on Deep Neural Networks, specifically Long Short Term Memory (LSTM) algorithms. Keypoint of this paper was to mitigate the effect of learning the wrong patterns. Pattern being: consequent measurements are very similar(when using one-minute resolution data). Therefore, if the network predicts the load for the next time step is the same as the load on the current time. Thus, the neural network is learning a naïve mapping, where it generates an output equal to the input

Mitigation strategy:

First was to introduce measurements from further in the past as inputs, for example, 5 steps back, as opposed to inputting the load from the previous time step. This was done so that the input and output would be different enough for the network to be able to learn a useful representation of the data.

7) **Paper:** [Solar Power Generation Forecast Based on LSTM]

This paper uses PCA to reduce dimensionality of data hence reducing the training time and then feeding that data to LSTM to forecast the solar electricity generated. Forecast are of different categories:

A. Long term forecast

B. Medium term forecast

C. Short term forecast

The long and the short term forecast can range anywhere between weeks/months and years. Short term forecast focuses on hourly output.

Why use PCA? PCA helps reduce noise and remove redundancy and hence only feeding data to model which helps it generalize things. Noise reduction helps to minimize correlation between remaining dimensions as small as possible. Purpose of removing redundancy is to have maximum variance in remaining data.

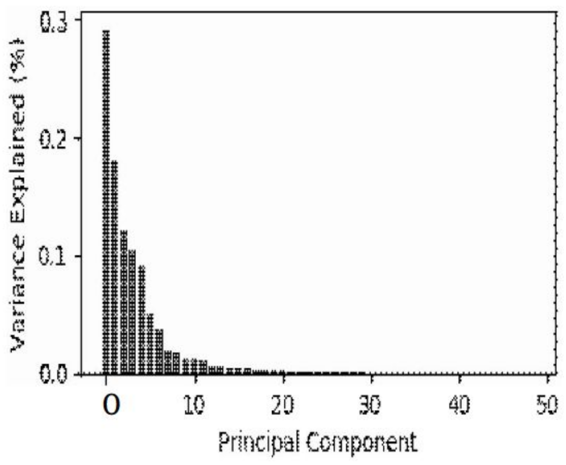


Figure1. The contributi on rate of principal comp onent

Fig.7 contribution rate of principal component

Based on the ordering of features on the basis of variance we can see from all the 49 inputs only 23 inputs will be consumed for training our model. For short term forecasting we need the prediction as fast as possible in order to get time to make arrangements for excess supply from other means if our renewable source can't satisfy. Due to the reduced dimension done by PCA we will be touching on crucial stuff impacting our result and hence crunching less data and ultimately getting results faster. LSTM models are best suited for time series problems and solar energy generated is one of them. Since the severity of sunshine depends on the time of the year and day. While traditional RNN models are prone to gradient vanishing problems, LSTM is immune to it.

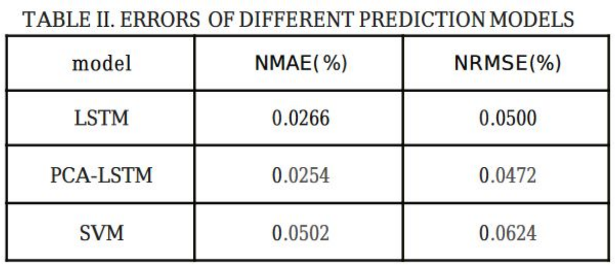


TABLE II. ERRORS OF DIFFERENT PREDICTION MODELS

model	NMAE(%)	NRMSE(%)
LSTM	0.0266	0.0500
PCA-LSTM	0.0254	0.0472
SVM	0.0502	0.0624

Fig.8 Model and there errors

The table above shows the result of running different models on European wind farm dataset. The metrics used are normalized mean absolute error and normalized root mean square error. As is evident both LSTM are doing very well as compared to SVM.

Comparing PCA LSTM with LSTM we can see that though PCA reduced dimension from 49 to 23 the NMAE only got changed by |0.0254-0.0266| percentage. Hence for speed critical use cases, PCA seems like a good option to reduce processing time.

8) **Paper:** [Deep Learning for Solar Power Forecasting –An Approach Using Autoencoder and LSTM Neural Networks]

Physical models use physical metrics like wind turbine / solar power curves. Since they rely on physical attributes is easy comprehensibility. Machine learning models the relationship between input parameters and output value. One of the most promising architectures in machine learning is deep learning since it learns feature representation of data avoiding the hassle of generating the features manually.
A multilayer perceptron consists of fully connected neurons with each neuron's input and out modelled mathematically function expressed as, where b is bias, f is the activation function and w represent the weight of a connection link. Back propagation is used to train this network.

$$output = f(\sum_{i}^{Inputs} (x_i \cdot w_i + b_i))$$

MLP network topology:

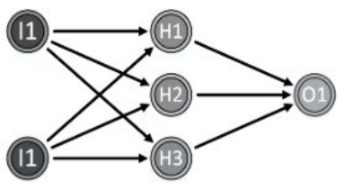


Fig. 1. An exemplary MLP network topology.

Fig.9 MLP topology

The paper then discussed architecture of a standard LSTM architecture. The experimental evaluation paper specifies, as it's reference model, a physical photovoltaic forecasting model. There's a Clear-sky filter used for estimating terrestrial solar radiation of the sun with a cloudless sky for a specific time and coordinates. Also helps to cut off the output during night.
The paper also used AUTO-LSTM. The Auto-LSTM algorithm combines the feature learning of an AutoEncoder with the temporal context usage of an LSTM in a two-step approach:
1) An Auto-Encoder (AE) will be used to realize the feature learning. AE is an MLP with the following architecture. The number of input and output remain the same. With the cells from input decreasing[Encoding part] first till the bottleneck neuron and then again expanding[Decoding part].

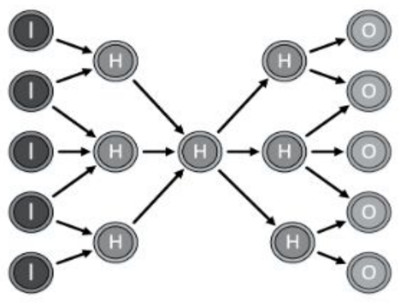


Fig. 2. An exemplary AutoEncoder topology.

Fig.10 A autoencoder topology

2) An LSTM network is attached to the encoding part of the AE. Hence, it uses temporal information in the form of sequences of the extracted features.

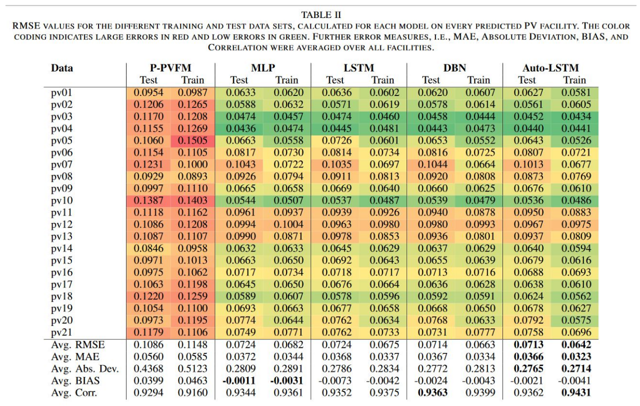


TABLE II
RMSE values for the different training and test data sets, calculated for each model on every predicted PV facility. The color coding indicates large errors in red and low errors in green. Further error measures, i.e., MAE, Absolute Deviation, BIAS, and Correlation were averaged over all facilities.

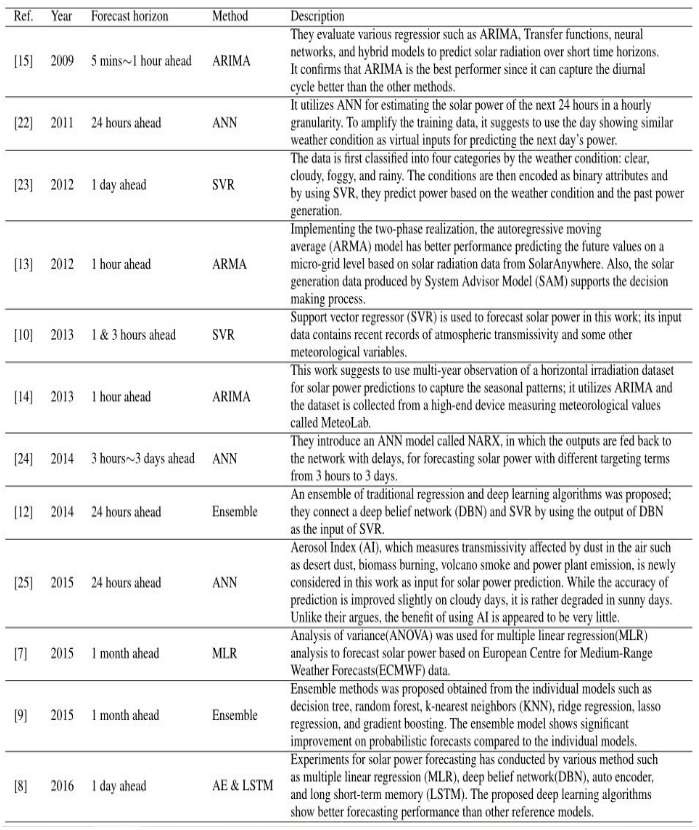
Data	P-PVFM		MLP		LSTM		DBN		Auto-LSTM	
	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train
pv01	0.0954	0.0987	0.0633	0.0620	0.0636	0.0602	0.0620	0.0607	0.0627	0.0581
pv02	0.1206	0.1265	0.0588	0.0632	0.0571	0.0619	0.0578	0.0614	0.0561	0.0605
pv03	0.1170	0.1208	0.0474	0.0457	0.0474	0.0460	0.0458	0.0444	0.0452	0.0434
pv04	0.1155	0.1269	0.0436	0.0474	0.0445	0.0481	0.0443	0.0473	0.0440	0.0441
pv05	0.1060	0.1505	0.0663	0.0558	0.0726	0.0601	0.0653	0.0552	0.0643	0.0526
pv06	0.1154	0.1105	0.0817	0.0730	0.0814	0.0734	0.0816	0.0725	0.0807	0.0721
pv07	0.1231	0.1000	0.1043	0.0722	0.1035	0.0697	0.1044	0.0664	0.1013	0.0677
pv08	0.0929	0.0893	0.0926	0.0794	0.0911	0.0813	0.0920	0.0808	0.0873	0.0769
pv09	0.0997	0.1110	0.0665	0.0658	0.0669	0.0640	0.0660	0.0625	0.0676	0.0610
pv10	0.1387	0.1403	0.0544	0.0507	0.0537	0.0487	0.0539	0.0479	0.0536	0.0486
pv11	0.1118	0.1162	0.0961	0.0937	0.0939	0.0926	0.0940	0.0878	0.0950	0.0883
pv12	0.1086	0.1208	0.0994	0.1004	0.0963	0.0980	0.0980	0.0993	0.0967	0.0975
pv13	0.1087	0.1107	0.0990	0.0871	0.0978	0.0853	0.0936	0.0801	0.0937	0.0809
pv14	0.0846	0.0958	0.0632	0.0633	0.0645	0.0629	0.0637	0.0629	0.0640	0.0594
pv15	0.0971	0.1013	0.0663	0.0650	0.0692	0.0643	0.0655	0.0639	0.0679	0.0616
pv16	0.0975	0.1062	0.0717	0.0734	0.0718	0.0717	0.0713	0.0716	0.0688	0.0693
pv17	0.1063	0.1198	0.0645	0.0650	0.0676	0.0664	0.0636	0.0628	0.0638	0.0610
pv18	0.1220	0.1259	0.0589	0.0607	0.0578	0.0596	0.0592	0.0591	0.0624	0.0562
pv19	0.1054	0.1100	0.0693	0.0663	0.0677	0.0658	0.0668	0.0650	0.0678	0.0627
pv20	0.0973	0.1195	0.0774	0.0644	0.0762	0.0634	0.0768	0.0633	0.0792	0.0575
pv21	0.1179	0.1106	0.0749	0.0771	0.0762	0.0733	0.0731	0.0777	0.0758	0.0696
Avg. RMSE	0.1086	0.1148	0.0724	0.0682	0.0724	0.0675	0.0714	0.0663	**0.0713**	**0.0642**
Avg. MAE	0.0560	0.0585	0.0372	0.0344	0.0368	0.0337	0.0367	0.0334	**0.0366**	**0.0323**
Avg. Abs. Dev.	0.4368	0.5123	0.2809	0.2891	0.2786	0.2834	0.2772	0.2813	**0.2765**	**0.2714**
Avg. BIAS	0.0399	0.0463	**-0.0011**	**-0.0031**	-0.0073	-0.0042	-0.0024	-0.0043	-0.0021	-0.0041
Avg. Corr.	0.9294	0.9160	0.9344	0.9361	0.9352	0.9375	**0.9363**	0.9399	0.9362	**0.9431**

Fig.11 RMSE values for different models and pv

The performance of different models is then compared and concluded that Auto-LSTM works the best. The primary reason for that can be it combines the feature extraction ability of the AutoEncoder with the forecasting ability of the LSTM. The choice of activation function is also crucial to getting the prediction error low by removing erroneous prediction data. An example quoted in paper is why use Relu over tanh by reasoning of reduced output of energy during winter since the former will only pass forward the value if it's positive.

9) **Paper:** Forecasting Solar Power Using Long-Short Term Memory and Convolutional Neural Networks

Deep learning allows us to deal with non-linear dependence of solar power generated with meteorological data like irradiation and temperature as against the traditional regression techniques. Along with that DL allows tolerance to error in data often arising due to faulty sensors.
Following table shows the summary of related work for prediction of renewable energy:



Ref.	Year	Forecast horizon	Method	Description
[15]	2009	5 mins~1 hour ahead	ARIMA	They evaluate various regressior such as ARIMA, Transfer functions, neural networks, and hybrid models to predict solar radiation over short time horizons. It confirms that ARIMA is the best performer since it can capture the diurnal cycle better than the other methods.
[22]	2011	24 hours ahead	ANN	It utilizes ANN for estimating the solar power of the next 24 hours in a hourly granularity. To amplify the training data, it suggests to use the day showing similar weather condition as virtual inputs for predicting the next day's power.
[23]	2012	1 day ahead	SVR	The data is first classified into four categories by the weather condition: clear, cloudy, foggy, and rainy. The conditions are then encoded as binary attributes and by using SVR, they predict power based on the weather condition and the past power generation.
[13]	2012	1 hour ahead	ARMA	Implementing the two-phase realization, the autoregressive moving average (ARMA) model has better performance predicting the future values on a micro-grid level based on solar radiation data from SolarAnywhere. Also, the solar generation data produced by System Advisor Model (SAM) supports the decision making process.
[10]	2013	1 & 3 hours ahead	SVR	Support vector regressor (SVR) is used to forecast solar power in this work; its input data contains recent records of atmospheric transmissivity and some other meteorological variables.
[14]	2013	1 hour ahead	ARIMA	This work suggests to use multi-year observation of a horizontal irradiation dataset for solar power predictions to capture the seasonal patterns; it utilizes ARIMA and the dataset is collected from a high-end device measuring meteorological values called MeteoLab.
[24]	2014	3 hours~3 days ahead	ANN	They introduce an ANN model called NARX, in which the outputs are fed back to the network with delays, for forecasting solar power with different targeting terms from 3 hours to 3 days.
[12]	2014	24 hours ahead	Ensemble	An ensemble of traditional regression and deep learning algorithms was proposed; they connect a deep belief network (DBN) and SVR by using the output of DBN as the input of SVR.
[25]	2015	24 hours ahead	ANN	Aerosol Index (AI), which measures transmissivity affected by dust in the air such as desert dust, biomass burning, volcano smoke and power plant emission, is newly considered in this work as input for solar power prediction. While the accuracy of prediction is improved slightly on cloudy days, it is rather degraded in sunny days. Unlike their argues, the benefit of using AI is appeared to be very little.
[7]	2015	1 month ahead	MLR	Analysis of variance(ANOVA) was used for multiple linear regression(MLR) analysis to forecast solar power based on European Centre for Medium-Range Weather Forecasts(ECMWF) data.
[9]	2015	1 month ahead	Ensemble	Ensemble methods was proposed obtained from the individual models such as decision tree, random forest, k-nearest neighbors (KNN), ridge regression, lasso regression, and gradient boosting. The ensemble model shows significant improvement on probabilistic forecasts compared to the individual models.
[8]	2016	1 day ahead	AE & LSTM	Experiments for solar power forecasting has conducted by various method such as multiple linear regression (MLR), deep belief network(DBN), auto encoder, and long short-term memory (LSTM). The proposed deep learning algorithms show better forecasting performance than other reference models.

Fig.12 Summary of related work for prediction of renewable energy

Introduction of cell status allows us to now remember the old information easily and hence it's no longer prone to gradient vanishing.

The following figure shows the relationship between time of the day, temperature, precipitation, irradiance.

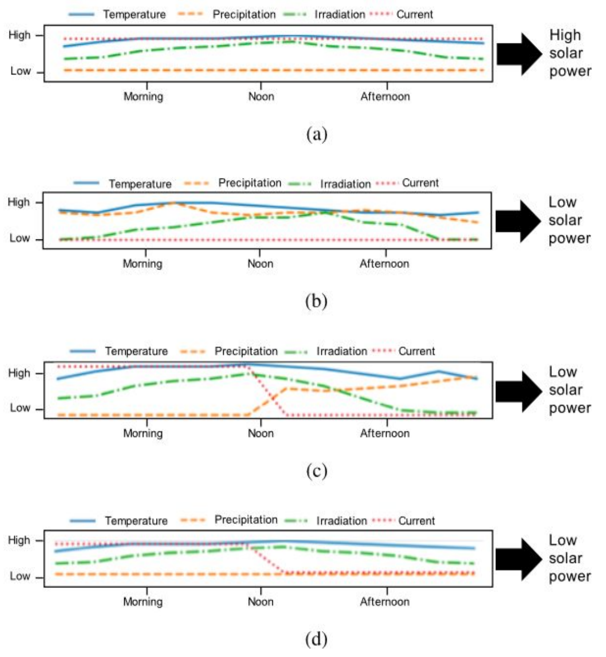


FIGURE 3. Weather condition affects solar power generation. (a) Sunny day. (b) Rainy day. (c) Sunny morning and rainy afternoon. (d) Sunny day with PV inverter malfunction.

Fig.13 Weather and its impact on solar power generation.

The following figure shows the use of LSTM with and without weather data. We can witness that MAPE is significantly less with weather data as compared to without.

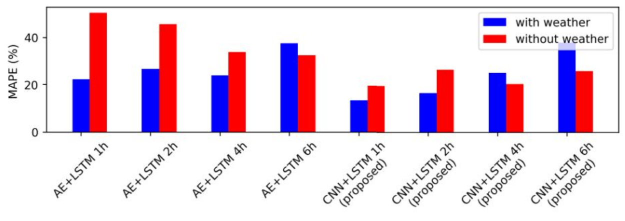


Fig.14 Weather and its impact on solar power generation.

IV. IMPLEMENTATION

In this section, we will discuss the overall implementation of the project. The implementation is broadly divided into 4 concrete phases:

1. Data gathering, cleaning and feature engineering
2. Setting a baseline model to gradually improve on.
3. Iterating over the baseline model to achieve more sophisticated models, improving the overall accuracy.
4. Design and implementation of a novel idea to improve over the initial best model achieved from the previous phase 3.

In this particular section, we will discuss phase 1, 2 and 3. However, phase 4 would be discussed in detail in section IV. Note that we would be primarily forecasting energy from solar and wind as they are the major renewable resources on which the economy of Spain relies.

Phase I: Data Collection and Cleaning

We acquired an open dataset of Spain's largest grid station for the purpose of load forecasting. However, the dataset was not sufficient to get to concrete results, for which we had to explore a similar thematic dataset and augment it. For this purpose, we used another dataset which consisted of the weather data for the same year in spain. The weather dataset consisted of real-time weather details for the major cities of Spain. After merging the dataset by taking a join on the dates, we moved to data cleaning. First we removed the columns that were entirely empty and of no use. Then we removed any outliers in the data to get more concrete and robust results. There were few outliers in the pressure and wind columns, which we initially set to NaN (Not a Number) and then recomputed via interpolation. There were also some inconsistencies in the data types which were resolved by explicit casting. Duplicate rows were dropped by keeping the first row only and categorical columns, for instance, weather description were encoded via label encoding. To augment our data, we used the date column from the dataframe and used it to derive multiple useful columns, for instance, hour, day, month, year, weekday, peak or off-peak hour etc. Finally, the dataset was normalized using standard and minmax scalar.

Our initial dataset also contains some columns which are values that were forecasted by Transmission System Operator (TSO). Though we will not use these values for our forecasting but these values can provide some reasonable indication to compare our results with. The RMSE value for TSO is 201.72 for solar generation.

Phase II: Baseline Model

Once, we gave a concrete shape to the initial dataset, we moved to define our baseline model. At this point, we had enough knowledge of the domain and had clearly framed the problem as discussed in the previous sections.

For the baseline model, we choose a simple yet comprehensive model i.e. Univariate LSTM model. Univariate LSTM is indeed one of the best choices for non-linear, time-series prediction. For our baseline model, we tend to keep a shallow architecture, as we were to gradually build more complex and deep networks later.

We fine-tuned the LSTM architecture starting with 80 hidden units, which is instantly flattened (flatten layer) and fed into a fully-connected dense layer of 160 neurons. We used the most common and effective ReLU (Rectified Linear Unit) as the activation function. Towards the end of the network we have a dropout layer which plays a significant role, avoiding overfitting of the network on training data. The network terminates with an output dense layer with a single neuron. We kept the number of epochs to be fixed at 30. More epochs might result in reduced RMSE (Root Mean Squared Error) but will tend to overfit the network to training data.

At this point, we were quite satisfied with our baseline model with an RMSE between 140 to 150. Our baseline model instantly surpassed forecasting by TSO.

Phase III: Iterating over the baseline model

Our baseline model proved to work well as the RMSE fluctuated between 140 to 150, which was 25% less than the RMSE of TSO forecasting. However, we believed that we had just scratched the surface and even more accurate forecasting was conceivable.

(i) Univariate CNN-LSTM

From that onwards, we moved to more complex and deep hybrid architectures. We started with a hybrid combination of univariate CNN-LSTM network architecture. This hybrid CNN-LSTM network is not something new or complicated, it simply combines the feature extraction capabilities of CNNs with the sequence analysis capabilities of LSTM networks. Such network architecture has proven to perform effectively, especially for time series modelling as discussed in the previous section.

The design and network architecture of this hybrid model starts with a 1-Dimensional convolutional layer, more specifically, 100 filters with a kernel size of 2 and causal padding. The activation function was kept to be ReLU as discussed earlier. We also tried other activation functions such as sigmoid and softmax but they didn't turn out to be that effective as ReLU. We concluded that this is because ReLu holds the property of non-saturation of its gradient, which greatly accelerates the convergence of stochastic gradient descent compared to the sigmoid or any other activation function.

From hereafter, we kept ReLU as an activation function for the subsequent network architectures. The ReLU activation layer is succeeded by a LSTM layer with 100 hidden units and a similar flatten layer. Towards the end of the network we have 50 hidden units with once again ReLU being the activation function as we finally move towards the last output layer consisting of a single unit. The RMSE for univariate CNN-LSTM turned out to be approximately in the range 160 - 170.

The univariate CNN-LSTM architecture did not go as per our expectation. We tried to fine tune the network by changing the kernel size and stride value. We even tried to change the padding type but the network did not provide any better results from our base model. At this point we set our hypothesis that CNN-LSTM hybrid architecture does not perform well with univariate features. Therefore, we moved to multivariate features to accept or reject our hypothesis.

As stated earlier, so far we were only dealing with univariate networks that were particularly designed for time series modeling. Since we had more data and hence more features in hand, we could certainly bring them into the picture for more accurate and precise results, together with lower value for RMSE.

(ii) Multivariate LSTM

Our first multivariate architecture would be a simple LSTM based network. For which, first we were required to find which all features shall we consider to be fed into the network. This is not a non-trivial problem. All we need to do is check correlation of features with our target variable, which in our case is solar generation and wind generation. Recall that correlation is a statistical measure of how two or more things change together. The value gives the magnitude of the relation while the sign i.e. positive or negative gives the direction. We used pearson correlation to investigate the features that were highly correlated with solar and wind generation. We also generated the correlation matrix to quickly visualize correlated features.

Once we had realized our correlated feature with the target variable, we moved to generate the architecture of the multivariate LSTM network. The architecture of multivariate LSTM was kept exactly similar to that of univariate LSTM except that now we have more features to be fed into the first layer of the network. Following is the overall network architecture of multivariate LSTM from input to 1 unit dense output.

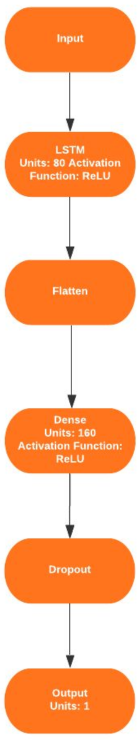


Fig.4.1: Overall Architecture of Multivariate LSTM

With a multivariate LSTM network the RMSE dropped drastically with an approximate range between 110 - 120. At this stage, our multivariate LSTM network successfully surpassed the forecasting of TSO by 48%. All we needed then was to design one more network architecture multivariate CNN-LSTM to confirm whether we could further increase our accuracy and subsequently reduce the RMSE or not. Nonetheless, we also wanted to accept or reject our prior defined hypothesis that CNN-LSTM do not work well with univariate features for forecasting.

(i) Multivariate CNN-LSTM

Our last iteration would be to design and implement multivariate CNN-LSTM architecture. The architecture was kept similar to the univariate CNN-LSTM but this time we will have multivariate features for the time series forecasting. Turned out that the multivariate CNN-LSTM worked the best among the 4 architectures we have designed and implemented thus far. With a multivariate CNN-LSTM network the RMSE dropped within an approximate range between 100 - 110. We will consider this as our best initial model for the purpose of forecasting solar and wind generation. This also provided us with some ground to accept our prior constructed hypothesis that univariate CNN-LSTM does not work well for time series forecasting especially for solar or wind generation.

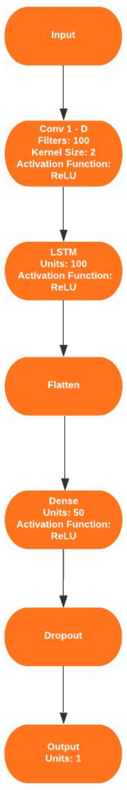


Fig.4.1: Overall Architecture of Multivariate CNN - LSTM

Solar Production Forecasting	
Model/Network	**RMSE**
TSO	201.7
Univariate LSTM	140 - 150
Univariate CNN-LSTM	160 - 170
Multivariate LSTM	110 - 120
Multivariate CNN-LSTM	100 - 110 **(Initial Best)**

Table 3.1: Summary of RMSE for all implemented network architecture

IV. PROPOSED IDEA (NOVELTY)

Now that we have in depth complete knowledge of Deep Neural Networks, how exactly they work and how to leverage them for the purpose of forcatsing, our goal for this section would be to brainstorm new ideas and present a concrete novel scheme(s) that is not only a step ahead of our best initial model but also applicable in the real world for forecasting. This is indeed the most challenging and non-trivial task.

We experimented with some of the popular algorithm picking techniques such as weighted majority algorithm which is a deterministic selection algorithm and randomized weighted majority algorithm which is non-deterministic in nature. We closely investigated each of these to see if we could leverage and integrate them with Deep Neural Networks under one single unified framework. To our best knowledge and the extensive literature review carried, thus far, we have not seen anyone using these methodologies under a single unified framework for forecasting.

1. <u>Deterministic Weighted Algorithm Selection</u>

First we will briefly discuss deterministic weighted majority algorithm selection. The idea behind the algorithm is quite simple, yet it can lead some powerful results. We take N experts for our forecasting purpose, where each of the N experts are weighted equally in the start e.g. 1. Each day we query our experts to predict/suggest some values and record their predictions/suggestions. We then take the final decision at T0, which is voted by the majority of experts. However, if the domain of the problem is continuous, then we take the decision of the expert with the maximum weight. As we move on to next time frame i.e. T1, we check which expert predicted/suggested the ground truth correctly. If an expert predicted/suggested the value correctly or at least nearest to the ground truth if the problem domain is continuous, we do nothing with the weight of that expert. However, we will reduce the weights of all the other experts who did not manage to predict/suggest the values correctly for T0. This is more formally referred to as punishing. This way as we move along the time series. We will trust more on the experts that have produced most accurate results and least on experts who made more mistakes as we proceed.

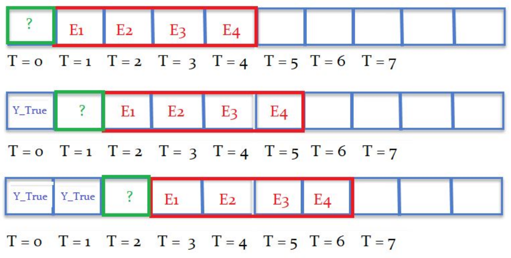


Fig. 4.2 Windowing Weighted Majority Algorithm

Extending this approach we have taken 4 experts for our purpose of forecasting. The 4 experts here are the windows at T1, T2, T3, T4, which will execute all the available models to predict the values at T1, T2, T3 and T4. In our case each expert or each window will execute 2 pretrained models namely multivariate LSTM and multivariate CNN-LSTM to forecast electricity, then they will pick one model of choice to be executed at T[i] timeframe based on the minimum regret. Finally, when experts are ready with their decisions, we will look at the current weights of the experts, and take the final decision of that expert which has the highest weight at a given time. As we slide, the next day, we will compare our expert's prediction with the true value of Y. If the expert was incorrect i.e. another expert was more close to the true value, we punish this expert by reducing its weight by W(i) = W(i) x (1 – e). Where epsilon (e) is,

$$\varepsilon^{*} = \sqrt{\frac{\log N}{M_T(i)}}$$

Mt(i) = Number of mistakes by i-th expert at T timeframe.

2. <u>Random Weighted Algorithm Selection</u>

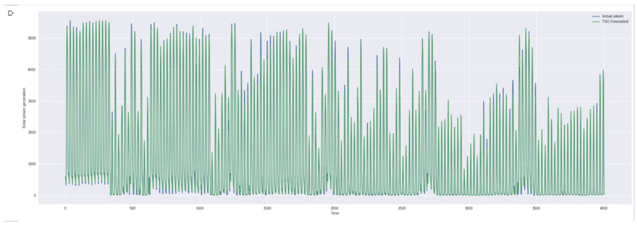
Next we investigated the randomized version of the weighted algorithm selection. The idea is very similar to what was discussed earlier, however the only difference here is that here we do not go with the expert that has the highest weight at a given time frame. Instead, we generate a random number between 0 and 1. Then we select the decision of the expert to whomes the probability interval the random number takes. Since each of the experts have different weights, their probability distribution is not uniform. This means that there is a greater chance to land on the probability interval of the expert with greater weight but we may also land on a probability interval of another expert whose weight is less. Therefore, the selection is random with some bias due to weights and non-uniform probability distribution.

As we move along, we update the weights of the experts given they provided the correct decision or not from the ground truth. If the decision did not match with the true value i.e. some other expert predicted a closer value, then we penalized its weight by W(i) = W(i) x (1 – e).

The results for the combination of random weighted algorithm selection and deterministic algorithm selection with DNNs surpassed our initial best model (multivariate CNN-LSTM). The RMSE for both of these techniques range around 90 - 95 which is ~ 15% lesser than our initial best model and ~ 35 % lesser than our baseline model and ~ 100% lesser than Transmission System Operator (TSO) forecast provided in the data.

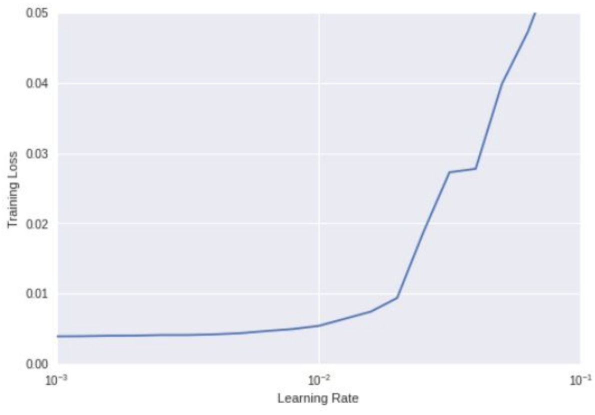
V. PRELIMINARY RESULTS

Solar Production Forecasting	
Model/Network	**RMSE**
TSO	201.7
Univariate LSTM	140 - 150
Univariate CNN-LSTM	160 - 170
Multivariate LSTM	110 - 120
Multivariate CNN-LSTM	105 - 110 **(Initial Best)**
Deterministic Weighted Algorithm Selection	93.62 (Novel 1)
Random Weighted Algorithm Selection	93.28 (Novel 2) **(Best)**

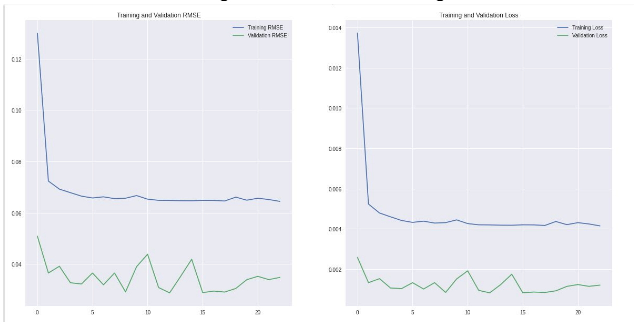


Transmission System Operator Forecast Given in Dataset (TSO)

1 . Univariate LSTM

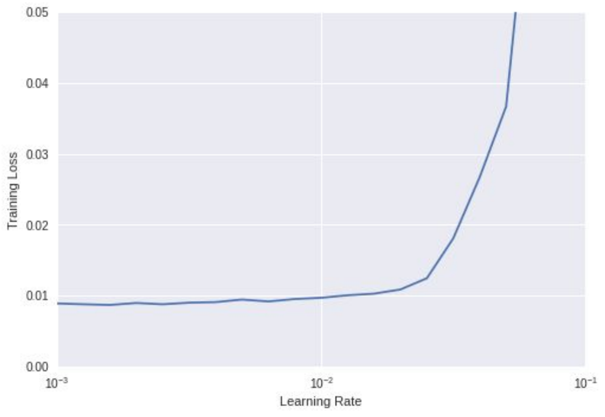


Training loss v/s learning rate

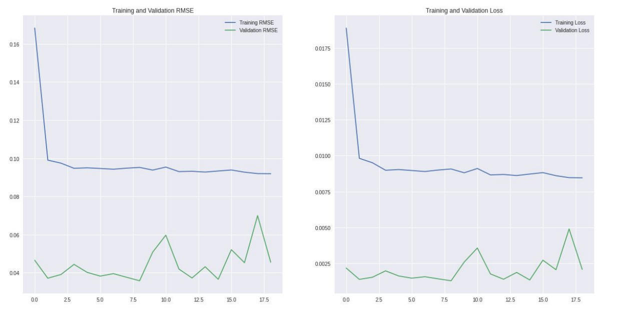


Left - Training and validation loss RMSE
Right - Training and validation loss

2. Univariate CNN-LSTM

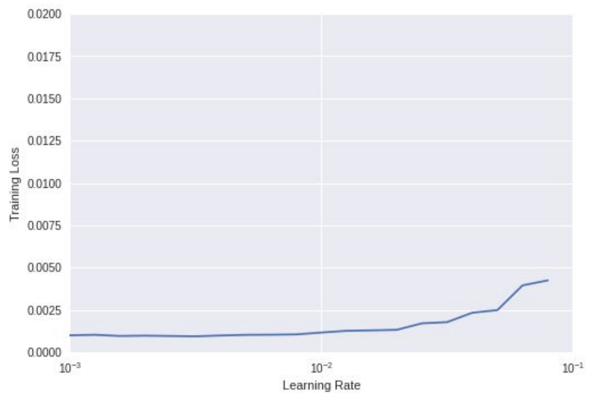


Training loss v/s learning rate

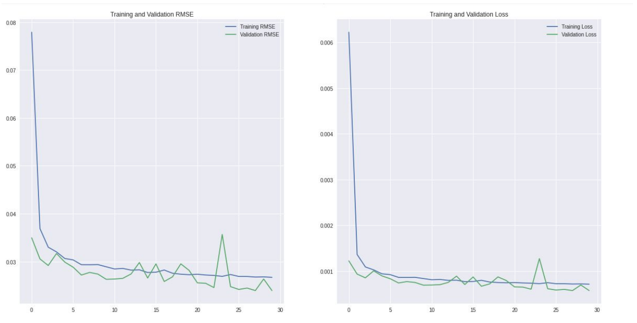


Left - Training and validation loss RMSE
Right - Training and validation loss

3. Multivariate LSTM

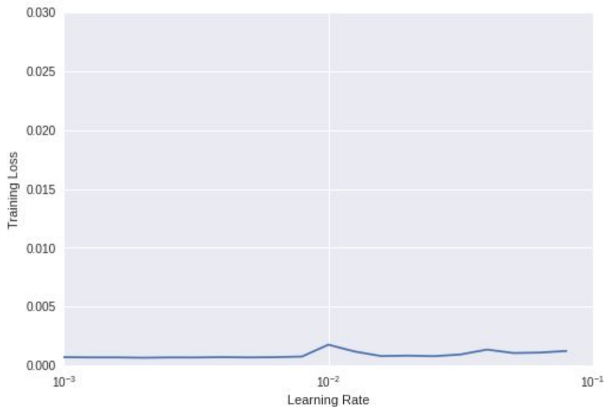


Training loss v/s learning rate

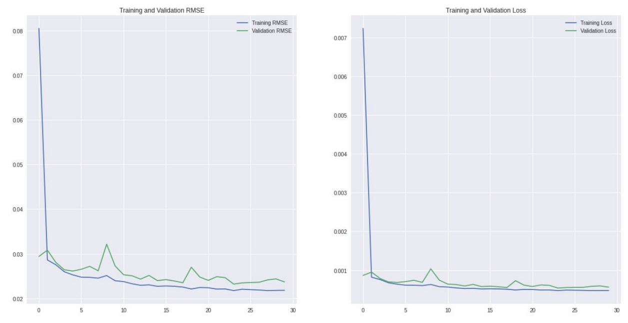


Left - Training and validation loss RMSE
Right - Training and validation loss

4. Multivariate CNN- LSTM



Training loss v/s learning rate



Left - Training and validation loss RMSE
Right - Training and validation loss

VII. SUMMARY

Our main aim is to predict the short term forecast for renewable energy sources like solar. Since the data was time sequence, deep learning techniques like CNN and LSTM came to be of utmost importance. Later, the hybrid technique of CNN-LSTM also showed some huge potential in the prediction process. We started off by working with univariate CNN-LSTM for which the RMSE came out to be around 185. This was overpowered by the univariate LSTM for which the RMSE was 138. As we moved onto the multivariate domain, we saw that the multivariate LSTM showed a far greater prediction power (RMSE 112) than the univariate counterparts. Ultimately, the winner came out to be the multivariate CNN-LSTM hybrid which recorded the lowest RMSE of 110. This was our baseline model which showed some promising results.

In order to even make our predictions more accurate, we incorporated a novel approach of using **Weighted Majority Algorithms (WMA).** The WMA'S two variants,

1) Deterministic Weighted Majority Algorithm
2) Randomized Weighted Majority Algorithm

WMAs are an effective technique to create an ensemble or a compound algorithm. The main concept is that instead of relying totally on one algorithm to quench our needs, we employ a tactical combination of two or more of the best performing algorithms. The deterministic variant initialises with constant weights for each of its agents (weight 1 in our case). The randomized variant assigns random weights to our agents. Following the principle of punishing the under performing agents, whilst uplifting the best performing agents, we get the appropriate amount of weights assigned to each of our agents. The weights signify the amount of importance which needs to be given to each agent keeping in mind to minimise our objective function of minimum RMSE in mind all the time.

So, when we applied the deterministic WMA, we saw a sudden reduction in the RMSE which came out to be 93.62, one of the lower RMSEs recorded till now. This was even beaten by the randomized WMA for which the RMSE came out to be 93.28. Such promising results helped us discover the power of weighted majority algorithms in the sequence data forecasting. This makes intuitive sense that the algorithms which are performing the best in specific spectrum of the sequence must be given more importance than the ones which are underperforming. Hence, the best performing algorithms in their best performing portions when integrated give us a hybrid algorithm which can outperform their individual counterparts.

VIII. REFERENCES

1) *CNN - Convolutional neural network*
2) *CNN and LSTM - Electricity Theft Detection in Smart Grid Systems: A CNN-LSTM Based Approach*
3) *LSTM - Essentials of Deep Learning : Introduction to Long Short Term Memory*
4) *Figure 2 source - Understanding LSTM Networks*
5) *CNN LSTM architecture - CNN Long Short-Term Memory Networks*
6) *DNN - https://ieeexplore-ieee-org.proxy.library.stonybrook.edu/document/8001465*

IX. Responsibility Division

Gaganraj Maheshwari	Literature review: #4, #5, #6 Section: Abstract, Motivating examples Implementation: Naive model, Univariate LSTM Univariate CNN-LSTM
Vibhor Shukla	Literature review: #1, #2, #3 Section: Introduction, Summary Implementation: Multivariate LSTM Deterministic Weighted Majority Algorithm
Uzair Bin Tariq	Literature review: #7, #8, #9 Section: Implementation, Results Implementation: Multivariate CNN-LSTM Randomized Weighted Majority Algorithm
Common Tasks	Novelty, Ideas, Brainstorming, Proposed Idea