**UNIVERSITY OF MASSACHUSETTS – DARTMOUTH**

**DSC550 – FINAL PROJECT REPORT**

**Study of LSTM applicability: Weather and Stock prediction**

**ARIJIT DEY (01944575)**

**Author**

**Dr. Iren Todorova Valova**

**Instructor for DSC 550**

### Department of Computer & Information Sciences

**University of Massachusetts – Dartmouth**

**12/07/2021**

Abstract

Long Short-Term Memory (LSTM) is a specific recurrent neural network (RNN) architecture that was designed to model temporal sequences and their long-range dependencies more accurately than conventional RNNs. In this paper, we explore LSTM RNN architectures for weather (temperature) and stock price prediction. This architecture makes more effective use of model parameters than the others considered, converges quickly, and outperforms a deep feed forward neural network having an order of magnitude more parameters.

Climate change has affected the weather forecast on a regular basis compared to reality. Meanwhile, weather forecast plays an important role in daily life and especially it affects developed countries in agricultural fields around the world. When we apply information technology software, we can assess the general weather condition of a given city, and with the help of recent modern scientific methods for more accurate analysis and prediction of weather based on those collected weather data for a period of a week earlier or longer for future weather forecasts. Therefore, in this study, machine learning model was applied, and it was allowed to study methods and feature engineering from data pre-processing, so through this, we find an unpredictable model of temperature conditions, a prognostic model is also monitored. For this work, various weather parameters were collected from the national climate data center through available software applications, with the application of the Long Short-Term Memory (LSTM) model is an artificial recurrent neural network (RNN) were trained for different combinations.

Stock market prediction has always been an interesting research topic among researchers mainly due to its capital gain by trading stocks and or to understand the information hidden in stock market data. Many machine learning algorithms and statistical models have been proposed by researchers for stock price prediction and stock price movement prediction. We have studied various machine learning methods and techniques for stock market prediction. Here we present recent growth in stock market prediction methods and models, perform a comparison among these models to find out the accuracy of the prediction of the stock market values and figuring out the advantages and disadvantages of these individual models. We are using LSTM models to predict future stock prices.

Introduction

Recurrent networks can in principle use their feedback connections to store representations of recent input events in form of activations (“short-term memory", as opposed to “long-term memory" embodied by slowly changing weights). This is potentially significant for many applications, including speech processing, non-Markovian control, and music composition. The most widely used algorithms for learning what to put in short-term memory, however, take too much time or do not work well at all, especially when minimal time lags between inputs and corresponding teacher signals are long. Although theoretically fascinating, existing methods do not provide clear practical advantages over, say, backprop in feedforward nets with limited time windows. This project presents “Long Short-Term Memory" (LSTM), a novel recurrent network architecture in conjunction with an appropriate gradient-based learning algorithm. LSTM is designed to overcome this error back- ow problems. It can learn to bridge time intervals more than 1000 steps even in case of noisy, incompressible input sequences, without loss of short time lag capabilities.

Recurrent Neural Networks

In the below diagram, a chunk of neural network, A, looks at some input  and outputs a value . A loop allows information to be passed from one step of the network to the next.

These loops make recurrent neural networks seem kind of mysterious. However, if you think a bit more, it turns out that they aren’t all that different than a normal neural network. A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. Consider what happens if we unroll the loop:

Diagram

Description automatically generated

Figure:1

These loops make recurrent neural networks seem kind of mysterious. However, if you think a bit more, it turns out that they aren’t all that different than a normal neural network. A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. Consider what happens if we unroll the loop:

A picture containing text, clock

Description automatically generated

Figure:2

This chain-like nature reveals that recurrent neural networks are intimately related to sequences and lists. They’re the natural architecture of neural network to use for such data.

LSTM Network

Long Short-Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. They were introduced by [Hoch Reiter & Schmid Huber (1997)](http://www.bioinf.jku.at/publications/older/2604.pdf). They work tremendously well on a large variety of problems and are now widely used.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer as shown in figure 3.

Diagram

Description automatically generated

Figure:3

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.

Figure:Diagram

Description automatically generated4

Diagram

Description automatically generated with medium confidence

Figure:5

In figure 5, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denotes its content being copied and the copies going to different locations

Diagram, schematic

Description automatically generated

Figure:6

Literature Overview/Bibliography

**Related work on stock data prediction**

During the pre-deep learning era, Financial Time Series modelling has mainly concentrated in the field of ARIMA and any modifications on this, and the result has proved that the traditional time series model does provide decent predictive power to a limit. For example, due to the asymmetric distribution in financial time series return, Minyoung Kim has replaced the traditional Maximum Likelihood Estimation with an asymmetric loss function. Compared the forecasting performance of ARIMA and artificial neural networks on Korean stock price index. The work showed that ARIMA provided more accurate forecasts than the back-propagation neural network. More recently, deep learning methods have demonstrated better performances thanks to improved computational power and the ability of learning non-linear relationships enclosed in various financial features. Compared three different deep learning architectures including RNN, LSTM, and CNN-sliding window models for the prediction of NSEI listed stocks. They concluded that CNN architecture can identify changes in trend of stocks and outperforms other models. Yan and Ouyang combined the wavelet transform of the financial time series with the LSTM and showed that the resulting model beat the performance of traditional Support Vector Machine, and K-nearest Neighbors. Thien Hai Nguyen et al. demonstrated that the integration of sentiment features extracted from social media can improve the accuracy of prediction. The performance of LSTM-RNN will be further boosted by feeding relevant data based on financial domain knowledge. Moreover, Kim Won has developed a hybrid approach to combine LSTM and GARCH models and the resulting model has much lower prediction errors.

**Review using Univariate LSTM and Multivariable LSTM Models on weather data**

On weather prediction earlier below ten alternative univariate LSTM models and ten multivariate LSTM models were listed in Table 1. results show that the model with 64 neurons and the SGD of univariate LSTM had the lowest RMSE for test set (RMSE = 11.20) in comparison with the models using other parameters. The model with 32 neurons and the Adam of multivariate LSTM had the lowest RMSE for the test set (RMSE = 10.78) in comparison with models using other parameters.

Table

Description automatically generated with medium confidence

Table 1

Table 1 shows the univariate and multivariate models with different neuron, optimizer, epochs, and batch size with different Root Mean Square Error (RMSE). This experiment with above combination of parameters was conducted and recorded the RMSE by NCBI in this article (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8201362/).

Project Methodology

Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). The Long Short-Term Memory or LSTM network is a recurrent neural network that is trained using Backpropagation through Time and overcomes the vanishing gradient problem. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three gates can be thought of as a "conventional" artificial neuron, as in a multi-layer neural network: that is, they compute an activation of a weighted sum. Intuitively, they can be thought as regulators of the flow of values that goes through the connections of the LSTM; hence the denotation "gate". There are connections between these gates and the cell.

Diagram

Description automatically generated

Figure:7

Using stacked LSTM

 A Stacked LSTM architecture can be defined as an LSTM model comprised of multiple LSTM layers. An LSTM layer above provides a sequence output rather than a single value output to the LSTM layer below. Specifically, one output per input time step, rather than one output time step for all input time steps as shown in figure 8.

Diagram

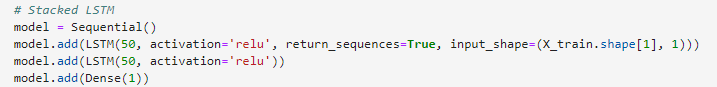
Description automatically generated

Figure:8

Implement Stacked LSTMs using Keras

Each LSTMs memory cell requires a 3D input. When an LSTM processes one input sequence of time steps, each memory cell will output a single value for the whole sequence as a 2D array.

Below with a model that has a double hidden LSTM layer that is also the output layer.



Experiment

*Stock Prediction LSTM Model/Flow:*

In the process of predicting the future of stock market there needs to be a detailed procedure to follow as described in this figure

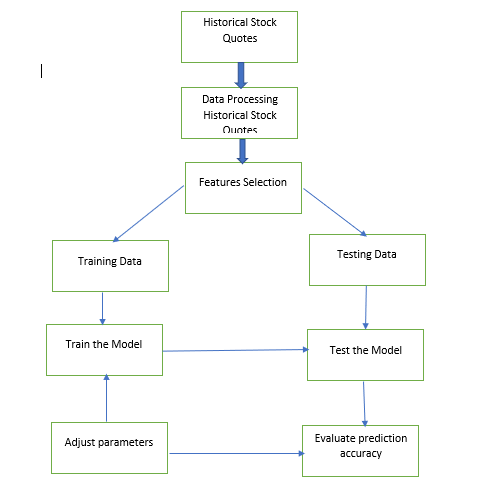


Figure:9

* The initial step will be to collect some historical stock data of any firm from yahoo finance that is used for predicting the values of stock prices.
* Then the data collected is preprocessed by using Data scaling and Data discretization techniques.
* In this step, only the features such as Date, open, high, low, close, and volume which are to be fed to the neural network are chosen.

Table

Description automatically generated

Table 2

The table 2 is displaying few top records of the stocks with the parameters of the stocks

* The historical stock data is divided into training data and testing data, by year like 2016 and 2017 respectively.

Chart, scatter chart

Description automatically generated

Figure:10

* The data are fed to the recurrent neural network and trained for prediction. Our LSTM model consist of a sequential input layer followed by LSTM layers, dense layer and then finally a dense output layer with linear activation function.
* The output value generated by the output layer of the RNN is compared with the target value. The error between the target and the obtained output value is minimized by using back propagation through time algorithm which adjusts the weights and the biases of the network.

*Weather Forecasting Model:*

* We are going to use the daily ambient temperature of Yilan County in Taiwan provided by the Environmental Protection Administration of Taiwan and applying median filter and Gaussian filter on the dataset

This dataset has two columns, first column is date and temperature reading is the second column.

Graphical user interface, text

Description automatically generated

* Setting our training and testing data set based on our set variables and then normalizing the data.

Graphical user interface, text, application

Description automatically generated

Now that we have our training set, we will define our predictors utilizing the LSTM architecture. We have the following:

**Bidirectional**: to ensure that information from earlier part of the sequence is available for the latter, and vice versa.

**LSTM** **(x3) layers**: the 50 units represent the 30-day windowed historical datasets while the *return sequences*parameterspecifies that each unit’s output will be used as the input of the next layer.

**Dropout (p=0.2)**: for regularizing effects and preventing the network from training set’s over-fitting.

**Dense layer**: With 1 prediction neurons where each represent a day into the future.

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated

Results and Discussion

***A brief description on the parameters tuned below for building the model:***

Let’s discuss about the attributes I have changed to tune the model so that it predicts the future stocks more accurate and what difference it actually makes in the predictions.

***LSTM layers:*** Stacking LSTM hidden layers makes the model deeper, more accurately earning the description as a deep learning technique.

Additional hidden layers can be added to a Multilayer Perceptron neural network to make it deeper. The additional hidden layers are understood to recombine the learned representation from prior layers and create new representations at high levels of abstraction. For example, from lines to shapes to objects.

***Model:*** A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.

A Sequential model is not appropriate when:

* Your model has multiple inputs or multiple outputs
* Any of your layers has multiple inputs or multiple outputs
* You need to do layer sharing
* You want non-linear topology (e.g. a residual connection, a multi-branch model)

***Activation:*** The purpose of the Rectified Linear Activation Function (or ReLU for short) is to allow the neural network to learn nonlinear dependencies. Specifically, the way this works is that ReLU will return input directly if the value is greater than 0

***Input parameter Layer:*** The LSTM input layer is specified by the “*input\_shape*” argument on the first hidden layer of the network.

The input to every LSTM layer must be three-dimensional.

The three dimensions of this input are:

* Samples: One sequence is one sample. A batch is comprised of one or more samples.
* Time Steps :One time step is one point of observation in the sample.
* Features: One feature is one observation at a time step.

This means that the input layer expects a 3D array of data when fitting the model and when making predictions, even if specific dimensions of the array contain a single value, e.g. one sample or one feature.

***Output parameter Layer:*** Dense layer is the regular deeply connected neural network layer. It is most common and frequently used layer. Dense layer does the below operation on the input and return the output.

output = activation(dot(input, kernel) + bias)

where,

input: represent the input data

kernel: represent the weight data

dot: represent numpy dot product of all input and its corresponding weights

bias: represent a biased value used in machine learning to optimize the model

activation: represent the activation function.

***Epochs:*** An epoch is a term used in machine learning and indicates the number of passes of the entire training dataset the machine learning algorithm has completed

*Weather Forecasting:*

In this section we will discuss different outcomes/predictions performed the experiments with different values of parameters and models using LSTM.

**Model 1:** In this experiment below are the parameter values chosen and performed the experiment.

***LSTM layer: 1***

***Optimizer: Adam***

***Learning rate: 0.001***

***Input parameter:50***

***Output parameter Layer: 1***

***Epochs: 10***

***Step: Single step***

Table

Description automatically generated

On performing the experiment, we observed that training and the validation losses after 6 epochs are constant as shown in figure 11.

Chart, line chart

Description automatically generated

Figure:11

On plotting the points after every 6 steps for true future and model prediction with Time step in x-axis, we observed that low dips of temperature are predicted accurately. When the dip of the temperature is large the prediction is not accurate, with that said it is a difference of 0.75 degree Celsius. This happens because of huge fluctuation of temperature in short duration of time(days), model in this case cannot predict actual rise of the temperature as shown in Figure 11(b).

Chart

Description automatically generatedChart, line chart

Description automatically generated

Figure:11 (a) & (b)

Chart

Description automatically generatedChart, line chart

Description automatically generated

Figure:11 (c) & (d)

Chart, line chart

Description automatically generated

Figure:11 (e)

**Model 2:** In this experiment below are the parameter values chosen and performed the experiment.

***LSTM layer: 1***

***Optimizer: RMS***

***Learning rate: 0.001***

***rho=0.8,***

***Momentum=0.0,***

***Epsilon=1e-07***

***Input parameter:50***

***Output parameter Layer: 1***

***Epochs: 10***

***Step: Multi step***

Table

Description automatically generated

On performing the experiment, we observed that training and the validation losses after 6 epochs are constant.

Figure:12

Chart, line chart

Description automatically generated

On plotting the points after very 10 steps for true future and model prediction with Time step in x-axis, we observed that low dips of temperature are predicted accurately while when the dip of the temperature is large in this model the prediction is more accurate compared to model 1, with that said it has a difference of 0.25. After tuning the parameters we build the model and feed the same data into int and we observed that the difference in actual and prediction is minimized by 66% , with that said the difference in is now 0.25 degree Celsius as shown in Figure 12(b)

Chart

Description automatically generatedChart, line chart

Description automatically generated

Figure:12 (a) & (b)

Chart

Description automatically generatedChart, line chart

Description automatically generated

Figure:12 (c) & (d)

Chart, line chart

Description automatically generatedChart

Description automatically generated

Figure:12 (e) & (f)

Chart, line chart

Description automatically generatedChart

Description automatically generated

Figure:12 (g) & (h)

**Model 3:** In this experiment below are the parameter values chosen and performed the experiment.

***LSTM layer: 2***

***Model: Sequential***

***Activation: Relu***

***Input parameter Layer 1: 50***

***Input parameter Layer 2: 50***

***Output parameter Layer: 1***

***Epochs: 50***

***Step: Continuous***

Table

Description automatically generated

In the above model,we have applied sequential LSTM layers to refine the input model and feed the output to next LSTM model to gain the accuray of the model.Also,trained this model with 50 epochs with slow learning rate, so that patterns of the training data is memorised and adapted by the model. On applying testing data we observed that true value and the predicted values are very accurate compared to above two models as shown in the Figure 13 where we tried to predict the entire dataset.

Again, we ran the model on first 75 days data to visulaize the comparison in large scale as shown in Figure 14, we observed there is slight mismacth between fisrt 10 days , but it predicts excatly same with matching pattern after 10-75 days.

Chart, line chart

Description automatically generated

Figure:13

Chart, line chart

Description automatically generated

Figure:14

***Summary of the dataset in subplots:***

Graphical user interface, chart, histogram

Description automatically generated

Figure:15

In above Figure15, we have plotted entire dataset , the prediction plots for n days and 75 days, Mean square error and scatter plot.

As per above Figure 15, MSE vs Epoch plot we observed that after 20 epochs the Loss is minimized the lowest and remains constant.so , during model builing 20 epochs is sufficient gto gtraing the data,

Also we saw the scatter plot, the actual vs predicted value, and we can see that all points of prediction is lies around the abosule line that is drwan, which means it has strong relation between actual and predicted values.

*Stock Prediction:*

**Model parameters:**

***LSTM layer: 4***

***Model: Sequential***

***Optimizer: RMSProp***

***Input parameter Layer 1: 50***

***Input parameter Layer 2: 50***

***Input parameter Layer 3: 50***

***Input parameter Layer 4: 50***

***Output parameter Layer: 1***

***Drop out = 0.2***

***Epochs: 50***

***Step: Continuous***

Table

Description automatically generated

In figure 16, we have plotted the loss value of the training of the dataset, the loss is high during the initial stages of the training, but as the training progresses the loss goes on decreasing, the lower the loss the better will be the prediction. But we saw a spike in the loss (MSE) around 50 epochs that was caused due to noisy data (not equally distributed) training data. But after 100 epochs we find the loss is constant for the model.

Chart, histogram

Description automatically generated

Figure:16

In Figure 17, The plot shows the stock price graph of IBM over a period with real stock data (In red) and predicted stock price by LSTM model (In blue), depicts actual and predicted stocks predictions are very close. It is observed that the actual as well as predicted values are both close and produce acceptable testing accuracy.

Chart, histogram

Description automatically generated

Figure:17

Reflection statement on Project Impact

Neural networks like Long Short-Term Memory (LSTM) recurrent neural networks can almost seamlessly model problems with multiple input variables.

This is a great benefit in time series forecasting, where classical linear methods can be difficult to adapt to multivariate or multiple input forecasting problems.

The behavior of time series data is highly stochastic, which increases the uncertainty in finding a pattern in the data.

Nonetheless, recent advancements in machine learning algorithm such as deep learning can improve the prediction performance with confidence.

This means that in addition to being used for predictive models (making predictions) they can learn the sequences of a problem and then generate entirely new plausible sequences for the problem domain.

Generative models like this are useful not only to study how well a model has learned a problem, but to learn more about the problem domain itself.

I am interested to explore multiple parameters to optimize the model to yield better performance in the availability of computing resources. Moreover, my idea is to expand the utility of a program to employ the similar time-series data from different domains.

Lessons learned from the project

In this project, we found that LSTM is a good tool for predicting data. However, we can see from here that there are several things to take home as lessons learnt using LSTM.

First, more input does not really mean that the model will be more accurate. Other than that, data conditioning may help in making the model more accurate.

Also, we have seen tuning the model parameters helps in building a model more predictive for a given set of data.

Lastly, even though we haven’t shown, LSTM needs a certain amount of data to be applied for predicting future data.

From this project, we can imagine that LSTM can be used for predicting statements, stocks, weather, trends, and a lot more.

Conclusions

Long Short-Term Memory are the best known for time-series predictions as they can process sequence data and, they can be integrated with other deep learning techniques such as convolutional neural networks (CNN) for processing sophisticated hybrid models, but also, they do take up a lot of memory for processing.

**I would like to suggest some of the ways to you can follow to increase the accuracy of the model:**

* Increase or decrease the number of epochs to define the number times that the learning algorithm will work through the entire training dataset.
* One of the best ways is to use a large dataset and train the model, but this might take a longer time. You can use GPUs with Kera’s by installing CUDA.
* Add more LSTM layers.

The use of LSTM as an improved in the case of weather forecasting (in general, time-series datasets). And being able to perform weather forecasting in a timely and efficient manner is not only beneficial but is critical to the masses who directly or indirectly rely on good weather.

References

|  |  |  |
| --- | --- | --- |
| Author | Publication Title | Publication Date |
| Minyoung Kim  (Chungnam National University) | Cost-Sensitive Estimation of ARMA Models for Financial Asset Return Data | October 2015 |
| Thien Hai Nguyen | PhraseRNN: Phrase Recursive Neural Network for Aspect-based Sentiment Analysis | 2015 |
| Hoch Reiter & Schmid Huber | Long short-term memory | 1997 |