Multi-Purpose Forecasting Model using Deep Learning

Khushhal Jindal
Dept. of Computer Science
University of Central Florida
Orlando, Florida 32816
khushhaljindal@knights.ucf.edu

Rohan Singh Rajput
Dept. of Electrical & Computer Engineering
University of Central Florida
Orlando, Florida 32816
rohan_rajput2804@knights.ucf.edu

***Abstract*—Advancements in the field of machine learning has shown great promise in effectively solving problems that involves a large amount of data. Forecasting is one such problem type in which good amount of data is generally present in order to discover trends, however relationship information in the data is unknown. In the recent times, there has been an enormous surge in employing Deep Neural Networks for forecasting due to their ability to capture nonlinear relationships from data. The focus of this project is to implement a multi-purpose forecasting model that works for varied time-series data by using Recurrent Neural Network (RNN) and its variants LSTM and GRU. Aforementioned models are evaluated on the metric root-mean-square error. In addition, we evaluate our models' performance by comparing with the results on diverse time-series datasets. We are able to achieve noteworthy results for varied datasets decreasing the error rate to a good extent.**

I. Introduction

Deep Learning is a one of the popular field in machine learning. It has shown a great capability towards solving complex computer vision and Natural Language Processing tasks.[?] It is also expanding in data science for prediction and forecasting. Time series is an important aspect in forecasting.[?] However, Time Series forecasting is a complex task and it is considered a difficult type of problem. It has many applications like in health care, weather, financial market etc. As it is very important and complex model to analyses and predict. For example weather is a important data to analyze and it depends on so many factors that some times it gets extremely difficult to analyze. Financial Market also depends on many factors and many times we do not have not enough correlated data which can help in extracting relevant information from it. Due to the complexity involved in implementing neural network with parameter tuning, and due to the simplicity of statistical models, many of the time-series forecasting problems are solved by using statistical techniques such as moving average, exponential smoothing which may seem to work well for a short-term, however they fail miserably for long-term as their capabilities to identify patterns from past data fades away. Also, they fail to account for the uncertainty in data. There are several other statistical techniques such as Autoregressive moving average (ARMA) which lacks in determining the degree of differencing needed to remove non-stationarity in data. These techniques assume can also suffer from false positives if data is integrated to the positive order. Essentially these techniques struggle to work well in identifying trends and forecasting as they are based on the strong assumption that existing patterns will continue go on in the future as well. Neural Networks have proved to can determine nonlinear relationships between inputs and outputs. These networks are also capable to find optimal indicators in order to construct favorable forecasting strategy. The advantage of using Recurrent Neural Networks is that they are adaptive; can learn to process arbitrary sequence of inputs and identify patterns. Input variables in time series problems often suffer from the complex sequence dependence among them and this can be solved by using Long Short-term Networks (LSTM) and Gated Recurrent Unit (GRU) networks, variants of RNN which makes it easier to train large architectures to achieve effective results. After implementation of these networks, hyperparameter tuning is performed to achieve better forecasting results and to gain better understanding about which parameters are influential in an accuracy change.

The paper is organized as follows: Section 2 states the Data Set information including the Data wrangling process done on it for significant feature extraction; Section 3 states our proposed approach where we provide a detailed explanation of our implementation. In Section 4, we present our experiments and Section 5 states some of our results; this is followed the Conclusion and Future Scope in Section 6.

II. DataSet Information

We have used Electric Power Consumption Dataset of an individual Household from UC Irvine Machine Learning repository[?]. It contains around 2 million measurements between the period December 2006 and November 2010 which is a period of around 47 months. Measurements have one-minute sampling rate. It has total nine attributes; different electrical quantities and some sub-metering values are available.[?]

A. Data Wrangling

Data Wrangling is one of the most significant process associated with any analysis and it becomes even more significant to convert data into a useful form depending on the model we are applying to solve a specific data problem. It helps not only in that aspect but also proves to be of some help when deciding about which model to use. It involves collecting data from a source and then analyze it to get better insights about the data and its format type. Then it involves a tedious process of Data Cleaning, when done improperly, can result in inaccurate analysis leading ineffective insights. So, there is a great need to assess data quality and make sure the data is consistent. Although various tools are widely available to perform specific tasks in the whole process, availability of a complete solution is very limited for the whole data wrangling process. And it is also considered as the most time-consuming aspect of any analysis task.

1) Data Collection: Data has been spread over for around 47 months with a total of around 2 million data points. Since, we are dealing with a lot of data values, we wanted to make sure that we are using the least erroneous data range so as to better analyze our model as well as its accuracy. We choose an year range January 1st 2007 to December 31st 2007 for our evaluation as yearly forecasting of power consumption is a better metric rather than doing it over the whole dataset. This reduces the data points to 525,600 for which we have done most of our work.

2) Data Cleaning: It involves dealing with missing data, inconsistent values. Data must be transformed and cleaned into a usable state so as to get better insights. There are about 8000 missing values in the dataset after reducing the points to 525,600 which made it a very significant task to achieve accuracy in forecasting. There are various ways to handle missing data, by deleting the missing data entries or just ignoring the data, both of which are not a good practice as it could lead to skewness in data which makes it inconsistent for analysis. One better approach to handle missing values is imputation, where all data values are imputed over the mean.

3) Data Manipulation: Due to very little changes in the household power consumption over a minute sampling, data preparation becomes really important as we need to rescale the data to achieve better insights and make it useful for forecasting. Data has to be prepared in a way so as to maximize the models accuracy before feeding it into the network. In our dataset, since each measurement in data is separated by same interval of a minute, time attribute is an important to consider. We scale data values according to time in order to have more relevant data so as to gain better insights about the forecasting. Time-scale is changed in three ways; first by taking an hourly sampling rate by updating the corresponding Global Active Power attribute values which states the power consumption on hourly basis reducing the data points from 525,600 to 8760, these numbers describing the minutes and hours in an year respectively. Second manipulation was for half-day period, 12 hour sampling which further reduced the data points to 730. Finally, we made the sampling for 24 hour period which gives us daily power consumption reducing the data points further to 365. Thus, we have three sets of data points which makes it helpful for us to gain better understanding about specific trends in data and also to use it for the comparison of training time of our model as data points are varied. For the manipulation task, pandas library in Python is used.

B. Feature Extraction

Feature extraction is really important as extracting relevant features not only makes it faster to process the data but also improves accuracy to a significant extent. There are times when a lot of the features are redundant and it is better to transform your data into a reduced set of features. In our dataset, there are nine attributes present. For improving the forecasting accuracy, inputs are needed to be made statistically independent. Global Active Power[?] values are selected for load forecasting as it contributes to the actual energy consumption of a household. Rest of the attributes since not exactly contributing to the power consumption, are dropped in order to make the forecasting more accurate and less time-consuming. We use Pandas library in Python for feature extraction by dealing with all the data using pandas dataframe.

C. Other Data Sets

1) Mean daily temperature: We take mean daily temperature in degree celsius for Fisher River near Dallas between the period Jan 01, 1988 to Dec 31, 1991. Metrics of the Data set is 1461 fact values in 1 timeseries. Data set is collected from the Time Series Data Library which was created by Rob Hyndman, Professor of Statistics at Monash University, Australia[**?**].

2) Daily foreign exchange rates: We take daily foreign exchange rates whose indicator is German/US between the period 31 December 1979 and 31 December 1998. Metrics of the Data set is 38192 fact values in 8 timeseries. Data set is collected from the Time Series Data Library which was created by Rob Hyndman, Professor of Statistics at Monash University, Australia[**?**].

3) Total Loans and Leases of Commercial Banks: We take dataset of total loans and leases of commercial banks in billions of dollars. Metrics of the Data set includes 2066 fact values in 1 timeseries. Data set is collected from the Time Series Data Library which was created by Rob Hyndman, Professor of Statistics at Monash University, Australia[**?**].

4) Heart Rate Time Series: We take dataset of heart rates of a person in 15 minutes at 0.5 second interval. So there are 1800 evenly-spaced measurements (in units of beats per minute) of instantaneous heart rate timeseries. Data set is collected from MIT-BIH database[**?**].

III. Proposed Approach

Deep learning has shown various remarkable achievements in recent times in a wide variety of fields. Deep learning networks are able to perform complex calculation by using large amount of data and by doing a better function approximation, this helps Deep Networks to achieve better results in the fields of Natural Language Processing and Computer Vision.

Our approach here is to make a model for time-series forecasting using Recurrent Neural Networks (RNN). RNNs are good in persisting the information. They are connected in a loop, by which they can pass the information from one stage to another. We are trying to utilize that feature of RNN for the forecasting model.

RNN are good keeping memory for long term sequence, but they suffer with a series drawback, and that is the vanishing gradient. When we train a very large deep learning model

A. TensorFlow

TensorFlow[**?**] is a open sourced deep learning library which was released by Google at November 2015. It was a modification of their previous library called DistBelief. Its computational model is a directed graph and the nodes represented by function/computations and the edges usually represents number, matrices and tensors.

The *operation* provide abstract computation with the *attributes* for the graph. Where as *kernels* are used for the device selection for the either GPU or CPU based application. It makes a *Session* to create interaction with the client programs to various platforms.TensorFlow main component is *client* which create Session to communicate with *master*. It can be executed on either on Single machine or on to Multiple machines for large scale computation.

B. Recurrent Neural Network (RNN)

Recurrent Neural Networks (RNN) are very popular model in Natural Language Processing. They work on the idea of sequences which are related to each other. The RNN are called *recurrent* because the next stage is depend upon previous stage calculation. RNN can be treated as the network with the *memory*. They work better because of they can retain the structure and dynamic behaviors of the states.[**?**]

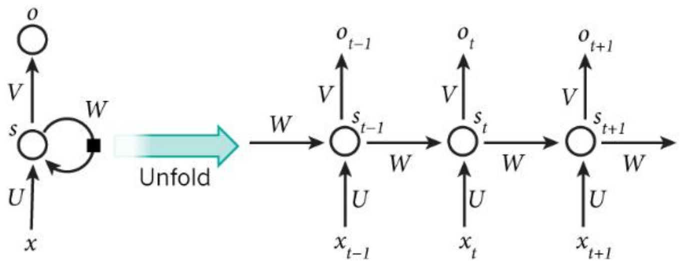

Fig. 1. Recurrent Neural Network

C. LSTM Networks

Long Short-Term Memory are special kind of RNN networks which are capable for long term based learning. Long term learning is their default behavior. LSTM networks have gating techniques which can be used to analyze long sequences. In our project we are using this technique for time series forecasting.[**?**]

D. Gated Recurrent Unit (GRU)

Gated Recurrent Unit are the slight modification of LSTM networks, it combines the input gate with forget

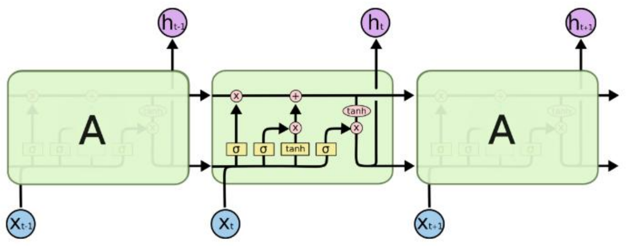

Fig. 2. Long Short Term Memory Networks

gate. It also combines cell state and hidden state which makes it a slightly better and simpler model of LSTM.[**?**]

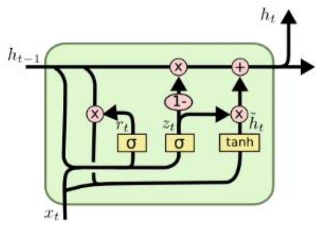

Fig. 3. Gated Recurrent Unit

IV. Experiments

We have performed our experiments using Python programming language which possess good capabilities to perform large computations efficiently and provides the library support for carrying out the experiments. We are using Keras Deep Learning library for the implementation, which is a wrapper around TensorFlow deep learning library. We have done our experimentation by implementing three RNN networks; First is Simple RNN. Second is a Long Short-Term Memory (LSTM) and last one is the Gated Rectified Unit (GRU).

We first start with preprocessing of data. The data has been taken into pandas dataframe in order to process it effectively with python. After that the data has been normalized with the value of using Gaussian distribution we have kept the mean zero and standard deviation 1. The data is then processed as an input to the RNN networks in a sequential manner to get a sequential output from the network.

Experiments have been performed on the i7 processor with 16 GB of memory along With Nvidia Titan-X (Pascal) 12 GB GPU. GPU has an advantage of processing data faster and perform highly efficient parallel computing. The training and testing set has been divided into 70 percent and 30 percent respectively. We are taking one step back numerical values. The dataset has been changed according to the model requirement. The model has fed to the single layer RNN with Simple RNN, LSTM and GRU models. We have used *Adam* optimizer for this and then we apply mean squared error to minimize the loss. The batch size is taken as one and the model has been trained for 10 epochs.

The fist set of experiment has been performed to determine the effect of different model on one set of dataset, firstly it is applied for the hourly power consumption of household's electric power with 8759 datapoints. The second experiment has been performed with 364 datapoints. Similar experiments have been performed with LSTM and GRU techniques as well.

The next set of experimentation is performed to determine the effect of hyperparmeters on the RNN, we have taken sliding window approach so we can take the last three values of the dataset. The number of epochs also increase with 10 to 100 with Single layer. Then the model is changed to multiple layer network where we increase number of layers from 1 to 3 neural network with $6X6X4$ layer architecture with 20 epochs. Finally we have taken 100 epochs with 3 layer multilayer network to see how output varies.

The last set of experiments involved using varied time-series datasets to understand the behavior of the neural networks, we have used Weather Data, Foreign Exchange Data, Commercial Bank Loan data and Heart Rate Data on the best model. For every experiment we collected train and test Root Mean Square Error(RMSE) to monitor the performance of the networks.

V. Results

We performed various experiments with various types of the dataset, models and hyper parameters so as to have a better insight about our model's applicability on other time-series datasets.

A. Model based implementation

In this model based approach we performed experimentation on hourly power consumption data with 8759 point with Simple RNN, LSTM and GRU based model. The result for this experiment is given below:

Same set of experiment is applied on LSTM based model.

Same set of experiment is applied on GRU based model.

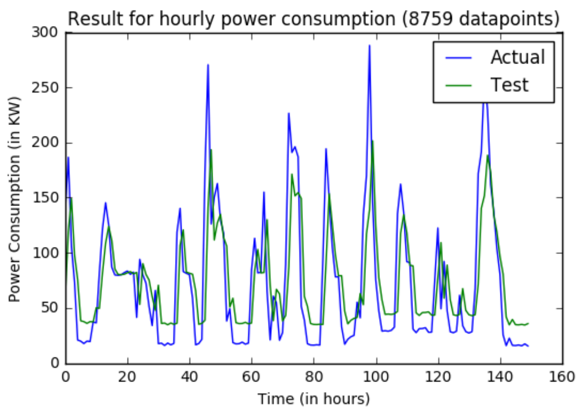

Fig. 4. Single Layer RNN

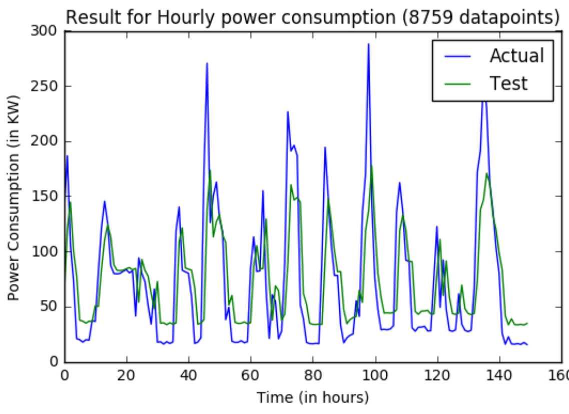

Fig. 5. Single Layer LSTM

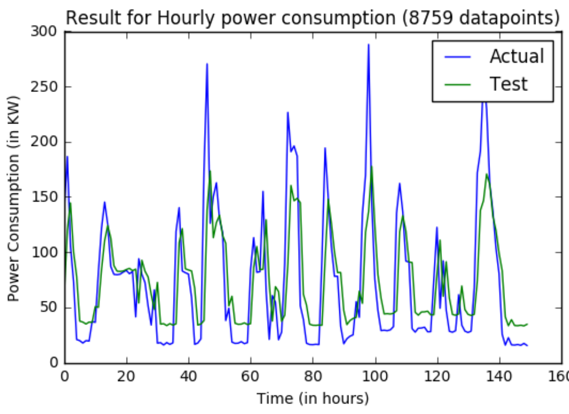

Fig. 6. Single Layer GRU

There results using the above models are stated below:

Number	Simple RNN	LSTM	GRU
Train(RMSE)	39.95	39.86	39.81
Test(RMSE)	45.85	45.62	45.46

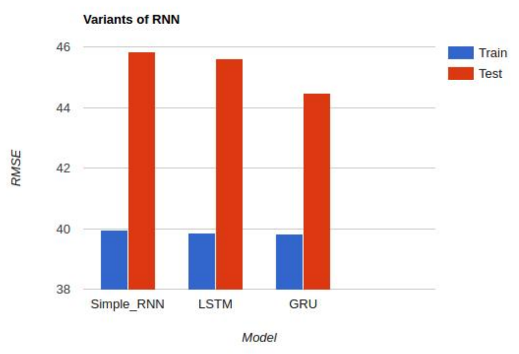

Fig. 7. Variants of RNN

So from the above table, it can be seen that the performance of GRU is better than LSTM and Simple RNN. However, LSTM still performed better than the Simple RNN as it resolves the vanishing gradient problem while learning and remembering only the important factors.

B. Hyper Parameter based Implementation

In this approach, we tuned hyper parameters with the GRU network to see if we can achieve any better performance from the network. We first started by taking sliding window approach and we set the window size as 3. So it can take upto last three values to make the prediction.

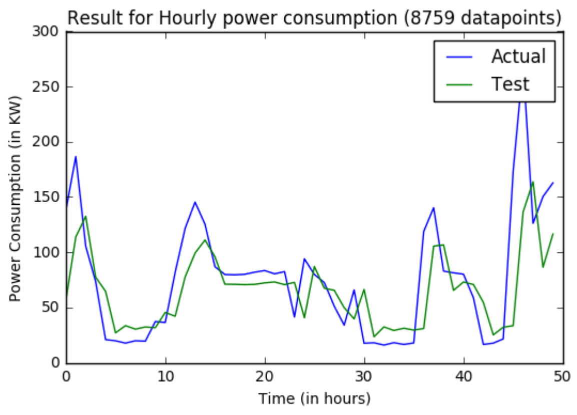

Fig. 8. 3 Step back value prediction, Sliding Window approach

Next we tried to increase the number of epochs for the network.

After that we tried to increase the number of layers for 20 epochs.

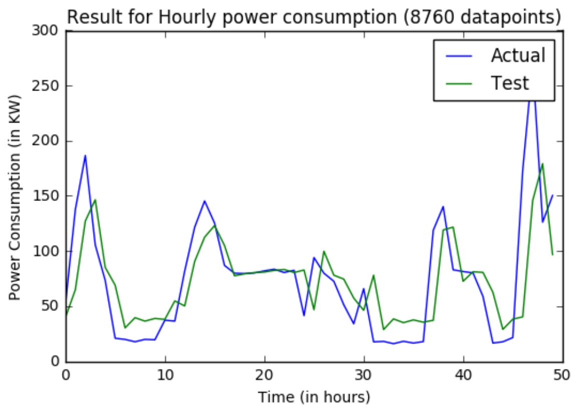

Fig. 9. 100 Number of Epochs (Single Layer)

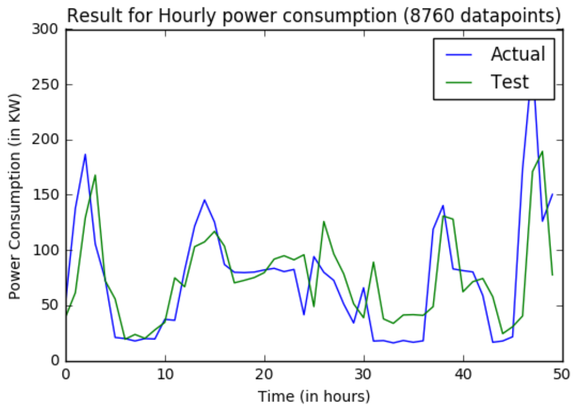

Fig. 11. Multi-Layer Network with 100 epochs

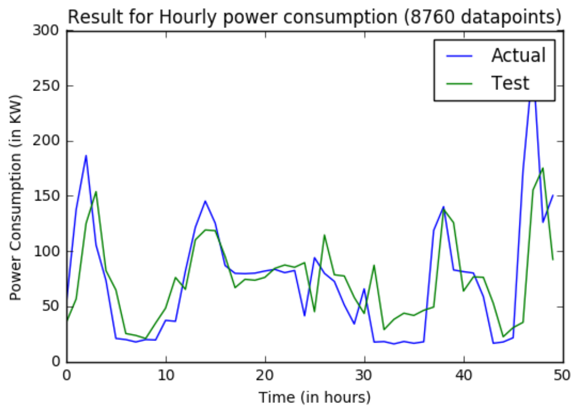

Fig. 10. Multi-Layer Network with 20 epochs

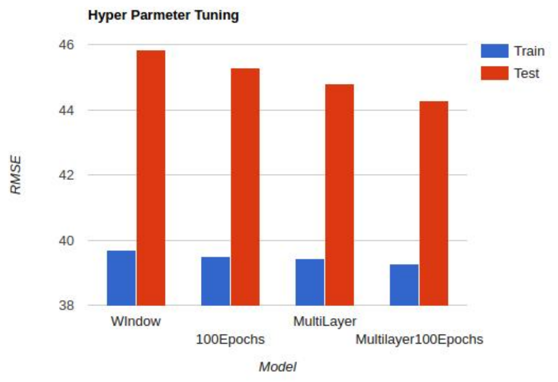

Fig. 12. HyperParameter Tuning

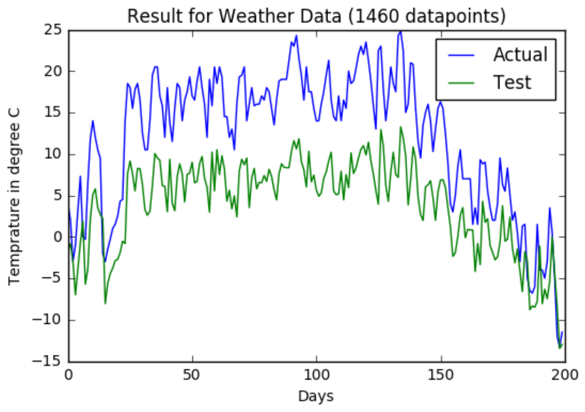

Fig. 13. Weather Data

Finally, we tried to increase the epochs in Multi-Layer Deep Network to 100 epochs.

Here, the error with Hyper Parameters obtained are given below.

It can be clearly seen by the graph that the performance of RNN increases with good parameter tuning.

C. Implementation with Different Types if Data Sets

After we tuned our model with the experimentation above, we tried to implement it on various time-series datasets to see the performance of this network over different types of data. First we applied it on the Weather Data Set, whose results graph is below.

The next data set we analyzed was the Foreign Exchange Rate dataset, whose results graph is given below.

The next data set we analyzed was the Commercial Bank Loan Dataset dataset, whose graph is given below.

Finally, the last data set we analyzed was the heart

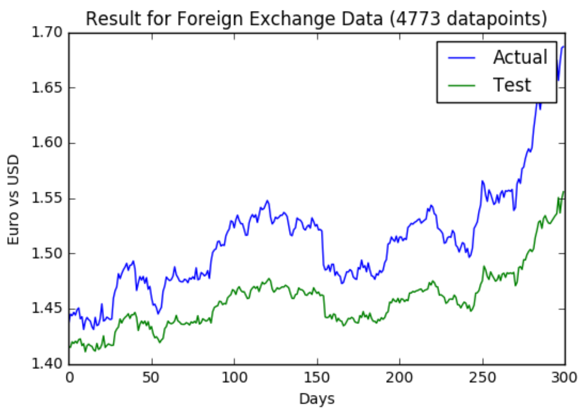

Fig. 14. Foreign Exchange Data

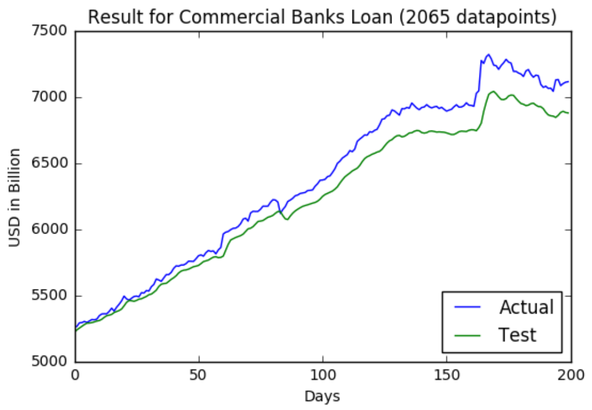

Fig. 15. Commercial Banks Loan

rate of a patient whose graph is given below.

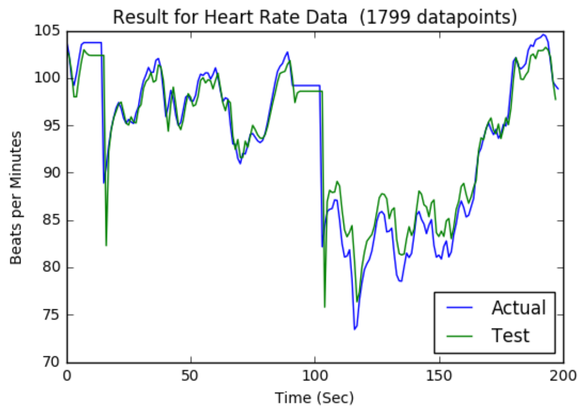

Fig. 16. Heart Rate Data

After completing this analysis we found that our network is adapting the trend of data with good patterns, however it completely depends upon the type of data we are using on our model. As it can be seen from the experimentation that our model adapted patterns in data, for example it has performed very well on Heart Beat Dataset because it follows a periodic pattern but on the other hand, it does not perform well on the weather data as it is very complex having very random patterns. It struggled a lot to predict the future values which shows the difficulty of task in predicting weather.

There is a scope of improvement for this model as we can introduce some more correlated features to get a better output from varied datasets.

VI. Conclusion and Future Scope

The attempt to use Deep Learning for Prediction and Forecasting provided several interesting and useful results. It can be seen that Deep learning can be used for Forecasting modeling, and it shows quite good performance with GPU rather than CPU. Recurrent Neural Network (RNN) having an advantage of memorizing dynamic states of the system and can be effectively used to improve time series forecasting. The variants of RNN, like LSTM and GRU showed significant performance over Simple RNN models. The Multilayer Deep Network works better than a single layer network. In time-series analysis, data is a important asset and large amount of data gives better model and good results. Hyper Parameters are also very important factor to achieve optimized result in forecasting. Sliding Window approach, better learning and dropout techniques can provide much better results. This model has been further tested upon various other datasets and it showed quite good performance over them.

As for the future scope, lots of improvement can be done on this approach. First this model can be used for benchmarking with the existing different statistical models and various other machine leaning techniques. It can be further improved by doing much more fine hyper parameter tuning. It can be incorporated with more features which can improve further results. We can also include some more correlation data, like in solar cell forecasting we can incorporate solar irradiation data and weather data which can help the networks to understand much better relationship within the data and achieve significantly good results. Deep learning has shown great success in Machine Learning tasks, and it has dominated the fields of Computer Vision and Natural Language Processing. It is capable to extract complex pattern from the data to provide meaningful information. We can also use it for forecasting models and with increased availability of the powerful computational resources such as GPU's, we can implement large computational models so as to achieve great success in that too.

References

[1] Large scale distributed deep networks (2012), J. Dean et al.
[2] Deep learning (Book, 2016), Goodfellow et al. (Bengio)
[3] Deep learning (2015), Y. LeCun, Y. Bengio and G. Hinton
[4] https://www.tensorflow.org/
[5] http://www.nvidia.com/object/cuda
[6] Christopher Olah Blog, http://colah.github.io/
[7] The Unreasonable Effectiveness of Recurrent Neural Networks http://karpathy.github.io/2015/05/21/rnn-effectiveness
[8] Introduction to Time Series Analysis http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc4.htm
[9] Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science
[10] https://datamarket.com/data/list/?q=provider:tsdl
[11] http://ecg.mit.edu/time-series/