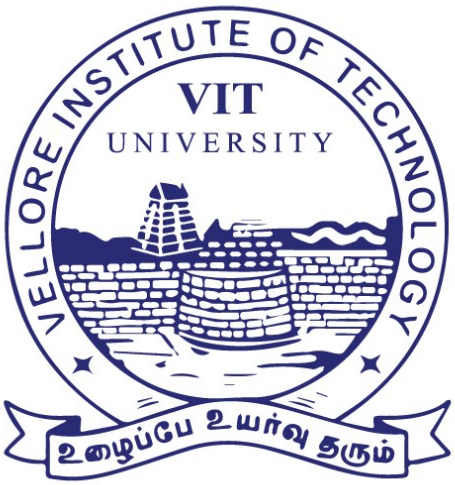


VIT®

UNIVERSITY

(Estd. u/s 3 of UGC Act 1956)

Title : Stock Market Prediction and Cluster Grouping Visualization Using Recurrent Neural Network(RNN) And T-distributed Stochastic Neighbor Embedding(t-SNE) on Tensorflow and Tensorboard

Course: Machine Learning
CSE 4020

Slot : D1

By
Sheril S. Philip (15BCB0120)
Debashis Karmakar (15BCB0049)

Video Link : https://drive.google.com/open?id=1nqZT8BswQ4LZyh-USn5fHRElXpsTwbBR

Introduction

1.1 General Instruction:

Out of all the books offering investing advice to research papers analyzing mathematical prediction models, the stock market has always been centre of attraction for public and academic interest. Number of publications propose strategies with good profits, while others demonstrate the random and unpredictable behaviour of share prices. This debate on how to predict stock market recently piqued our interest and led us to choose our Major Project topic within this area of research. The following observations influenced our decision:

- There is large amount of relevant financial data available on the internet which is increasing day by day.
- Large number of C. Sc disciplines including software engineering, databases, distributed systems and machine learning have increased possibility to apply skills.
- The opportunity to expand our knowledge in finance and investing, as we had only little prior exposure to these fields.

The following sections define the goal of the project and give an overview of the system that was built.

Basics:

In order to clarify the goal of the project, following are the dominant schools of thought on investing must first be introduced.

Fundamental analysis

This approach is to analyse fundamental attributes in order to identify promising companies.

This includes characteristics such as financial results, company's assets, liabilities, and stock and growth forecasts. It's very important to understand that this type of analysis is not static; newly released financial information, corporate announcements and other news can influence the fundamental outlook of a company. Fundamental analysis requires expertise in a particular sector and is often conducted by professional analysts. Their recommended investments are regularly published and updated.

Technical analysis

In contrast to fundamental analysis, technical analysis does not try to gain deep insight into a company's business. It assumes the available public information does not offer a competitive trading advantage. Instead, it focuses on studying a company's historical share price and on identifying patterns in the chart. The intention is to recognize trends in advance and to capitalize on them.

Goal

The goal was to build a system capable of the following tasks:

1. **Collecting fundamental and technical data from the internet**

The system should be able to crawl specific websites to extract fundamental data like news articles and analyst recommendations. Furthermore, it should be able to collect technical data in the form of historical share prices.

2. Simulating trading strategies

The system should offer ways to specify and simulate fundamental and technical trading strategies. Additionally, combining the two approaches should be possible.

3. **Evaluating and visualizing trading strategies**

The system should evaluate and visualize the financial performance of the simulated strategies. This allows a comparison to be made between technical, fundamental and the combined approaches.

Financial information sources on the Web.

1. www.google.finance.com
2. Moneycontrol.com- maintain excellent electronic versions of their daily issues.
3. Quantopian.com
4. www.nseindia.com

This rich variety of on-line information and news make it an attractive resource from which to mine knowledge. Data mining and analysis of such financial information can aid stock market predictions.

1.2 Relevant current/open problems.

- Data-are-humongous, nowadays we are seeing a rapid-explosion of numerical-stockquotes and textual-data. They are provided from all different-sources.
- Demand forecasts are important since the basic op management process, going from the vendor raw-materials to finished goods in the customers' hands, takes some time. Most firms cannot-wait for demand to elevate and then give a reaction. Instead, they make-up their mind and plan according to future demand so-that they can react spontaneously to customer’s order as they arrive.
- Generally, demand forecasts-lead to good-ops-and great-levels of customer satisfaction, while bad forecast will definitely-lead to costly ops and worst-levels of customer satisfaction.
- A confusion for the forecast is the horizon, which is, how distant in the future will the forecast project? As a simple rule, the away into the future we see, the more blurry our vision will become -- distant forecasts will be inaccurate that short-range forecasts.

1.3 Problem statement

As we discussed problems above we are going to implement the following:

- In this project, we-are trying to review the possibility to apply two-known techniques which is Recurrent neural network and t-SNE in stock market prediction. Extract useful information from a huge amount of data set and data mining is also able to predict future trends and behaviors through them. Therefore, combining both these methods could make the prediction much suitable and reliable.
- The most important for predicting stock market prices are Recurrent neural networks because they are able to learn nonlinear-mappings between inputs and outputs.
- It may be possible to perform better than traditional analysis and other computer based methods with the neural-networks ability to learn-nonlinear, chaotic-systems.

1.4 Overview of proposed solution approach

- Basically the main objective of this project is to collect the stock information for some previous years and then accordingly predict the results for the predicting what would happen next. So for we are going to use of two well-known techniques Recurrent neural network and t-SNE for stock market prediction. Extract useful information from a huge amount of data set and data mining is also able to predict future trends and behaviors through neural network. Therefore, combining both these techniques could make the prediction more suitable and much more reliable.
- As far as the solutions for the above problems, the answer depends on which way the forecast is used for. So the procedures that we will be using have proven to be very applicable to the task of forecasting product demand in a logistics system. Many techniques, which can prove useful for forecasting-problems, have shown to be inadequate to the task of demand forecasting in logistics systems.

Novelty/Benefits:

- Prediction of stock market is a common study.
- However grouping of stock portfolios based on daily close change patters is just a theory which almost no one has applied.
- Hence in this project we attempt clustering stocks based on weights, biases,loss on tensorflow and project the same using t-SNE method offered by tensorboard.

1.5 Comparative Study of Prediction Techniques Table 9.

Criteria	Technical Analysis	Fundamental Analysis	Traditional Time Series Analysis	Machine Learning Techniques
Data Used	Price, volume,highest, lowest prices	Growth, dividend payment, sales level, interest rates, tax rates etc.	Historical data	Set of sample data
Learning methods	Extraction of trading rules from charts	Simple trading rules extraction	Regression analysis on attributes is used	Inductive learning is used
Type of Tools	Charts are used	Trading rules	Simple Regression and Multivariate analysis used for time series.	Nearest neighbor and Neural Networks are used
Implementation	Daily basis prediction	Long –term basis prediction	Long –term basis prediction	Daily basis prediction

1.6 Details of Empirical Study:

Collection of Stock quotes, Analyst Advice and News:

The information of stock-market is collected once-a-day for the companies in NSE-&-BSE with-a-database of 1 year.

1. NSE-&-BSE Stock-Index-Dataset: The released data on financial-websites ,numerical-quotes of daily close.

2. It's collected from google-finance; which are downloaded using a python script as .csv file.

Numerical Representation

Quotes of stocks are subtracted from the previous close to get a slope of the direction the market is heading.

2. Literature Review

[1]STOCK PRICE PREDICTION USING LSTM,RNN AND CNN-SLIDING WINDOW MODEL

Sreelekshmy Selvin, Vinayakumar R, Gopalakrishnan E.A, Vijay Krishna Menon, Soman K.P
Centre for Computational Engineering and Networking (CEN)

The above paper presents an accurate model of how to use neural network for stock prediction.

The existing forecasting methods make useof both linear (AR,MA,ARIMA) and non-linear algorithms(ARCH,GARCH,Neural Networks),but they focus on predicting the stock index movement or price forecasting for a single company using the daily closing price. The proposed method here is a model independent approach. Here they are not fitting the data to a specific model, rather they are identifying the latent dynamics existing in the data using deep learning architectures. In this work they use three different deep learning architectures for the price prediction of NSE listed companies and compares their performance. They are applying a sliding window approach for predicting future values on a short term basis. The performance of the models were quantified using percentage error.

[2] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

The above blog post gives us the clearest understanding of the workings of LSTM

[3] https://lilianweng.github.io/lil-log/2017/07/08/predict-stock-prices-using-RNN-part-1.html

The above is used as the tutorial for our project.

3. Integrated Summary

The most interesting task is to predict the market. So many methods are used for completing this task. Methods, vary from very informal ways to many formal ways a lot. These tech. are categorized as Prediction Methods, Traditional Time Series, Tech Analysis Methods, Mach Learning Methods and Fundamental Analysis Methods. The criteria to this category is the kind of tool and the kind of data that these methods are consuming in order to predict the market. What is mutual to the technique is that they are predicting and hence helping from the market's future behaviour.

Technical Analysis Methods:

Method of guessing the correct time to vend or purchase a stock pricing. The reason behind tech analysis is that share prices move in developments uttered by the repetitively altering qualities of investors in answer to different forces. The tech data such as price, volume, peak and bottom prices per trade-off period is used for graphic representation to forecast future stock activities.

Fundamental Analysis Techniques:

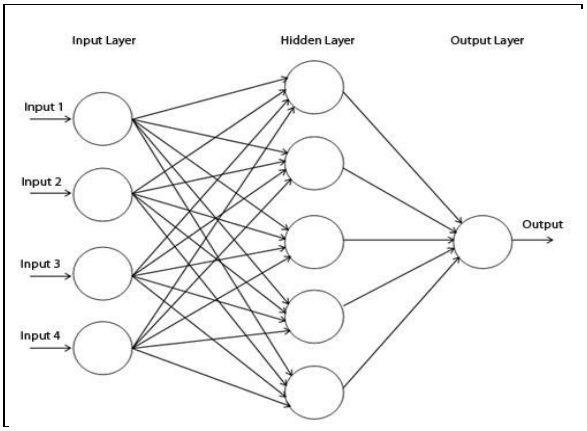
This practice uses the theory of firm foundation for preferred-stock selection. Data of fundamental analysis can be used by forecasters for using this tech of prediction for having a fully clear idea about the market or for investment. The growth, the bonus pay out, the IR, the risk of investing so on are the standards that will be used to get the real value for an asset in which they could finance in the market. Main target of this process is to determine an inherent value of an strength.

Traditional Time Series Prediction: Past data is used here and it uses this data to find coming values for the time series as a linear grouping. Use of Regression depictions have been used for forecasting stock market time series. Two rudimentary types of time series are simple and multivariate regressions.

Machine Learning Methods: The main reason is inductive learning. These types of methods use samples of data that is needed for creating an hope for the underling function that had produced all of the other data. Taking out a deduction from different samples which are given to the model is the main aim for this. The Nearest Neighbour and the Neural Networks Practices have been used for forecasting of the market.

Prediction Module

Multi-layered Feed-Forward network



This neural network has one layer of input, concealed layer, and one yield layer.

Input layer: Made up of units; the qualities measured for each drill tuple matches to the input to the network. Inputs are served to this layer instantaneously. The input passes through input layer and weighted & instantaneously served to the next layer i.e. hidden layer.

Hidden Layer: The productions of the input layer are input to this concealed layer. The number of concealed layer is random; in rehearsal only one concealed layer is used. The weighted output of the concealed layer are input to the next or output layer, which actually releases the network forecast for given tuples.

Output Layer: This layer actually discharges the network forecast for given tuples. Multilayer feed forward network are able to model the class forecast as a nonlinear grouping of the input. For given concealed units and enough preparation samples can carefully estimate to any function

Our Change to this generic module is that we use Recurrent Neural network(RNN)

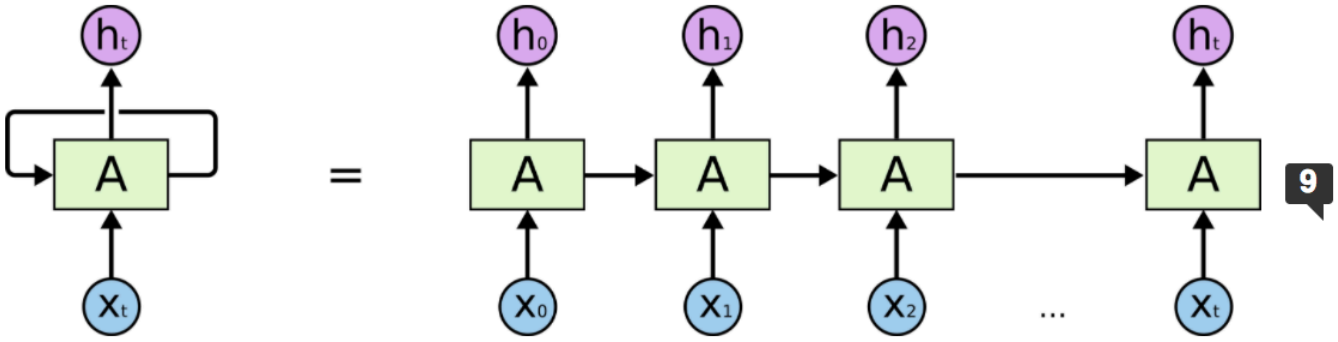
RNN:

Humans don't start their thinking from scratch every second. As you read this essay, you understand each word based on your understanding of previous words. You don't throw everything away and start thinking from scratch again. Your thoughts have persistence.

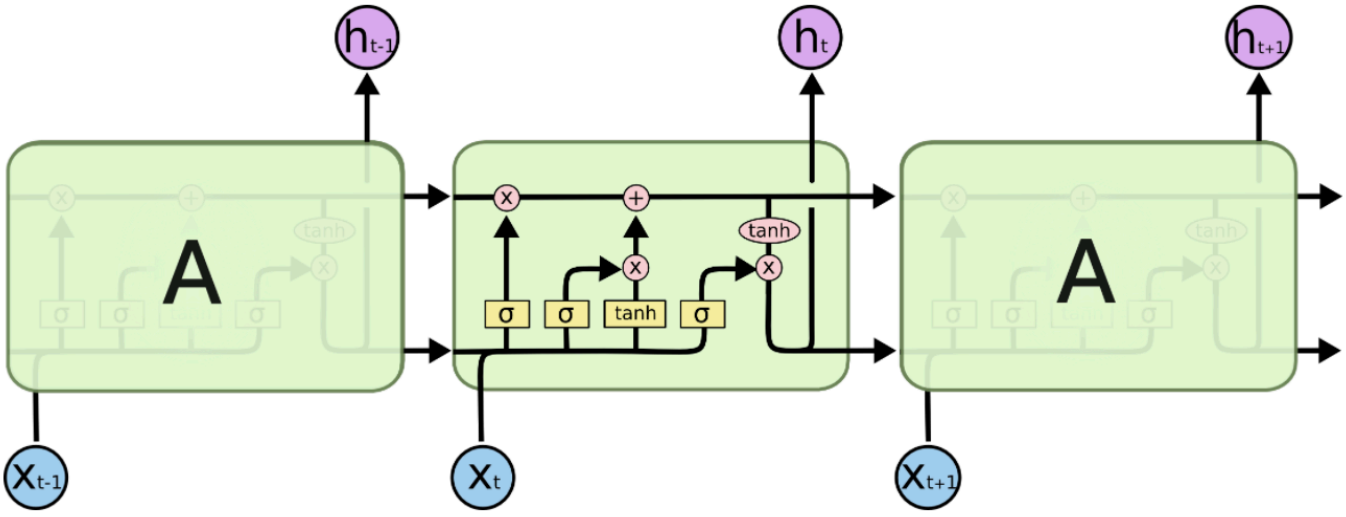
Traditional neural networks can't do this, and it seems like a major shortcoming. For example, imagine you want to classify what kind of event is happening at every point in a movie. It's unclear how a traditional neural network could use its reasoning about previous events in the film to inform later ones.

Recurrent neural networks address this issue. They are networks with loops in them, allowing information to persist.

Similar to the above Multilayer Network , but with an additional feature of looping in previous data.



An unrolled recurrent neural network.



The repeating module in an LSTM contains four interacting layers.

ANALYSIS, DESIGN & MODELING

3.1 Overall description of the project

Project is overall based upon the myraid data which is going to be mined from various stock related portals and after fetching the desired data they have been used for the predictions of related results.

Collection of Stock quotes

We are collecting the stock information once in day.

Nifty Stock-Index-Dataset of top 200 stocks (cnx200): The released data on financial-websites ,numerical-quotes of daily close. It's collected from google-finance; which are downloaded using a python script as .csv file.

Fundamental Data :

Meta-data about the list of stocks like industry they belong to etc.

3.2 FUNCTIONAL REQUIREMENTS:

The prediction shall abide by the following functional requirements:

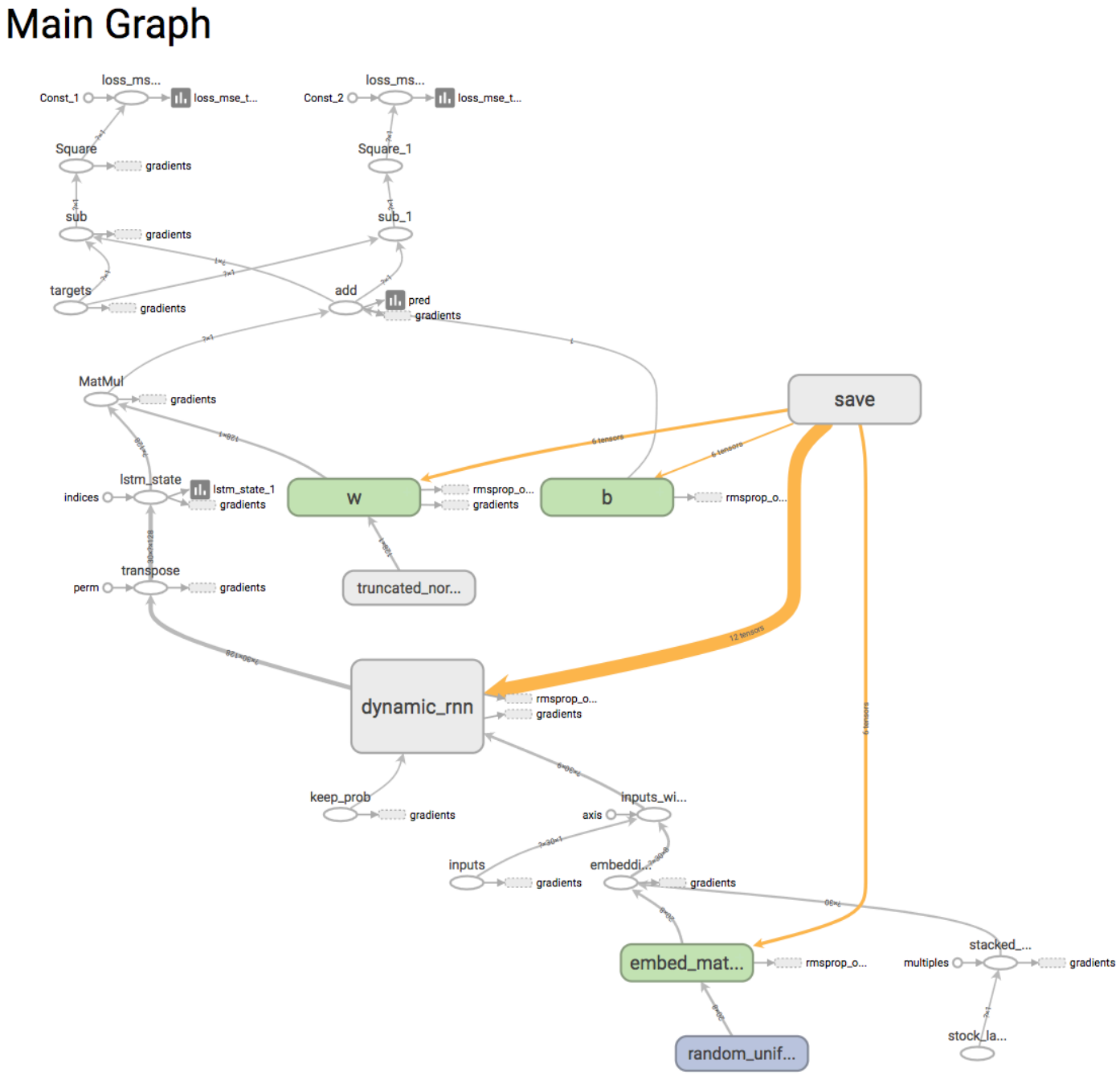
1. Prior to application of stock recommendations, the database is updated by the latest values.
2. The charts and comparison of the companies would be done only on the latest data stock market data.
3. Clusters of stock symbols are noted to check for their similarity.

3.3 NON FUNCTIONAL REQUIREMENTS:

1. **Reliability**: The reliability of the product will be dependent on the accuracy of the data- date of purchase, how much stock was purchased, high and low value range as well as opening and closing figures. Also the stock data used in the training would determine the reliability of the software.
2. **Security:** The user will only be able to access the website using his login details and will not be able to access the computations happening at the back end.
3. **Maintainability:** The maintenance of the product would require training of the software by recent data so that there commendations are up to date. The database has to be updated with recent values.
4. **Portability:** The website is completely portable and the recommendations completely trustworthy as the data is dynamically updated.
5. **Interoperability:** The interoperability of the website is very high because it synchronize all the database with the wamp server.

3.4 Design Diagrams

TensorFlow DFD



IMPLEMENTATION AND TESTING

Work-flow:

- Collect 7 years daily data for 200 stock portfolios.
- Subtract the value for two consecutive days to get the change in prices over night.
- Feed the data through 128 hidden layer model of RNN and train it for 6 years worth data.
- Test the model out on the last 1 year test data .
- Use TensorBoard to plot the summary statistics.
- Images of prediction vs. actual would be in the ./Images folder.
- ./Logs will have checkpoint data that can be viewed on Tensorboard.
- View the projection on Tensorboard and note the clusters formed by t-NSE

Risk Analysis and Mitigation Plan

1. Since, we are making software which involves updations/modifications, heavy computations and to and-fro activity may be required. But, the software should never take more than reasonable amount of time, which is the goal of the project. Although, it's a risk if it takes much time.

Probability: Low (1)

Impact: High (5)

2. There are light and background constraints for the application. Due to unavailability of resources or server, we might not be able to use the application.

Probability: High (5)

Impact: High (5)

3. We will never be able to check if our code is upto the requirement, for complex objects, as we will have to update/ modify time to time. Although, this risk can be reduced by machine learning by implementing automatic database updating of new scanned objects.

Probability: Low (1)

Impact: High (5)

Testing and requirement analysis

SOFTWARE REQUIREMNTS:

- Python 3.6
- Tensorflow 0.1.4
- Pandas
- Any normal browser to deploy TensorBoard

HARDWARE REQUIREMENTS(Minimum):

- 2.7 GHz Intel Core i5 -- Processor
- 8 GB 1867 MHz DDR3 -- Memory

Component decomposition and type of testing required

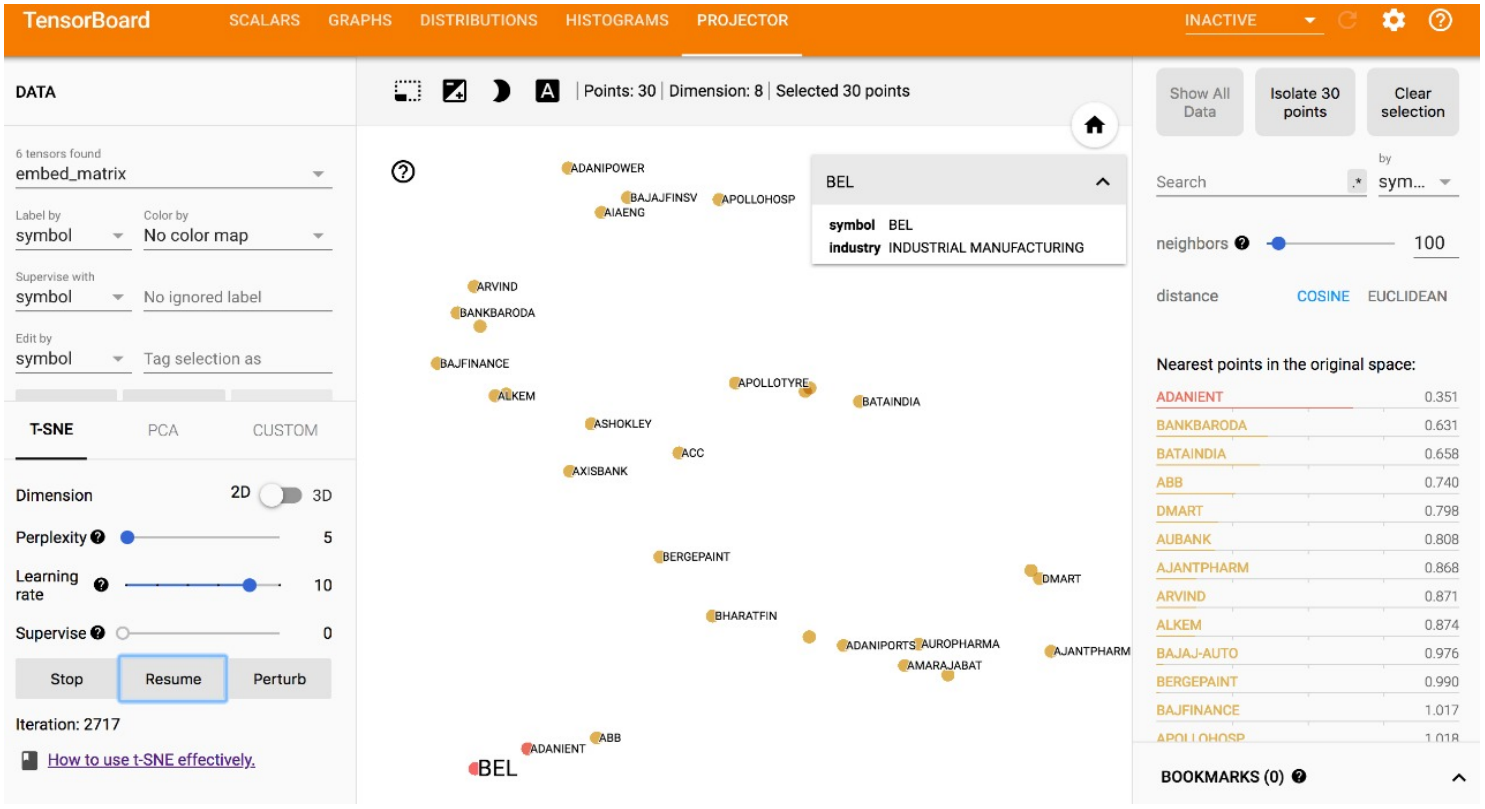
S No.	List of Various Components (modules) that require testing	Type of Testing Required	Technique for writing test cases
1	Data (Stock Price)	Requirement	Black Box(Boundary Values)
2	Algorithms (BP)	Unit	White Box
3	Neural Network (output	Unit	White Box
4	Graphs and Table	Volumes	White Box

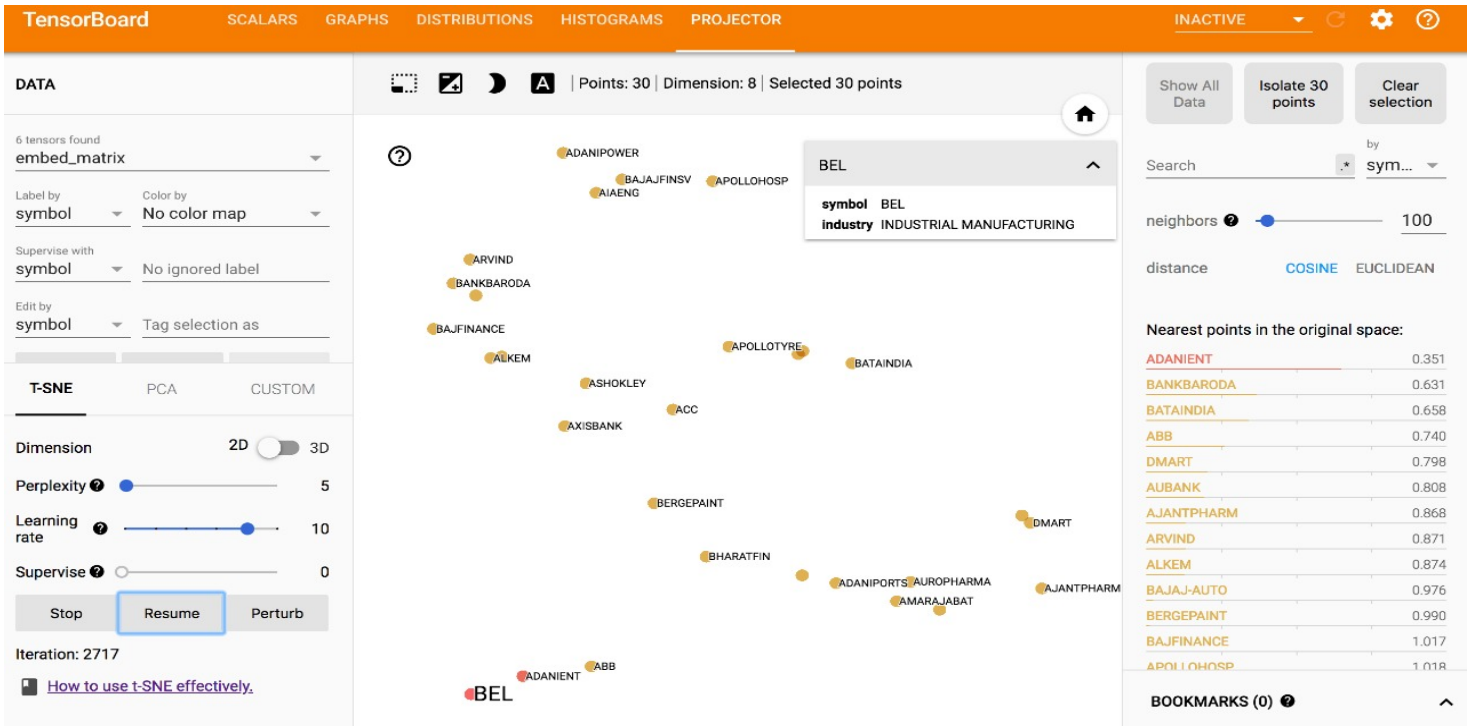
Screenshot of Output :

Training :

```
319, 320, 321, 322, 323, 324, 325])}
Start training for stocks: ['DALMIABHA', 'DHFL', 'DISHTV', 'DIVISLAB', 'LALPATHLAB', 'DRREDDY', 'EDELWEISS', 'EICHERMOT', 'EMAMILTD', 'ENDURANCE', 'ENG
INERSIN', 'EXIDEIND', 'FEDERALBNK', 'GAIL', 'GMRINFRA', 'GSKCONS', 'GLAXO', 'GLENMARK', 'GODREJCP', 'GODREJIND']
Step:1 [Epoch:0] [Learning rate: 0.001000] train_loss:2687.952637 test_loss:216.970230
Step:4001 [Epoch:8] [Learning rate: 0.000961] train_loss:106.212952 test_loss:216.993332
Step:8001 [Epoch:16] [Learning rate: 0.000886] train_loss:1212.211426 test_loss:223.897903
Step:12001 [Epoch:25] [Learning rate: 0.000810] train_loss:210.056305 test_loss:235.573959
Step:16001 [Epoch:33] [Learning rate: 0.000747] train_loss:4.368201 test_loss:262.348999
_
```

Tensorboard display of stochastic distribution :

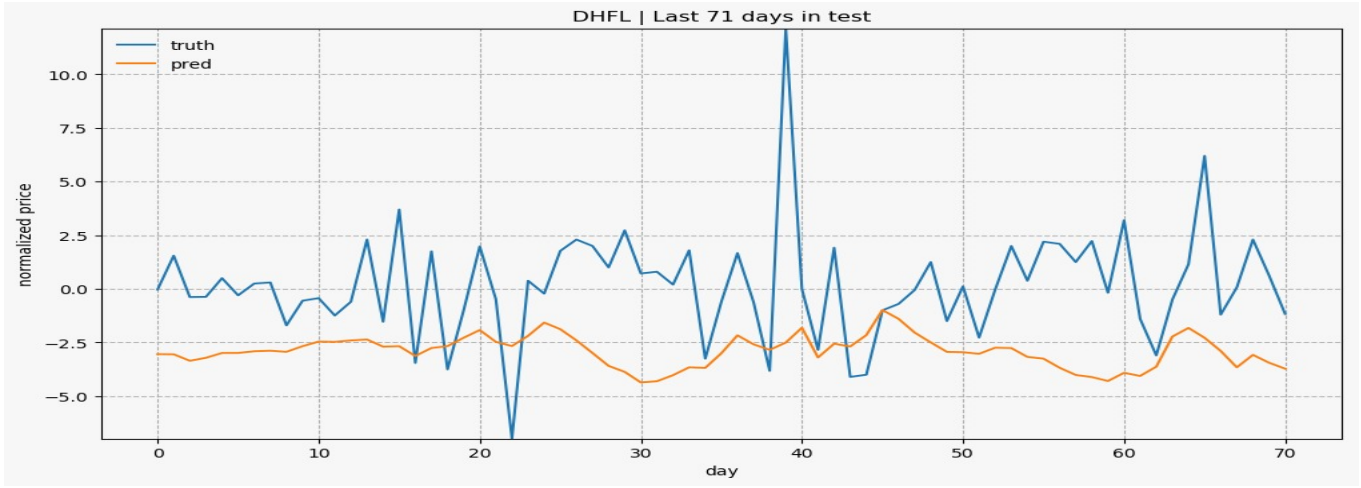


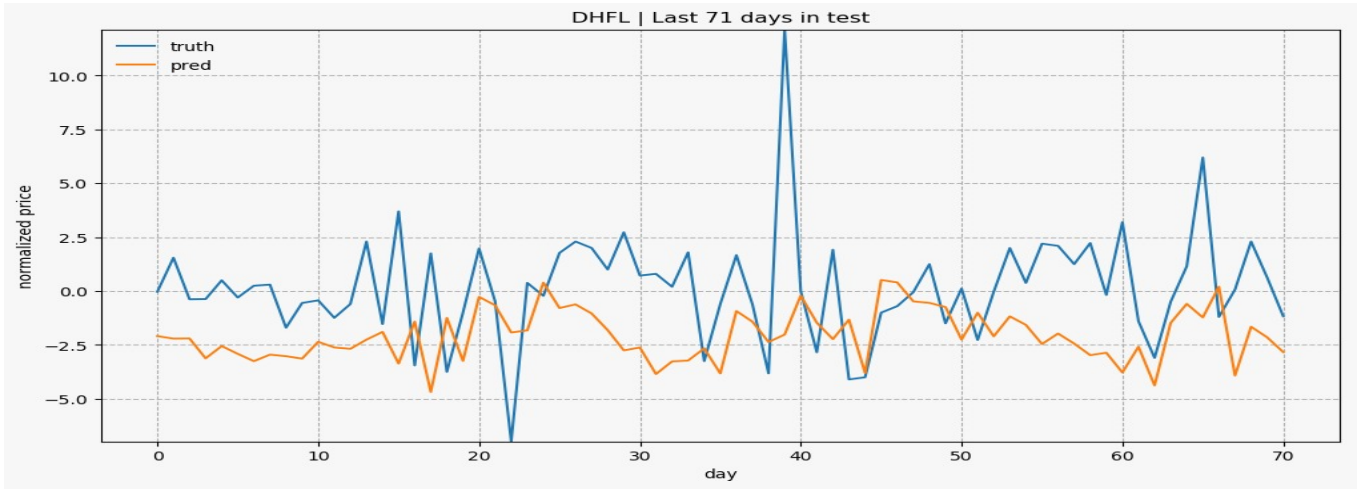


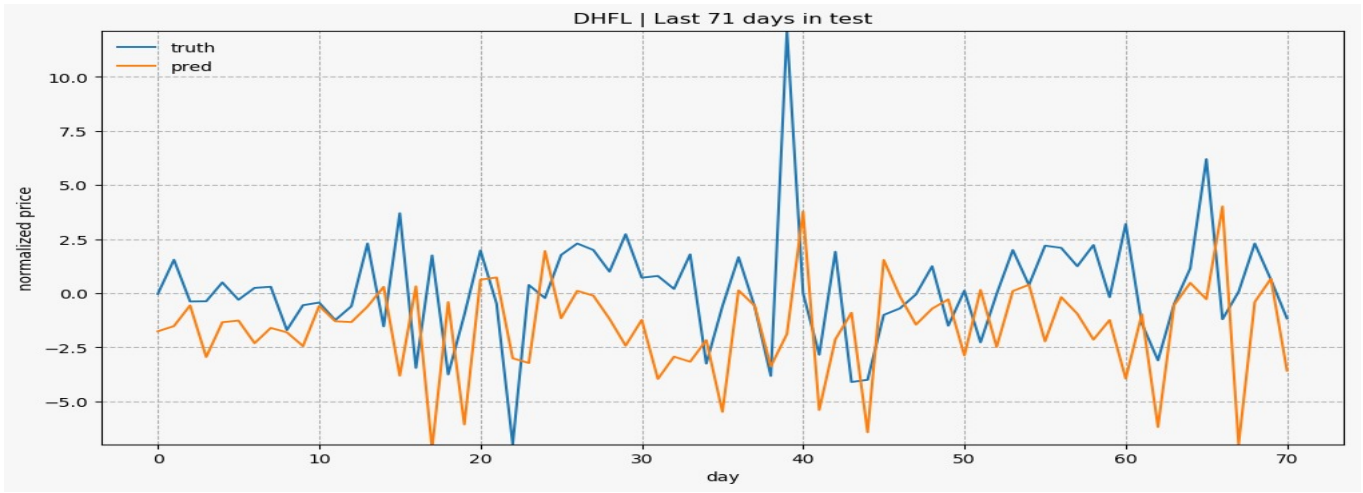
Result/Discussions(tensorboard) :

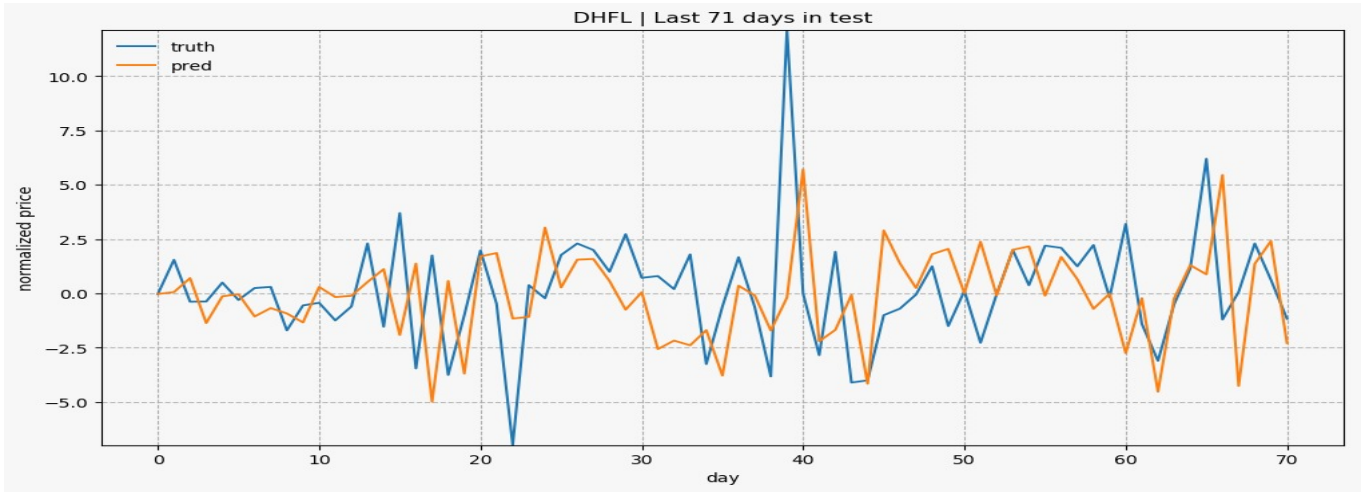
- From the above clusters it is clear that some pairs of stocks have common patters as identified above. These patters remain the same over multiple epochs .
- We see that despite difference in industrial sector, BEL, ADANIENT and ABB tend to show the same change in patters over the last 7 years .Hence they are statistically neighbours.
- The above graph is actually a 2D representation of a 3D figure hence we see a marked difference in the positioning to bank of baroda even thoug its second closest to BEL.

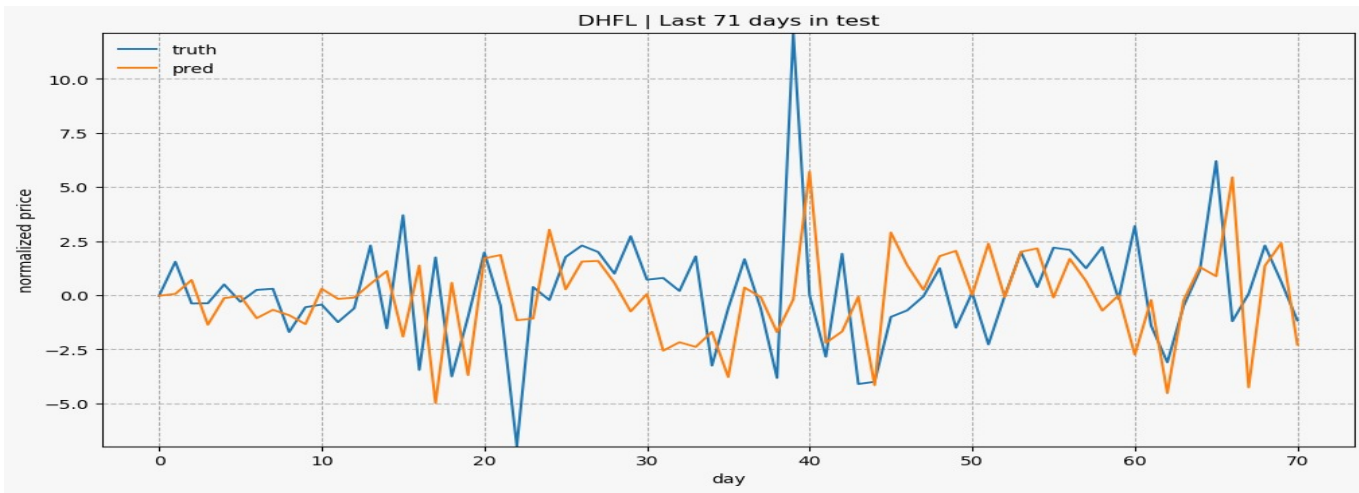
Given Below is the stock name DHFL's Prediction on over 50 epochs :




DHFL | Last 71 days in test
truth
pred
normalized price
day


DHFL | Last 71 days in test
truth
pred
normalized price
day


DHFL | Last 71 days in test
truth
pred
normalized price
day


DHFL | Last 71 days in test
truth
pred
normalized price
day

Result/Discussions(prediction graph) :

- The above graph is a truth vs prediction comparison of DHFL on over 50 epochs. The data represented here is the change of stock prices over a day.
- We see that other than a few glitches at some point the graph is almost correct at predicting values.
- Given more epochs to run we are sure to find better results.

Limitations of the Solutions

The solution has a few limitations which are relevant to the proper functioning of our application:

1. Normal Neural network applications require scaling to be done to the input values . In our model we have not scaled the data , instead we directly took the difference of close prices between days. Though it is not an accurate way of training the RNN, it helps us cluster them accurately using t-SNE.
2. Blindly following data is not always accurate . Some of the clusters formed make no sense in the real world. In fact totally opposite stock porfilios are shown to be clustered together..
3. Using LSTM or long stort term memory we need to look back almost 4-5 years to predict 200 days worth data. Such a method is against the practices of traditional analysts who preach that to predict prices of 1 year look back only 1 year and not more.

CONCLUSION

Evaluating the Stock market prediction has at all times been tough work for analysts. Thus, we attempt to make use of vast written data to forecast the stock market indices. If we join both techniques of textual mining and numeric time series analysis the accuracy in predictions can be achieved. Recurrent neural network is qualified to forecast NIFTY market upcoming trends. Financial analysts, investors can use this prediction model to take trading decision by observing market behaviour.

FUTURE WORK

- More customized model to adapt to stock data.
- Twitter feeds message board, Extracting RSS feeds and news, for sentiment analysis.
- Considering internal factors of the company likes Sales, Assets etc.

References

[1] A. V. Devadoss and T. A. A. Ligori, “Forecasting of stock prices using multi layer perceptron,” Int J Comput Algorithm, vol. 2, pp. 440–449, 2013.

[2] J. G. De Gooijer and R. J. Hyndman, “25 years of time series forecasting,” International journal of forecasting, vol. 22, no. 3, pp. 443–473, 2006.

[3] V. K. Menon, N. C. Vasireddy, S. A. Jami, V. T. N. Pedamallu, V. Sureshkumar, and K. Soman, “Bulk price forecasting using spark over nse data set,” in International Conference on Data Mining and Big Data. Springer, 2016, pp. 137–146.

[4] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, Time series analysis: forecasting and control. John Wiley & Sons, 2015.

Code:

CsvLoader.py

```
from googlefinance.client import get_price_data
import pandas as pd
# uses google financial api to get last 7 years weekly close

#list = ["ABB",""]

df = pd.read_csv("ind_nifty200list.csv")
df= df.loc[:,"Symbol"]

#print(df)
list = df.tolist()
#print(list)
interval=60*60*24

period = "7Y"

for i in list:
    #print(i+"\n")
    param = {
    'q': i, # Stock symbol (ex: "AAPL")
    'i': interval, # Interval size in seconds ("86400" = 1 day intervals)
    'x': "NSE", # Stock exchange symbol on which stock is traded (ex: "NASD")
    'p': period # Period (Ex: "1Y" = 1 year)
    }

    df = get_price_data(param)
    df = df.sort_index(ascending=False, axis=0)
    print(df.head())
    print("making csv for "+i+"\n")
    df.to_csv('/Users/debashiskarmakar/Documents/Sem6/stock-rnn-master/data_indian/'+i+'.csv',
            encoding='utf-8')
```

data_model.py

```
import numpy as np
import os
import pandas as pd
```

```
import random
import time

random.seed(time.time())

class StockDataSet(object):
    def __init__(self,
                 stock_sym,
                 input_size=1,
                 num_steps=30,
                 test_ratio=0.1,
                 normalized=False,  #change made here
                 close_price_only=True):
        self.stock_sym = stock_sym
        self.input_size = input_size
        self.num_steps = num_steps
        self.test_ratio = test_ratio
        self.close_price_only = close_price_only
        self.normalized = normalized

        # Read csv file
        #raw_df = pd.read_csv(os.path.join("data", "%s.csv" % stock_sym))
        raw_df = pd.read_csv(os.path.join("data_indian", "%s.csv" % stock_sym))
        # Merge into one sequence
        if close_price_only:
            self.raw_seq = raw_df['Close'].tolist()
        else:
            self.raw_seq = [price for tup in raw_df[['Open', 'Close']].values for price in tup]

        #modified here
        B = []
        A = self.raw_seq
        for i in range(1,len(A)-1):
            diff = A[i] - A[i-1]
            B.append(diff)

        self.raw_seq = B
        #print(B)
```

```
    #till here

    self.raw_seq = np.array(self.raw_seq)
    self.train_X, self.train_y, self.test_X, self.test_y = self._prepare_data(self.raw_seq)

def info(self):
    return "StockDataSet [%s] train: %d test: %d" % (
        self.stock_sym, len(self.train_X), len(self.test_y))

def _prepare_data(self, seq):
    # split into items of input_size
    seq = [np.array(seq[i * self.input_size: (i + 1) * self.input_size])
           for i in range(len(seq) // self.input_size)]

    if self.normalized:
        seq = [seq[0] / seq[0][0] - 1.0] + [
            curr / seq[i][-1] - 1.0 for i, curr in enumerate(seq[1:])]

    # split into groups of num_steps
    X = np.array([seq[i: i + self.num_steps] for i in range(len(seq) - self.num_steps)])
    y = np.array([seq[i + self.num_steps] for i in range(len(seq) - self.num_steps)])

    train_size = int(len(X) * (1.0 - self.test_ratio))
    train_X, test_X = X[:train_size], X[train_size:]
    train_y, test_y = y[:train_size], y[train_size:]
    return train_X, train_y, test_X, test_y

def generate_one_epoch(self, batch_size):
    num_batches = int(len(self.train_X)) // batch_size
    if batch_size * num_batches < len(self.train_X):
        num_batches += 1

    batch_indices = list(range(num_batches))
    random.shuffle(batch_indices)
    for j in batch_indices:
        batch_X = self.train_X[j * batch_size: (j + 1) * batch_size]
        batch_y = self.train_y[j * batch_size: (j + 1) * batch_size]
        assert set(map(len, batch_X)) == {self.num_steps}
```

```
        yield batch_X, batch_y
```

Main.py

```
import os
import pandas as pd
import pprint

import tensorflow as tf
import tensorflow.contrib.slim as slim

from data_model import StockDataSet
from model_rnn import LstmRNN

flags = tf.app.flags
flags.DEFINE_integer("stock_count", 100, "Stock count [100]")
flags.DEFINE_integer("input_size", 1, "Input size [1]")
flags.DEFINE_integer("num_steps", 30, "Num of steps [30]")
flags.DEFINE_integer("num_layers", 1, "Num of layer [1]")
flags.DEFINE_integer("lstm_size", 128, "Size of one LSTM cell [128]")
flags.DEFINE_integer("batch_size", 64, "The size of batch images [64]")
flags.DEFINE_float("keep_prob", 0.8, "Keep probability of dropout layer. [0.8]")
flags.DEFINE_float("init_learning_rate", 0.001, "Initial learning rate at early stage. [0.001]")
flags.DEFINE_float("learning_rate_decay", 0.99, "Decay rate of learning rate. [0.99]")
flags.DEFINE_integer("init_epoch", 5, "Num. of epoches considered as early stage. [5]")
flags.DEFINE_integer("max_epoch", 50, "Total training epoches. [50]")
flags.DEFINE_integer("embed_size", None, "If provided, use embedding vector of this size. [None]")
flags.DEFINE_string("stock_symbol", None, "Target stock symbol [None]")
flags.DEFINE_integer("sample_size", 4, "Number of stocks to plot during training. [4]")
flags.DEFINE_boolean("train", False, "True for training, False for testing [False]")

FLAGS = flags.FLAGS

pp = pprint.PrettyPrinter()

if not os.path.exists("logs"):
    os.mkdir("logs")


def show_all_variables():
```

```
    model_vars = tf.trainable_variables()
    slim.model_analyzer.analyze_vars(model_vars, print_info=True)


def load_sp500(input_size, num_steps, k=None, target_symbol=None, test_ratio=0.05):
    if target_symbol is not None:
        return [
            StockDataSet(
                target_symbol,
                input_size=input_size,
                num_steps=num_steps,
                test_ratio=test_ratio)
        ]

    # Load metadata of s & p 500 stocks
    #info = pd.read_csv("data/constituents-financials.csv")
    info = pd.read_csv("data_indian/ind_nifty200list.csv")
    info = info.rename(columns={col: col.lower().replace(' ', '_') for col in info.columns})
    #info['file_exists'] = info['symbol'].map(lambda x: os.path.exists("data/{}.csv".format(x)))
    info['file_exists'] = info['symbol'].map(lambda x: os.path.exists("data_indian/{}.csv".format(x)))
    print(info['file_exists'].value_counts().to_dict())

    info = info[info['file_exists'] == True].reset_index(drop=True)
    #info = info.sort_values('market_cap', ascending=False).reset_index(drop=True)

    if k is not None:
        info = info.head(k)

    #print("Head of S&P 500 info:\n", info.head())
    print("Head of ind_nifty200list info:\n", info.head())

    # Generate embedding meta file
    #info[['symbol', 'sector']].to_csv(os.path.join("logs/metadata.tsv"), sep='\t', index=False)
    info[['symbol', 'industry']].to_csv(os.path.join("logs/metadata.tsv"), sep='\t', index=False)

    return [
        StockDataSet(row['symbol'],
                     input_size=input_size,
                     num_steps=num_steps,
```

```
                test_ratio=0.05)
        for _, row in info.iterrows()]

def main(_):
    pp.pprint(flags.FLAGS.__flags)

    # gpu_options = tf.GPUOptions(per_process_gpu_memory_fraction=0.333)
    run_config = tf.ConfigProto()
    run_config.gpu_options.allow_growth = True

    with tf.Session(config=run_config) as sess:
        rnn_model = LstmRNN(
            sess,
            FLAGS.stock_count,
            lstm_size=FLAGS.lstm_size,
            num_layers=FLAGS.num_layers,
            num_steps=FLAGS.num_steps,
            input_size=FLAGS.input_size,
            embed_size=FLAGS.embed_size,
        )

        show_all_variables()

        stock_data_list = load_sp500(
            FLAGS.input_size,
            FLAGS.num_steps,
            k=FLAGS.stock_count,
            target_symbol=FLAGS.stock_symbol,
        )

        if FLAGS.train:
            rnn_model.train(stock_data_list, FLAGS)
        else:
            if not rnn_model.load()[0]:
                raise Exception("[!] Train a model first, then run test mode")


if __name__ == '__main__':
    tf.app.run()
```