Electricity Load Forecasting of IIITD's campus

Fahad Nayyar
IIITD
fahad17049@iiitd.ac.in

Krishna Kariya
IIITD
Krishna17060

Vrinda Mittal
IIITD
Vrinda17297@iiitd.ac.in

Abstract

Load prediction can be very central and very helpful for power management and planning. In our country, several cities face the issue of load shedding and irregular electricity hours. Predicting load could help authorities manage their energy utilization and plan power back up efficiently. We plan to predict overall power consumption on IIITD's campus. Our dataset includes time series data of average hourly power consumption for past 1 year of 14 different buildings on campus. We are also using average overall power consumption data at 5 minute granularity for the months of aug-sep. We have done a comparative analysis of 4 different models for time series data: ARIMA, RNN, LSTM, GRU. We will also show potential of a simple linear regression for this problem.

Keywords: *Machine learning, Time series data, LSTM (long term short memory), RNN (recurrent neural networks), GRU (gated recurrent units), ARIMA (auto regressive moving average).*

1. Introduction

With the increase in global warming, issues such as effective management of electricity is becoming increasingly important. Electricity Load prediction can help various state Governments to plan their power budget. Another use case is for power back-up management. It becomes tricky to put a threshold on how much power should be stored. Predicting electricity load for the next day could help authorities decide how much power should be generated and stored.

There is also risk attached to power leakage, fires due to overloading, etc. If we can precisely predict electricity load at the granularity of an hour or 5 minutes we can probably make good anomaly detectors to lower the risks of power leakage, fires due to overloading, etc.

For this project we want to tackle the problem of electricity load prediction at institute level. For this we will be using the average electricity power consumption data of 14 buildings in IIITD's campus and use "machine learning time series data forecasting techniques".

Since demand patterns for electricity are very complex and depend on various factors like seasonality (temperature, humidity, wind speed), cost of electricity, etc, there is no generalized or state-of-the art solution for electricity load forecasting. Our analysis could help recognize which forecasting technique can work best for IIITD's campus.

We will focus on 3 sequential models RNN, LSTM, GRU and one state-of-the-art time series model ARIMA. We have also done 24 separate linear regressions for each hour of the day in order to predict average electricity load at each hour, given the load of the previous hour.

2. Related Work

We have taken motivation for this problem from the two projects discussed on **[1]** and **[2]**. **[1]** has done time series forecasting on electricity data for a house using sequential linear regression and ARIMA. **[2]** has done a comparative study between 7 different models to predict electricity load on data obtained from State Load Dispatch Centre (SLDC), Delhi.

We want to do electricity load forecasting for the different buildings in our college based on recently collected data by meter readings maintained by FMS. To the best of our knowledge, this data has not been used so far such a problem analysis so far.

We referred to **[3, 4]** videos to get a better understanding about RNN, LSTM, GRU and **[5]** for ARIMA.

3. Methodology

3.1 Dataset.

We are using 2 different datasets.
For comparative study of RNN, LSTM, GRU and ARIMA (as discussed in section 3.2) we have used average electricity load at 5 minute granularity for the months of August-September (till 29th sep). On the basis of model trained we predict the load on 30th september.

Size of dataset: 60 X 24 X 12 = 17280 data points.

For linear regression we have used hourly average power consumption data of past one year.

Size of dataset: [14] x [8764]
[building] x [time points]

3.2 Comparative study.

In this section we will describe our comparative study of lstm, rnn, gru and arima models.We are showing predicted load vs time graph and RMSE error for analysis of each model.

Data Pre-processing: The data was made stationary by detrending using "One lag dereferencing" and was rescaled to [-1, 1] scale. This technique is followed for RNN, LSTM and GRU. These models were trained on data of months of August-september. We made each training data vector containing load of a specific time for last 10 days and the label was the load of the $11th$ day at the same corresponding time. So in all we have a matrix of dimension [17280]X[11] (load at a point of dat X load on same point of day in past 10 days).

A. Recurrent Neural Networks (RNN)

It is the state of the art ml algorithm for sequential data. It remembers its input due to an internal memory. In RNN, the information cycles through a loop. When it makes a decision, it takes into consideration the current input and also what it has learned from the inputs it received previously.

Model Architecture: The model consisted of two layers one with two simple RNN cells with hyperbolic tangent function (*tanh*) as activation function and followed by a dense layer without any activation. Metric used for loss was mean square error optimized with Adam optimizer with a learning rate of 0.001. The model was trained for 50 epochs. The code was written in Python using Keras library. We took this model from **[2]**.

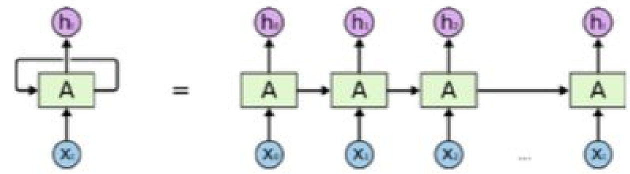


FIg. 1: RNN Architecture.

B. Long short-term memory (LSTM)

Long Short-Term Memory (LSTM) networks are an extension of recurrent neural networks.
LSTMs enable RNNs to remember their inputs over a long period of time. This is because LSTMs contain their information in a memory, from which they can read, write and delete.
This memory can be seen as a gated cell, where gated means that the cell decides whether or not to store or delete information (e.g if it opens the gates or not), based on the importance it assigns to the information. The assigning of importance happens through weights, which are also learned by the algorithm. This simply means that it learns over time which information is important and which not.
In an LSTM you have three gates: input, forget and output gate. These gates determine whether or not to let new input in (input gate), delete the information because it isn't important (forget gate) or to let it impact the output at the current time step (output gate).

Model architecture:
The model for LSTM consisted of two layers one with one LSTM cell with hyperbolic tangent function (*tanh*) as activation function followed by a dense layer without any activation. Metric used for loss was mean square error optimized with Adam optimizer with a learning rate of 0.001. The model was trained for 50 epochs. The code was written in Python using *Keras* library. We took this model from **[2]**.

Model Architecture 2:
This model consisted of 5 layers ; 2 LSTM layers each followed by a dropout layer and finally connected to a dense layer. Metric used was loss was mean square error optimized with Adam. The model was trained for 10 epochs. We used this model to see forecasting of individual building data. After our analysis of Average power data.

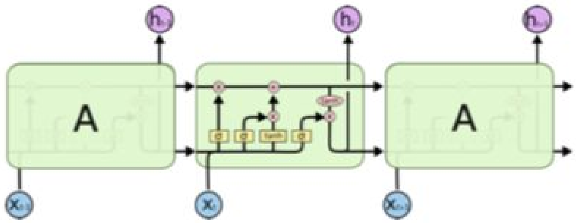
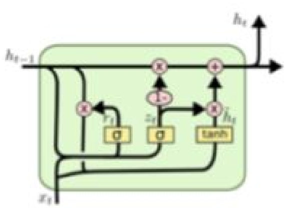


FIg. 2: An LSTM cell.

C. Gated Recurrent Unit (GRU)

A slightly more dramatic variation on the LSTM is the Gated Recurrent Unit, or GRU. It combines the forget and input gates into a single update gate. It also merges the cell state and hidden state, and makes some other changes. GRU ultimates has lesser features to learn and is more efficient.

Model Architecture: The model consisted of two layers one with one GRU cell with hyperbolic tangent function (*tanh*) as activation function followed by a dense layer without any activation. Metric used for loss was mean square error optimized with Adam optimizer with a learning rate of 0.001. The model was trained for 50 epochs. The code was written in Python using *Keras* library. We took this model from **[2].**



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

FIg. 3: A GRU cell.

Model Architecture 2:
This model consisted of 5 layers ; 2 GRU layers each followed by a dropout layer and finally connected to a dense layer. Metric used was loss was mean square error optimized with Adam. The model was trained for 10 epochs. We used this model to see forecasting of individual building data. After our analysis of Average power data.

D. AutoRegressive Integrated Moving Average (ARIMA)

An ARIMA model can be understood by outlining each of its components as follows:
Autoregression (AR) refers to a model that shows a changing variable that regresses on its own lagged, or prior, values.
Integrated (I) represents the differencing of raw observations to allow for the time series to become stationary, i.e., data values are replaced by the difference between the data values and the previous values.
Moving average (MA) incorporates the dependency between an observation and a residual error from a moving average model applied to lagged observations.

It requires 3 parameters p, d and q which we got from grid search. The details are:

p: the number of lag observations in the model; also known as the lag order.
d: the number of times that the raw observations are differenced; also known as the degree of differencing.
q: the size of the moving average window; also known as the order of the moving average.

Model Architecture: After getting hyper parameters from grid search CV, we used python's statsmodel library for predictions.
We took this model from both **[1]** and **[2[**.

E. Linear Regression

Preliminary data Analysis
We plotted Load v/s Hour graph for each of the 14 buildings. For this we picked up 10 random days from dataset, and plotted a graph for it. It can be viewed from the graph that there are jumps in values of Load at hour h and hour h+1. Thus, there is a linear relationship between the load at hour h and hour h+1 at a particular day. The same relationship can be viewed for all the buildings including the plot for total power (FIg. 4).

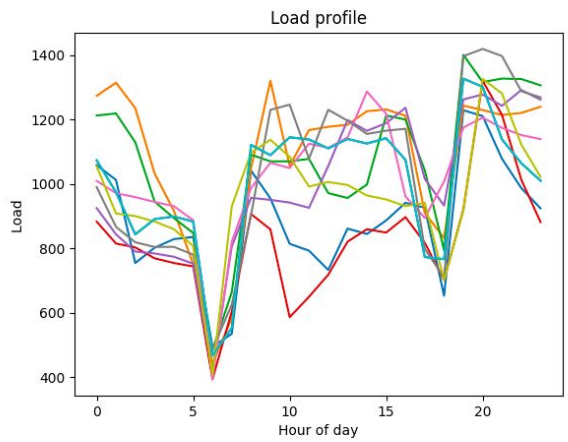


Fig.4 Load profile for average electric power

Model training and testing.
After the analysis of the data, it was clear that there is a linear relationship between the load at hour h and hour h+1. So, we thought of making 24 models for each hour prediction. The input (features) of a model is the Load at hour h-1 and output is the Load at hour h. We applied Lasso and Ridge Regression with Linear Regression. We found the hyper parameters for Lasso (0.64) and Ridge (0.01) using GridSearchCV. We made a scatter plot with line graph for each of the model (24*3 models) for 14 buildings.
xwww

4. Results

4.1 Comparative study results.

A. Recurrent Neural Networks (RNN)

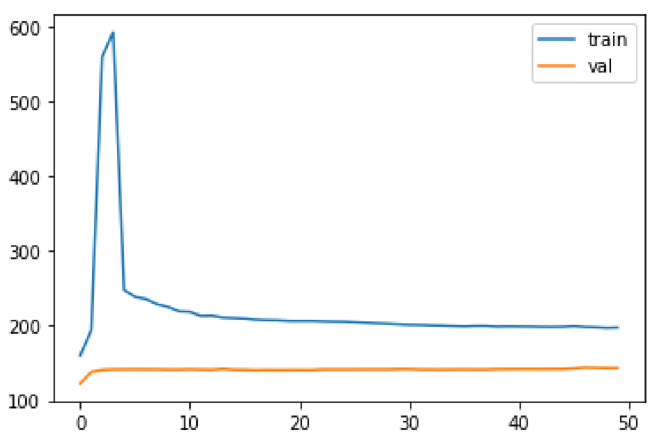


Fig. 5: RNN error vs iterations curve.

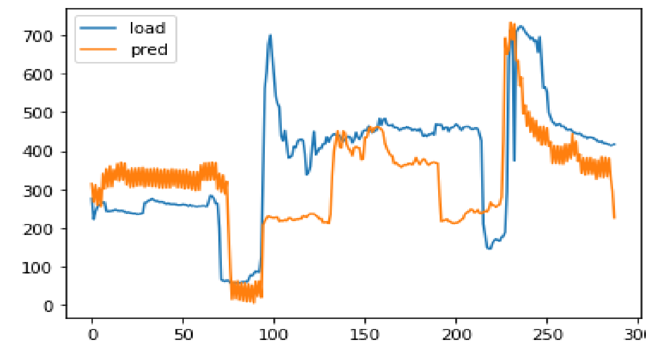


Fig. 6: RNN prediction and actual load on 30th september.

B. Long short-term memory (LSTM)

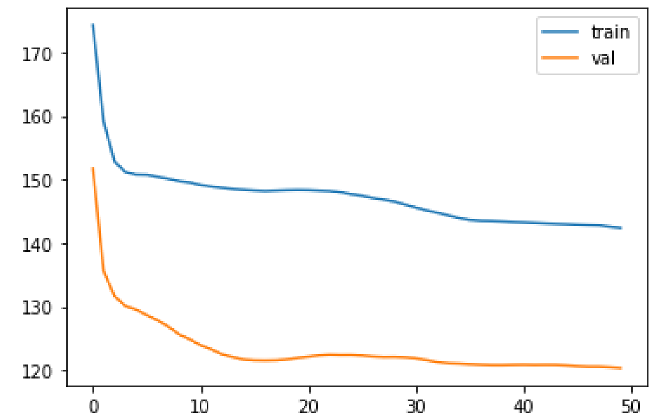


Fig. 7: LSTM error vs iterations curve.

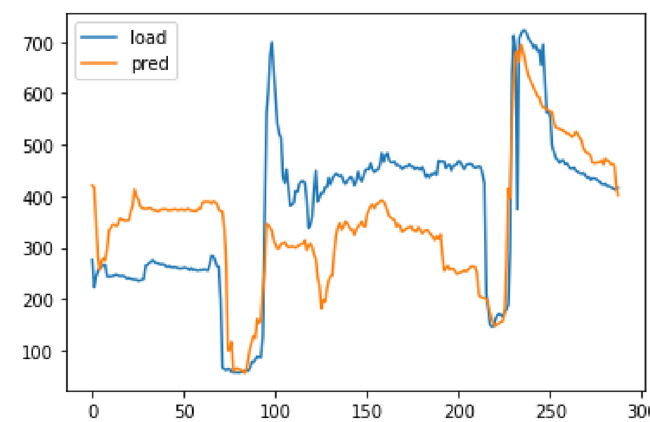


Fig.8:LSTM prediction and actual load on 30th september.

C. Gated Recurrent Unit (GRU)

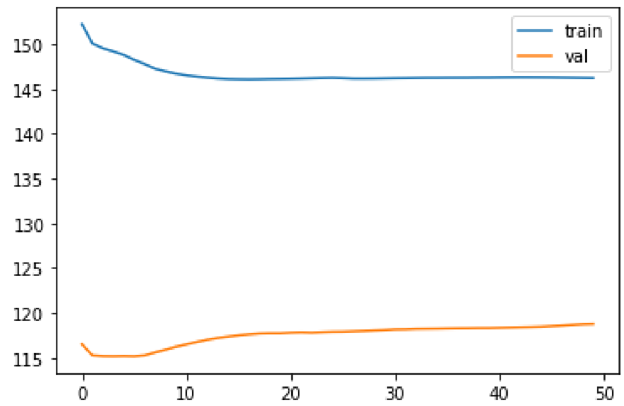


Fig. 9: GRU error v/s Iteration curve

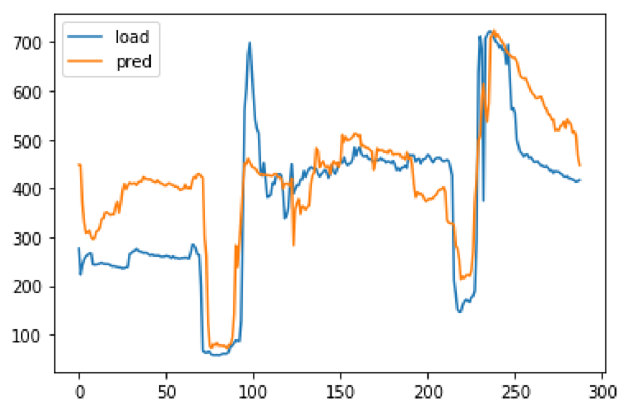


Fig. 10: GRU prediction and actual load on 30th september.

D. AutoRegressive Integrated Moving Average (ARIMA)

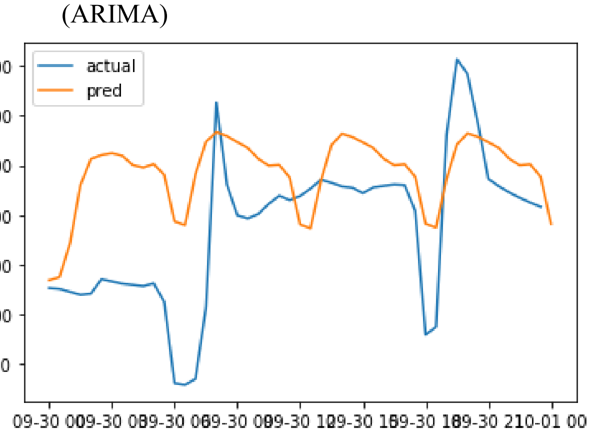


Fig.11: ARIMA prediction and actual load on 30th september

RMSE table:

Model	RMSE
RNN	144
LSTM	125
GRU	104
ARIMA	169

Table. 1: RMSE of each model

E. Linear Regression Results.

For some of the buildings, the linear regression model fitted the dataset very well, while for some of the buildings it was bad fit.

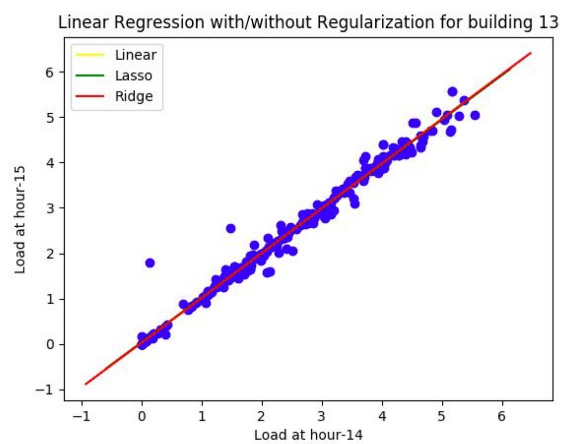


Fig. 12: An example where linear dependence holds.

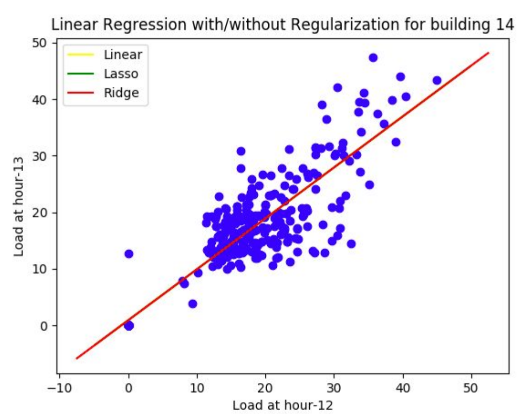


Fig. 13: An example where linear dependence doesn't hold

For some specific hours, the data is not linearly dependent on previous hour. Error for some cases is very high in case of linear model.

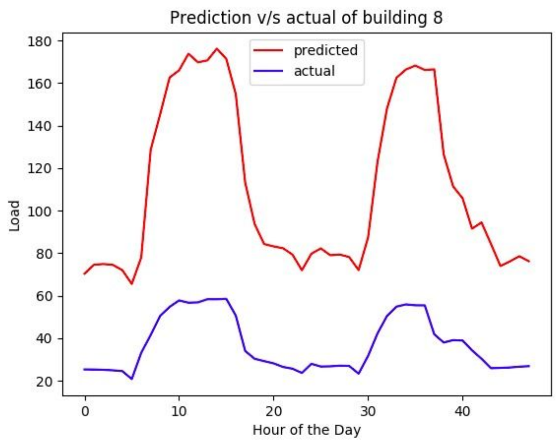


Fig. 14: Forecasting using (sequential) Linear Regression

F. Results on LSTM and GRU models used for individual buildings

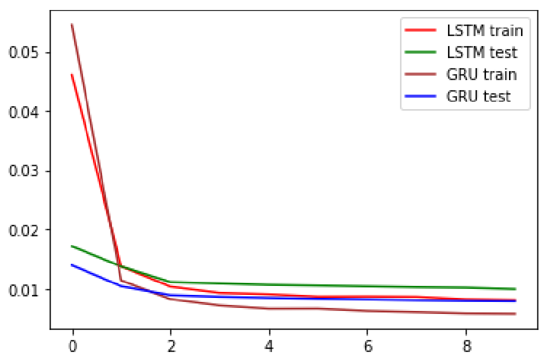


Fig. 15: loss v/s epoch (IIITD Girls Hostel)

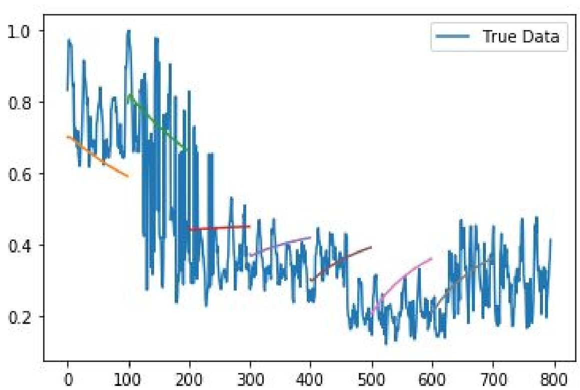


Fig. 16: LSTM forecast v/s true data (IIITD Girls Hostel)

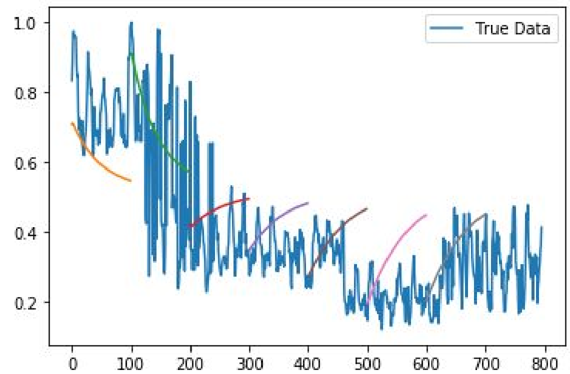


Fig. 17: GRU forecast v/s true data (IIITD Girls Hostel)

5. Conclusions

Recurrent neural network models (rnn, LSTM, GRU) perform better as compared to other techniques(Linear regression, ARIMA). GRU gives best results for electricity load pattern at IIITD.

Linear regression is also a good and simple model if we want to predict the next hour's usage based on current hour's load.

If we are successful in predicting load across various buildings at IIITD then we may be able to devise good anomaly detection schemes based on self attonation of data or thresholding techniques to make our campus more secure and safe from fires due to overloading.

6. References

[1] http://cs229.stanford.edu/proj2017/final-r eports/5206915.pdf
[2] https://github.com/pyaf/load_forecasting
[3] https://www.youtube.com/watch?v=9TFnjJkfqmA

[4] https://www.youtube.com/watch?v=6_MO12fPC-0

[5] https://www.youtube.com/watch?v=yZ0g-DIfVpc