Modeling approach for Time Series forecast

Shashwat Trivedi(15075063)
Harsh Rangwani(15075018)
Pradeep Jangid(15075018)

Department of Computer Science and Engineering
INDIAN INSTITUTE OF TECHNOLOGY (BHU) VARANASI
Varanasi 221005, India

Abstract

Time Series forecasting is a challenging task as it involves carefully learning of trends present in the data and predicting values based on the observed trends. In our project we use two different techniques of modeling the time series data, the ARIMA technique and the Seq2Seq RNN based technique. Our results show that the Seq2Seq RNN based techniques are efficiently able to model the time series sequential data which is comparable to the previosly existing ARIMA techniques.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Time series forecasting has been a problem that has fascinated many as it involves predicting what will happen in the future. It has immense applications in the field of Finance, Meteorology, Astronomy etc. Recently Machine Learning approaches are being applied for the problem of Time Series Forecasting as machine learning models are capable of learning the patterns and generating generalized prediction. The predictions can depend on various factors i.e. trends, seasonality etc. . The existence of large number of factors makes machine learning a suitable way to approach this problem. There have been many machine learning techniques that have been used for time series modeling which include SVM's [1], multitask neural networks [2] etc. In the project we will be using two techniques for time series forecasting i.e. ARIMA (Autoregressive Integrated Moving Average) [3] which uses traditional machine learning techniques and Seq2Seq RNN approach which is a deep learning based approach. We perform our experiments on the data from US airlines which is available for the public usage.

Chapter 2

Methodologies

2.1 ARIMA (Auto Regressive Integrated Moving Average)

The ARIMA method is used for regressing the values based on regressing weight parameters on time series lagged values. It differs from the earlier techniques by making use of the difference step which is helpful in removal of seasonality. There are three important parameters on which the model depends: **p** is the order of time lags, d is the degree of differencing and **q** is the order of the moving average [1]. These parameters are obtained through a random search in the parameter space.

2.2 RNN seq2seq

The RNN seq2seq models have been shown to work efficiently for natural language processing tasks such as Machine Translation[4][5], etc. The underlying sequence learning capabilities of RNN can be exploited for time series prediction also. The Seq2seq model take input sequence (univariate or multivariate) and are trained to produce a target sequence. A Seq2seq model thus consists of Encoders and Decoder. Encoder composes of a RNN cell that generates hidden states and cell state. The RNN cell encodes the information

[1] https://machinelearningmastery.com/tune-arima-parameters-python/

(trend and seasonality) of the sequence upto the current sequence input. The final encoder state is passed on to the decoder which also consists of RNN cell similar to encoder. Decoder along with the encoder hidden state takes actual sequence as input. It unrolls the RNN cell to output hidden state of each cell, which is further passed on the dense layer to predict the future sequence.

2.2.1 RNN Cell - LSTM

The LSTM(Long short term memory) [6] is used to memorize the values over long intervals. It is designed to overcome the problem of long-term dependency problem.

LSTM architecture have four gates. The first step of LSTM is to decide what information to keep and what information needs to be forgotten. The ”forget gate layer” which is a sigmoid layer to get a score on how much information is to be thrown away. The second step calculates what information should be update , this is done by ”update gate layer”. And the third step creates a new value that could be added to current state. After that the previous state is converted into new state. The final step is to decide what information is output.

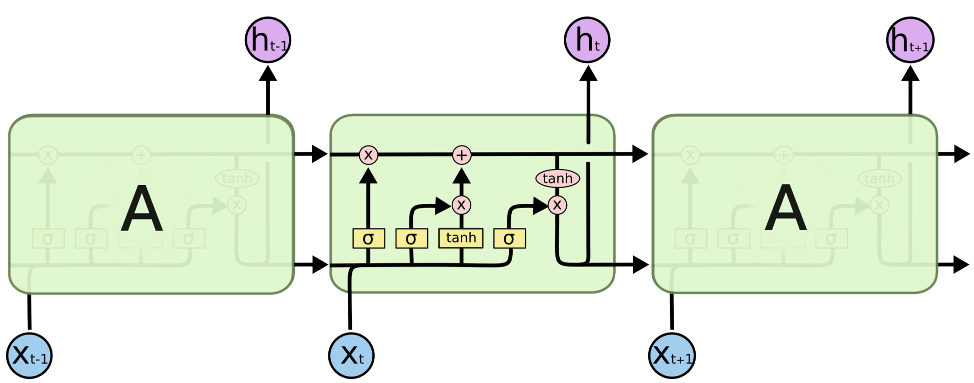

Figure 2.1: Unrolled LSTM cell

2.3 Transformer

A recurrent neural network finds its application in tasks based on sequential input, but RNN has the disadvantage of capturing long dependencies. However, LSTM and many NLP model based on bidirectional LSTM overcome this disadvantage and achieved human-level accuracy. A technique that pays attention to an only important pattern for judgement and helped the model to focus on significant aspects of sequence and mimic the human tendency[7] is Attention[8],

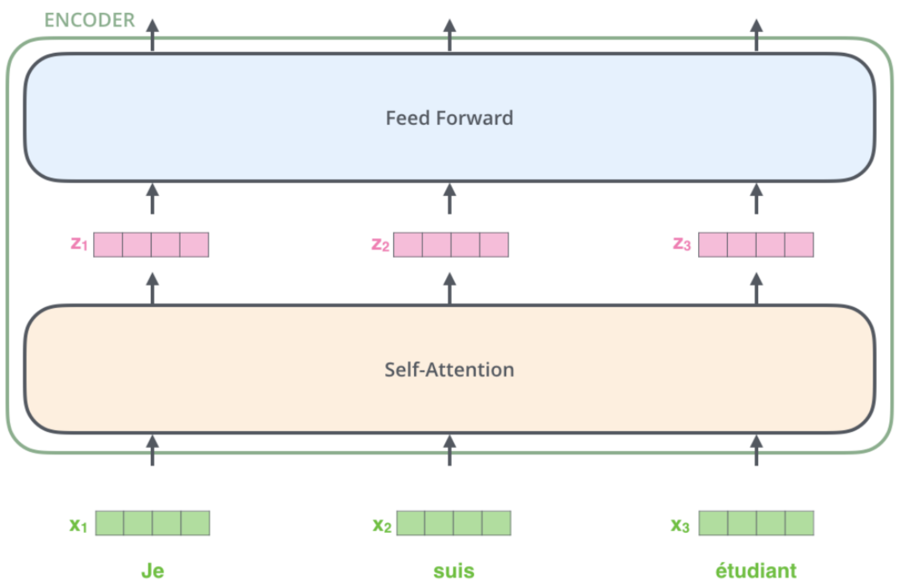

Figure 2.2: Transformer Cell

The transformer contains two major layers one is a self-attention layer, and the other is a feed-forward dense layer. vaswani2017attention showed that better accuracy are achievable in complex tasks like machine translation using only attention. .

2.3.1 Self-Attention

Self Attention layer takes sequential input representation from all tokens and generates a new embedding for each input based on all of them. The self-attention is calculated using three vectors created from each of the encoders input vectors. So for every token, it creates a

Query vector, a Key vector, and a Value vector. Separate trainable weight matrices generate these three vectors.

Dot product of Query vector and Key vector gives the attention score for each token. Next, these attention scores are passed through a softmax layer so that resultant is normalised and adds up to one. This gives the attention weight for each token, i.e. how much each token must attend to for each of the other tokens.

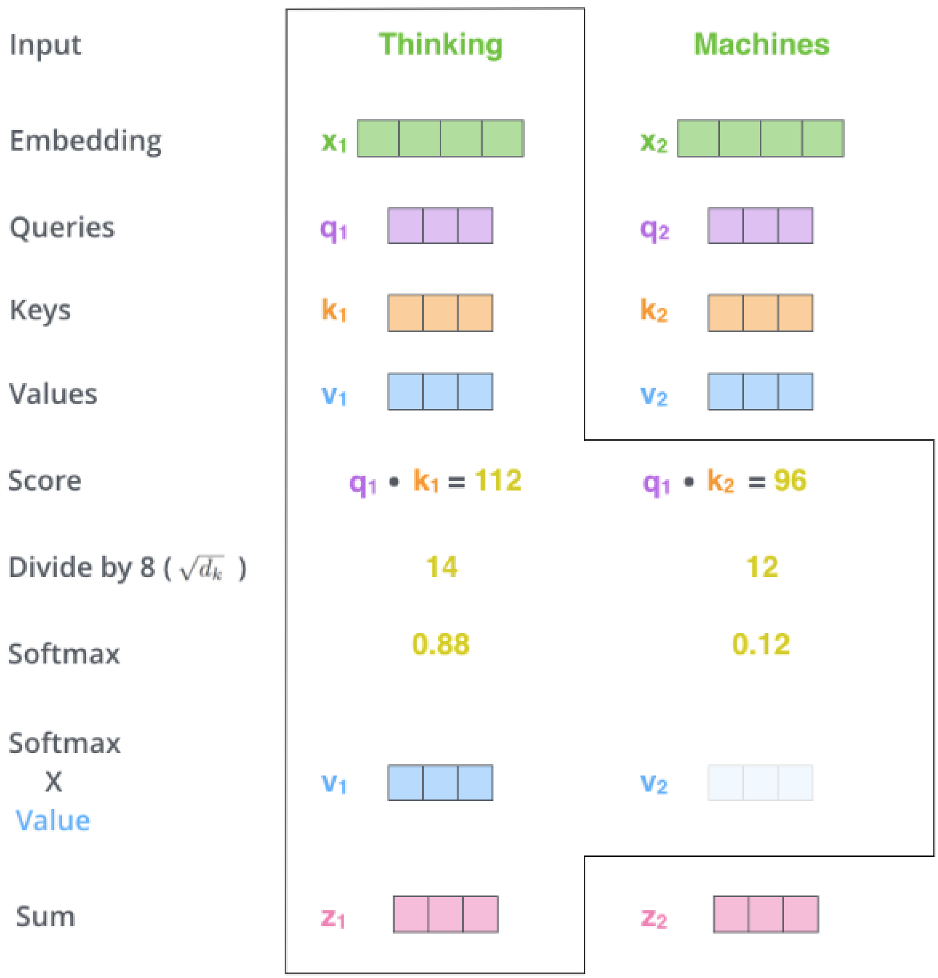

Figure 2.3: Attention Mechanism

Final attention vector is calculated by weighted average of value vectors based on attention weights calculated previously.

Chapter 3

Experiment

3.1 Dataset

In this project we use the US airlines dataset for prediction of important variables in the aviation industry. The dataset is composed of data for the time period October 2002 to March 2017 out of which the data of April 2016 to March 2017 is choosen for testing. The four variables are: **Passengers**, **Flight**, **ASM** (Seats * Mile Flown) and **RPM** (Paying Pessenger * Miles flown). In the task we take 162 values as input and forecast the next 12 values. For each variable we train a new model for prediction as each of them may have different trends and values associated with it.

3.2 Evaluation

For evaluating the performance of the model a metric should be choosen which takes care of the scale of prediction as well as the difference from the expected value. For comparing the two approaches we used a MAPE (Mean Absolute Percentage Error) as the estimate of performance for the two systems.

$$M = \sum_{t=1}^{n} \mid \frac{A_t - F_t}{A_t} \mid \tag{3.1}$$

rWhere A_t is the actual value and F_t is the forcasted value from the model. In case of ARIMA we minimize the Akaike Information Criteria (AIC) through maximum likelihood estimation whereas in the Seq2Seq model we minimize the MSE (Mean Squared Error) of the predictions.

Chapter 4

Results

For each of the variables we created three different models and compared the performance of the models across them. For the ARIMA model the hyperparameters are set to p = 12, d = 1 and q = 2. The hyperparameters of the deep learning based Seq2Seq RNN and Transformer are present in the code. One of the limitations of RNN is that it requires a lot of similar data to generalize well as we had the data from different airports which were similar. The MAP's of the three different approaches are summarized below in the following tables as following:

Dimension	American-Dallas	Delta-Atlanta	United-Chicago	Avg.
Passengers	3.076	1.816	7.922	4.271
Flights	2.839	3.302	11.143	5.761
ASM	2.133	1.113	6.330	3.192
RPM	6.274	1.705	5.273	4.417

Table 4.1: Results through ARIMA Approach

Dimension	American-Dallas	Delta-Atlanta	United-Chicago	Avg.
Passengers	3.743	3.513	4.122	3.793
Flights	5.909	2.887	3.603	4.133
ASM	3.418	1.739	3.013	2.723
RPM	4.678	2.980	2.523	3.394

Table 4.2: Results through Seq2Seq RNN Approach

Dimension	American-Dallas	Delta-Atlanta	United-Chicago	Avg.
Passengers	1.14	4.65	1.7	**2.50**
Flights	0.8	3.8	5.2	**3.26**
ASM	0.63	1.2	1.9	**1.24**
RPM	1.7	2.2	0.8	**1.57**

Table 4.3: Results through Seq2Seq Transformer Approach

We compare the results of three models, ARIMA, Seq2Seq using RNN and Seq2Seq RNN with transformer Encoder. The evaluation was done on data collected from U.S. Department of Transportation which is available publicly [9].We tested the models on American-Dallas with airport Fort-Worth, Delta Atlanta, and United-Chicago, for comparison with the results from the previous models. With each dataset containing time-series from passengers, flights, available seat-mile and revenue.

The time-series graph results are shown in the figure 4.1 4.2 4.3 along with the MAPE values for comparison. The Seq2Seq transformer approach outperforms the other approaches by around 30% to 50% on the test time series. Although having 12 outputs is not a common scenario in time series forecasting but having it to predict 12 is difficult as the system has to learn trends and patterns more specifically for generating them.

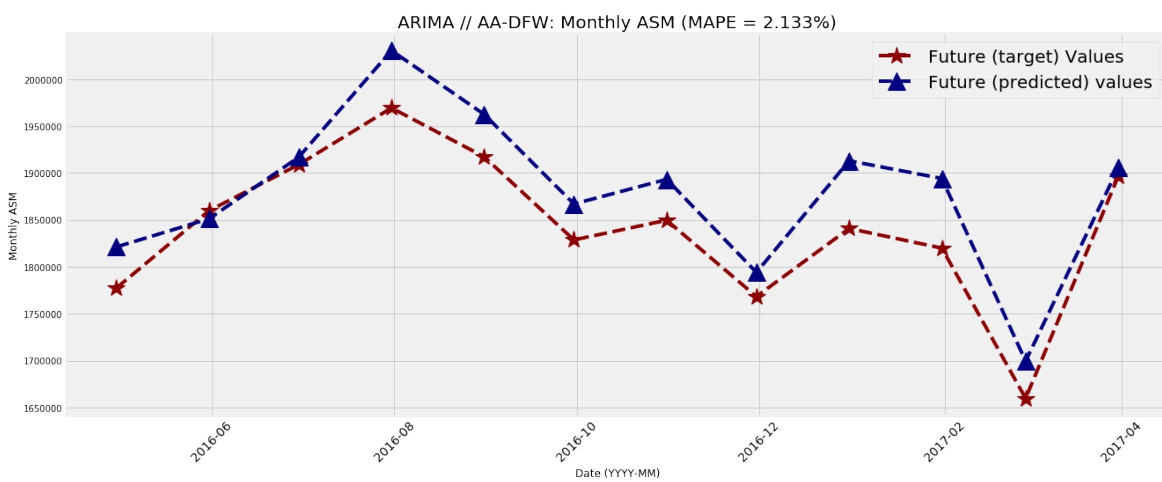

Figure 4.1: ARIMA : Ground truth vs Forecast

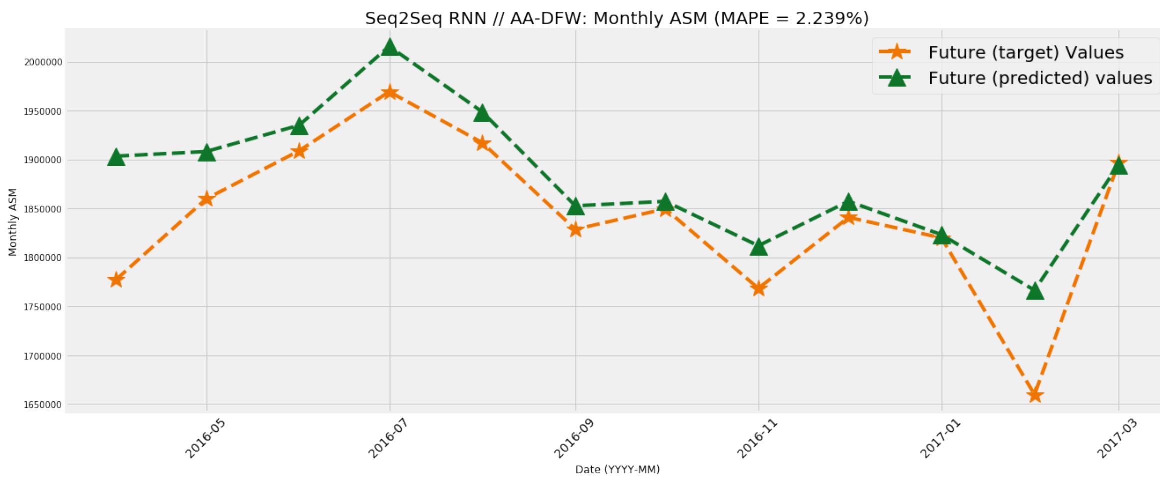

Figure 4.2: Seq2Seq RNN : Ground truth vs Forecast

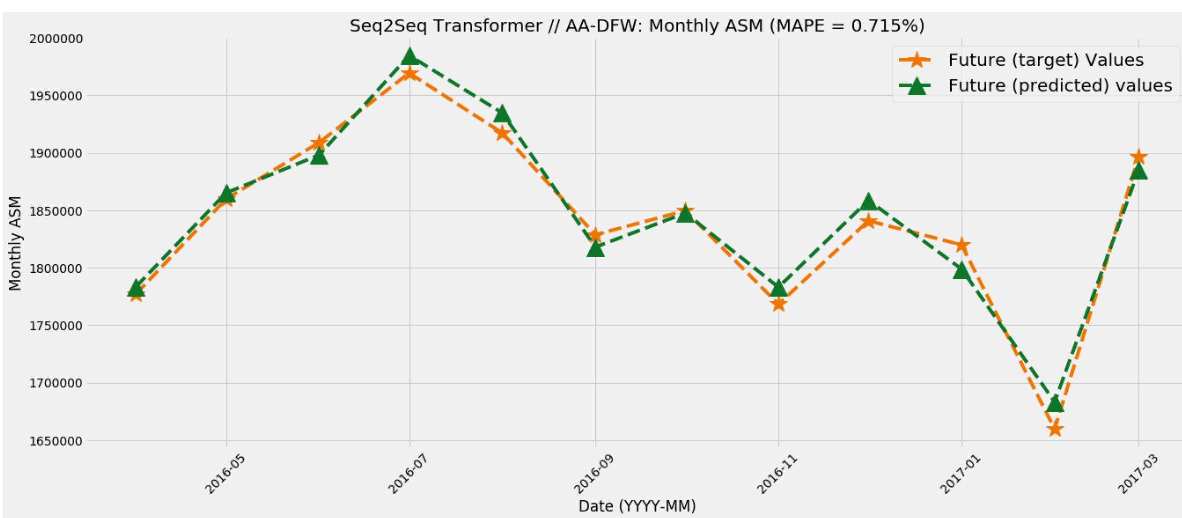

Figure 4.3: Seq2Seq Transformer : Ground truth vs Forecast

Bibliography

[1] Y. Bao, T. Xiong, and Z. Hu, "Multi-step-ahead time series prediction using multiple-output support vector regression," *Neurocomputing*, vol. 129, pp. 482–493, 2014.

[2] R.-G. Cirstea, D.-V. Micu, G.-M. Muresan, C. Guo, and B. Yang, "Correlated time series forecasting using multi-task deep neural networks," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 1527–1530.

[3] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the arima model," in *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*. IEEE, 2014, pp. 106–112.

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[5] S. Liu, N. Yang, M. Li, and M. Zhou, "A recursive recurrent neural network for statistical machine translation," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2014, pp. 1491–1500.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *CoRR*, vol. abs/1409.0473, 2014. [Online]. Available: http://arxiv.org/abs/1409.0473

[8] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” in *Advances in neural information processing systems*, 2015, pp. 577–585.

[9] [Online]. Available: https://www.transtats.bts.gov/Data$_{E}lements.aspx$