Stock Market Prediction using Natural Language Processing

Course Title: Natural Language Processing

Course Code: CSE4022

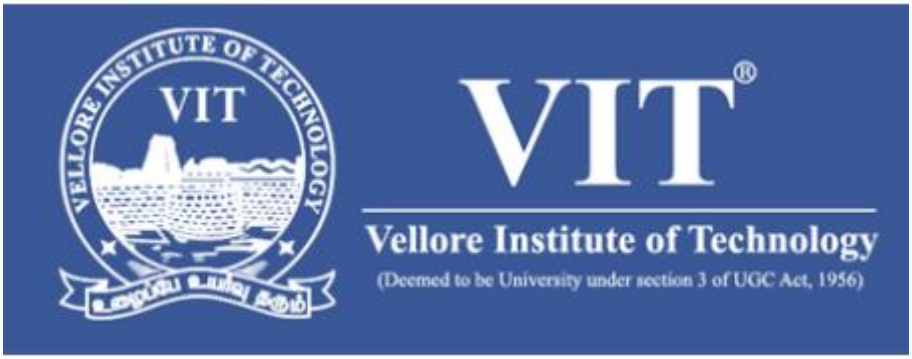
Submitted By:

Sno.	Name	Register No.
1	Harshit Mahajan	16BCI0154
2	Chiluka Nag Lohith	17BCE0219
3	Sarthak Ghoshal	17BCE2219

Slot: C1 + TC1

Name of the Faculty: Prof. Rajeshkannan R.

SCHOOL OF COMPUTER SCIENCE & ENGINEERING



Abstract

In order to make prediction in stock trends, we will use Machine learning techniques to evaluate past data pertaining to the stock market and world affairs of the corresponding time period. We will build a model that will be able to buy and sell stock based on profitable prediction, without any human interactions. The model uses Natural Language Processing (NLP) to make smart "decisions" based on current affairs, article, etc. With NLP and the basic rule of probability, our goal is to increases the accuracy of the stock predictions. Good and effective prediction systems for stock market help traders, investors, and analyst by providing supportive information like the future direction of the stock market. In this work, we present a recurrent neural network (RNN) combine with convolution neural network (CNN) approach to predict stock market indices.

Introduction

Natural Language Processing is a technique used by a computer to understand and manipulate natural languages. By natural languages, we mean all human derived languages. Natural language processing or NLP for short is used to analyse text and let machines derive meaning from the input. This human-computer interaction allows us to come up with many different applications to bring man and machine as one. For example, on Google, if we use Google translation, that is NLP and so is speech recognition. In our project, we make use of some established NLP techniques to evaluate past data pertaining to the stock market and world affairs of the corresponding time period, in order to make predictions in stock trends.

In order to proceed with this objective, we needed to understand what Sentimental Analysis is. Sentimental Analysis is an analytical method that the computer uses to understand a natural language and deduce if the message is positive, neutral or negative. In our case, Sentimental analysis refers to the deduction of the news headlines if they increase the stock or reduce it. By doing so, we end up with the 'emotional' status of the data which is what sentimental analysis gives its user.

Literature Survey

Making investment decisions in stock market is risky sometimes because it is the fastest and also easiest way of making money as well as losing money. Therefore, investing on stock market needs careful planning with deep analysis which now a days is possible using advanced technologies with large computational power, neural network, relational database etc.

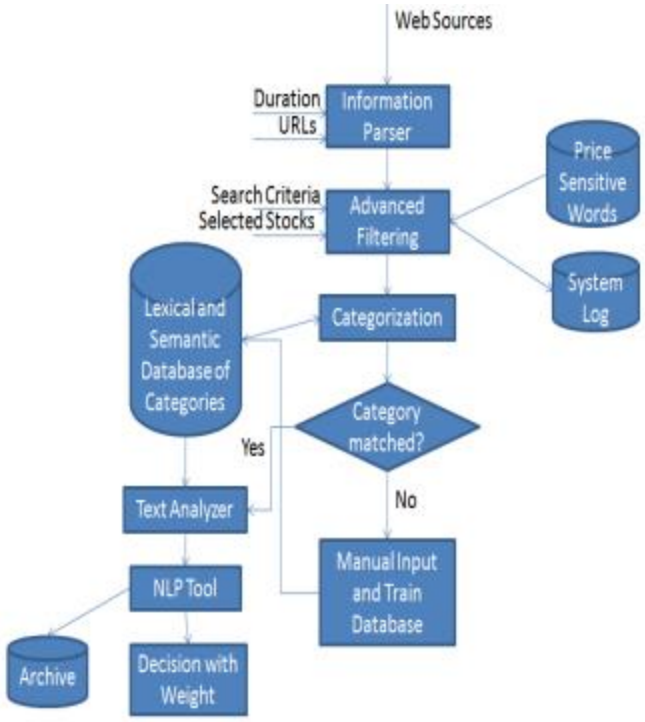
Stock market analysis can be separated into two categories. One is Technical and another one is fundamental. Technical analysis looks at the price trend of a security and uses this data to forecast its future price movements where fundamental analysis, on the other hand, looks at economic factors, known as fundamentals. Though both type of analysis are important but technicians believe that all the information they need about a stock can be found in its charts and therefore technical traders, believe there is no reason to analyze a company's fundamentals because these are all accounted for in the stock's price.

However to understand the long term performance of a stock, to estimate the risk factors involve d with an investment, to understand the entry and exit point in a volatile stock market, fundamental analysis is mandatory and news, records, rumors about stocks can provide necessary input for fundamental analysis. In this research, we applied text mining techniques on different news and articles published by legitimate companies on internet to extract high quality information which ultimately made it possible to analyze, decide and also update our database for other type of analysis. We proposed a framework for information gathering and processing where we used a natural language processing tool with our suggested algorithms.

In 2002, Antonina Kloptchenko, Tomas Eklund, Barbro Back, Jonas Karlsson, Hannu Vanharanta and Ari Visa used data and text mining techniques to study hidden patterns about the future financial performance of companies from the quantitative and qualitative parts of their financial reports. Because from their point of view, annual reports are an important medium for the company's communication with its stakeholders . However, their text mining approach can only be applied on annual reports of a company, not web sites or news achieve. Later on in 2006, Marc-André Mittermayer and Gerhard F. Knolmayer performed a survey on text mining systems for predicting market response to news where they briefly described and compared eight existing prototypes in chronological order . But none of the techniques showed mechanism of retrieving decisions from reumors using natural language processing.

On the same year Robert P. Schumaker and Hsinchun Chen introduced a predictive machine learning approach AZFinText System for financial news articles analysis using several different textual representations like Bag of Words, Noun Phrases, Named Entities and investigated 9,211 financial news articles and 10,259,042 stock quotes during a five week period. Their technique was very efficient and rapid but they did not show any decision making process (based on historic data) from the retrieved information.

In 2007, Vatsal H. Shah and Dr. Mehryar Mohri presented a machine learning techniques for stock prediction where they particularly discussed the application of Support Vector Machines, Linear Regression, prediction using decision stumps, expert weighting and online learning in detail along with the benefits and pitfalls of each method. The machine learning techniques they discussed were slightly focused on text and language processing. Nan Li and Desheng Dash Wu presented text sentiment analysis, also called emotional polarity computation where they studied online forums hotspot detection and forecast using emotion analysis and text mining approaches in early 2009. Their technique showed a totally a different idea of forecasting compare to other techniques. Afterward in 2009, Xiangyu Tang and Chunyu Yang, Jie Zhou proposed an algorithm for stock price forecasting by combining news mining and time series analysis. The idea of using time series analysis with text mining was discussed in their work.



Problem Description

There are a lot of complicated financial indicators and also the fluctuation of the stock market is highly violent. However, as the technology is getting advanced, the opportunity to gain a steady fortune from the stock market is increased and it also helps experts to find out the most informative indicators to make a better prediction. The prediction of the market value is of great importance to help in maximizing the profit of stock option purchase while keeping the risk low.

Datasets

- **Reuters.csv**

  Reuters, the news and media division of Thomson Reuters is the world's largest international multimedia news provider reaching more than one billion people every day. Reuters provides trusted business, financial, national, and international news to professionals via Thomson Reuters desktops, the world's media organizations, and directly to consumers at Reuters.com and via Reuters TV.

  After crawling data from https://www.reuters.com/ from year 2004 to 2019, our dataset look something like this:

Ticker	Company	Publish Date	Headline	First sent	Category
AA	Alcoa Corporation	20110707	Alcoa profit see higher o aluminum pric	*Analysts expec profit of 34 cts/sh vs year-ago 13 cts	topStory

			surge		
C	Citigroup Inc	20120308	CORRECTED-Citigroup CEO Pandit collect $14.9 million	NEW YORK Citigroup Chie Executive Vikran Pandit finally go his payday. The third biggest U.S bank company paic Pandit $14.8(million in 201 compared with $1 in 2010 according to a filing with the Securities and Exchange Committee.	topStory
DIS	Walt Disney Company (The)	20130807	US STOCKS Wall Street drop for a third day o Fed concerns	NEW YORK Aug 7 Major U.S. stock indexes fell for a third straight day o Wednesday a concerns grew ove the longevity of the Federal Reserve' stimulus policy which has beer widely credited with fueling the market's gains thi year.	normal

- **StockReturns.json**

  Trading data from Yahoo! Finance was collected and then normalized using the S&P 500. This is stored in the stockReturns.json file.

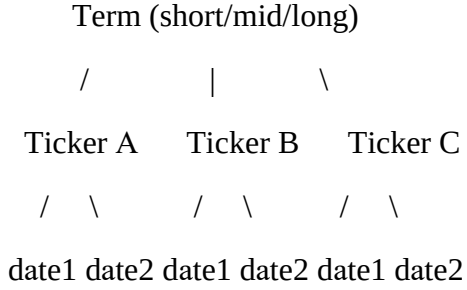

```
{
    “short”:
        {
                "HTGX": {
                "20140722": -0.0099,
                "20140723": 0.0014,
                "20140724": -0.0005,
                "20140725": 0.0031,
                "20140729": 0.0036 }
        }
     “mid”:
```

```
        {
                "HTGX": {
                "20140722": -0.0049,
                "20140723": 0.0087,
                "20140724": 0.0068,
                "20140725": 0.0248,
                "20140729": 0.0208 }
         }
        “long”:
         {
                   "HTGX": {
                  "20140722": -0.0008,
                  "20140723": 0.0041,
                  "20140724": 0.0003,
                  "20140725": -0.0022,
                  "20140729": -0.0091 }
         }

    }
```


In our dataset, the ticker for the S&P is ^GSPC. Each ticker is compared the S&P and then judged on whether it is outperforming (positive value) or under-performing (negative value) the S&P. Each value is reported on a daily interval from 2004 to now.

Below is a diagram of the data in the json file. Note there are three types of data: short: 1 day return, mid: 7 day return, long 28 day return.



- **glove.6b.100d.txt**

  GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

  This dataset is downloaded from https://kaggle.com .

  Sample of the dataset is given below. Note that this is 100 dimension datset.

the 0.418 0.24968 -0.41242 0.1217 0.34527 -0.044457 -0.49688 -0.17862 -0.00066023 -0.6566 0.27843 -0.14767 -0.55677 0.14658 -0.0095095 0.011658 0.10204 -0.12792 -0.8443 -0.12181 -0.016801 -0.33279 -0.1552 -0.23131 -0.19181 -1.8823 -0.76746 0.099051 -0.42125 -0.19526 4.0071 -0.18594 -0.52287 -0.31681 0.00059213 0.0074449 0.17778 -0.15897 0.012041 -0.054223 -0.29871 -0.15749 -0.34758 -0.045637 -0.44251 0.18785 0.0027849 -0.18411 -0.11514 -0.78581
, 0.013441 0.23682 -0.16899 0.40951 0.63812 0.47709 -0.42852 -0.55641 -0.364 -0.23938 0.13001 -0.063734 -0.39575 -0.48162 0.23291 0.090201 -0.13324 0.078639 -0.41634 -0.15428 0.10068 0.48891 0.31226 -0.1252 -0.037512 -1.5179 0.12612 -0.02442 -0.042961 -0.28351 3.5416 -0.11956 -0.014533 -0.1499 0.21864 -0.33412 -0.13872 0.31806 0.70358 0.44858 -0.080262 0.63003 0.32111 -0.46765 0.22786 0.36034 -0.37818 -0.56657 0.044691 0.30392

Algorithm Steps

Data Pre-processing →

First we will reformat the StockReturns.json file's data and covert that data from continuous numeric data normalized to S&P returns into binary (positive/negative) i.e. is if the value is greater or equal to 1 then replace it with '1' or '0' otherwise. Then it will be cleanY.

After reformat it will look like this:

Ticker	Pub_Date	Price	Y
AAPL	20040708	0.0015	1

Now we will merge cleanY and reruters.csv file. Merging will be done by the value of 'Pub_date' and 'Ticker'.

After merging our merged dataset look like this:

Ticker	Company	Pub_Date	Headline	First_sent	Category	Price	Y
AAPL	Apple Inc	20110719	Instant view: Apple revenue again blows past	NEW YORK Apple In posted quarterly revenue	normal	-0.0082	0

Now cleaning of each column will take place. Some cleaning of datset is:

- Replacing double spaces into a single space
- Replace U.S. to United States so U won't get deleted with next replacement
- Remove all capitalized words at the beginning of the sentence, since those are mostly places (aka NEW YORK)
- Remove unnecessary punctuation (hyphens and asterisks)
- Remove dates

Now turn ‘headline’ and ‘first_sent’ column into tokens and merged it into out ‘merged’ dataset. It means tokenizing string into words. And group whole dataset group by ‘ticker’ and ‘pub_date’.

Remove observation with missing stock prices.

After combining ‘headline_token’ and ‘first_sent_token’ and merged it into our ‘finalData’ variable it look like this:

ticker	AAPL
company	Apple Inc
pub_date	20111101
headline	Telecom KT to launch iPhone 4S in S.Korea
first_sent	South Korea's top mobile carrier SK Telecom a...
category	topStory
price	0.0077
y	1
headline_token	[Telecom, KT, to, launch, iPhone, 4S, in, S, K...
first_sent_token	[South, Korea's, top, mobile, carrier, SK, Tel...
final_text	[Telecom, KT, to, launch, iPhone, 4S, in, S, K...

Name: (AAPL, 20111101, 784), dtype: objectis:

Training and Testing dataset →

Split the data into training set and testing set of X and y.

X = ['Samsung estimates Q2 profit down 26 pct ', 'Canada eyes Nortel patent sale review-minister ', 'Analysis: Young startups demand steeper prices from VCs ']

y = [1.0, 1.0, 1.0, 0.0]

Now we will fit tokenizer on X_train and X_test. For this we will use two methods from “*keras.preprocessing.text.Tokenizer”,* they are:

1) **fit_on_texts:** Updates internal vocabulary based on a list of texts. This method creates the vocabulary index based on word frequency. So if you give it something like, "The cat sat on the mat." It will create a dictionary s.t. word_index["the"] = 1; word_index["cat"] = 2 it is word -> index dictionary so every word gets a unique integer value. 0 is reserved for padding. So lower integer means more frequent word (often the first few are stop words because they appear a lot).

2) **texts_to_sequences:** Transforms each text in texts to a sequence of integers. So it basically takes each word in the text and replaces it with its corresponding integer value from the word_index dictionary. Nothing more, nothing less, certainly no magic involved.

And then we will convert y_train into one-hot encoded version

Build the model →

The two most important things in our code are the following:

1) The Embedding layer and
2) Keras embedding layer

1. Word Embedding

A word embedding is a class of approaches for representing words and documents using a dense vector representation.

It is an improvement over more the traditional bag-of-word model encoding schemes where large sparse vectors were used to represent each word or to score each word within a vector to represent an entire vocabulary. These representations were sparse because the vocabularies were vast and a given word or document would be represented by a large vector comprised mostly of zero values.

Instead, in an embedding, words are represented by dense vectors where a vector represents the projection of the word into a continuous vector space.

The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used.

The position of a word in the learned vector space is referred to as its embedding.

Two popular examples of methods of learning word embedding from text include:

- Word2Vec.
- GloVe.

In addition to these carefully designed methods, a word embedding can be learned as part of a deep learning model. This can be a slower approach, but tailors the model to a specific training dataset.

2. Keras Embedding Layer

Keras offers an Embedding layer that can be used for neural networks on text data.

It requires that the input data be integer encoded, so that each word is represented by a unique integer. This data preparation step can be performed using the Tokenizer API also provided with Keras.

The Embedding layer is initialized with random weights and will learn an embedding for all of the words in the training dataset.

It is a flexible layer that can be used in a variety of ways, such as:

- It can be used alone to learn a word embedding that can be saved and used in another model later.
- It can be used as part of a deep learning model where the embedding is learned along with the model itself.
- It can be used to load a pre-trained word embedding model, a type of transfer learning.

The Embedding layer is defined as the first hidden layer of a network. It must specify 3 arguments:

It must specify 3 arguments:

input_dim: This is the size of the vocabulary in the text data. For example, if your data is integer encoded to values between 0-10, then the size of the vocabulary would be 11 words.

output_dim: This is the size of the vector space in which words will be embedded. It defines the size of the output vectors from this layer for each word. For example, it could be 32 or 100 or even larger. Test different values for your problem.

input_length: This is the length of input sequences, as you would define for any input layer of a Keras model. For example, if all of your input documents are comprised of 1000 words, this would be 1000.

Training Neural Network

Recurrent Neural Network (RNN) →

In this stage, the data is fed to the neural network and trained for prediction assigning random biases and weights. Our LSTM model is composed of a sequential input layer followed by 3 LSTM layers and dense layer with activation then a dropout layer and then finally a dense output layer with linear activation function.

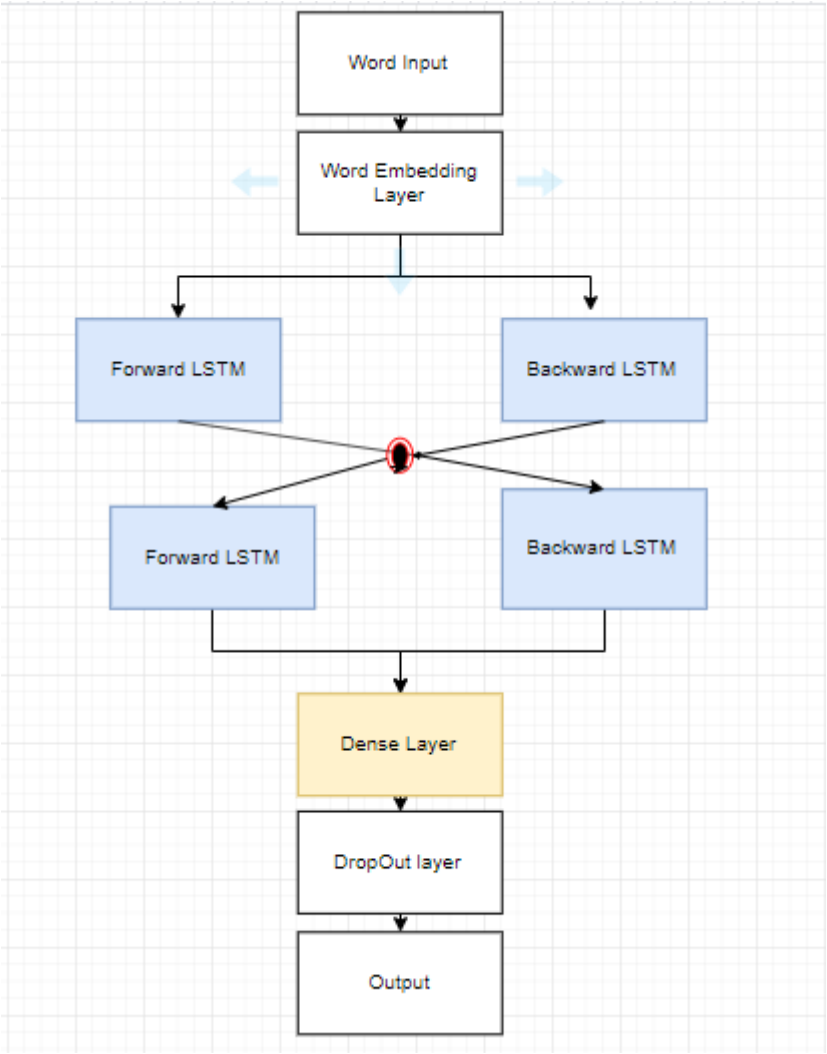


Fig: Architecture of Model

Word_Embedding_Layer

The Embedding layer is used to create word vectors for incoming words. It sits between the input and the LSTM layer, i.e. the output of the Embedding layer is the input to the LSTM layer.

The weights for the Embedding layer can either be initialized with random values, or more commonly, they are initialized with third-party word embedding such as word2vec, GloVe or fasttext (or others) and these weights can optionally be fine-tuned during training. Using third

party embedding to build word vectors is as a form of transfer learning, since you transfer the semantic information between words that were learned during the embedding process.

The Bidirectional LSTM

Bidirectional LSTMs focus on the problem of getting the most out of the input sequence by stepping through input time steps in both the forward and backward directions. In practice, this architecture involves duplicating the first recurrent layer in the network so that there are now two layers side-by-side, then providing the input sequence as-is as input to the first layer and providing a reversed copy of the input sequence to the second. This approach was developed some time ago as a general approach for improving the performance of Recurrent Neural Networks (RNNs).

Although Bidirectional LSTMs were developed for speech recognition, the use of Bidirectional input sequences is now a staple of sequence prediction with LSTMs as an approach for lifting model performance.

Dense Layer

It's the most basic layer in neural networks. Dense is the only actual network layer in that model. A Dense layer feeds all outputs from the previous layer to all its neurons, each neuron providing one output to the next layer.

DropOut Layer

Dropout is a regularization technique, which aims to reduce the complexity of the model with the goal to prevent overfitting.

Using “dropout", you randomly deactivate certain units (neurons) in a layer with a certain probability p from a Bernoulli distribution (typically 50%, but this yet another hyperparameter to be tuned). So, if you set half of the activations of a layer to zero, the neural network won’t be able to rely on particular activations in a given feed-forward pass during training. As a consequence, the neural network will learn different, redundant representations; the network can’t rely on the particular neurons and the combination (or interaction) of these to be present. Another nice side effect is that training will be faster.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
word_input_layer (InputLayer (None, 100)               0
_________________________________________________________________
word_embedding_layer (Embedd (None, 100, 100)          1020300
_________________________________________________________________
bidirectional_3 (Bidirection (None, 100, 1000)         2404000
_________________________________________________________________
bidirectional_4 (Bidirection (None, 1000)              6004000
_________________________________________________________________
dense_layer (Dense)          (None, 500)               500500
_________________________________________________________________
dropout_2 (Dropout)          (None, 500)               0
_________________________________________________________________
output_layer (Dense)         (None, 2)                 1002
=================================================================
Total params: 9,929,802
Trainable params: 9,929,802
Non-trainable params: 0
_________________________________________________________________
None
WARNING:tensorflow:From C:\Users\Intel\Anaconda3\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from tenso
rflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 7687 samples, validate on 855 samples
Epoch 1/1
7687/7687 [==============================] - 1716s 223ms/step - loss: 0.7092 - acc: 0.4971 - recall: 0.4971 - precision: 0.4971
```

Convolution Neural Network CNN →

CNNs have wide applications in image and video recognition, recommender systems and natural language processing. CNNs, like neural networks, are made up of neurons with learnable weights and biases. Each neuron receives several inputs, takes a weighted sum over them, pass it through an activation function and responds with an output.

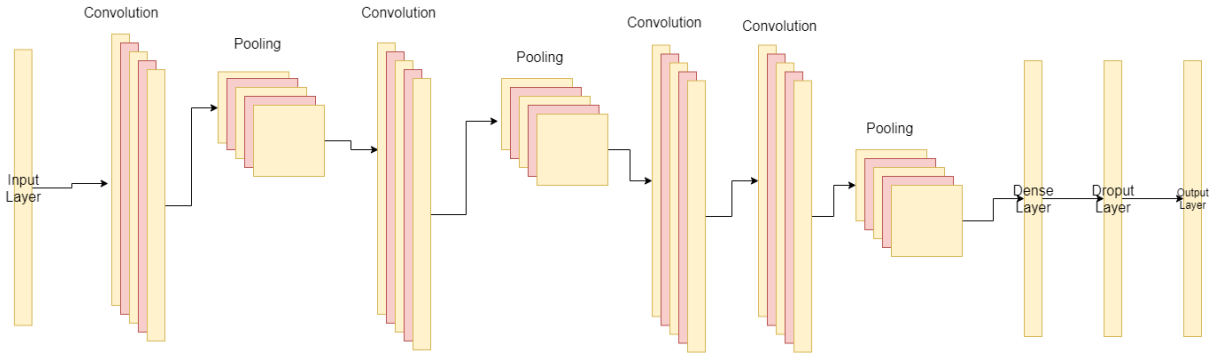


Figure: Architecture of CNN

Convolution is a mathematical operation where we have an input I, and an argument, kernel K to produce an output that expresses how the shape of one is modified by another.

let's explain in terms of an image.

We have an image "x", which is a 2D array of pixels with different color channels(Red,Green and Blue-RGB) and we have a feature detector or kernel "w" then the output we get after applying a mathematical operation is called a feature map.

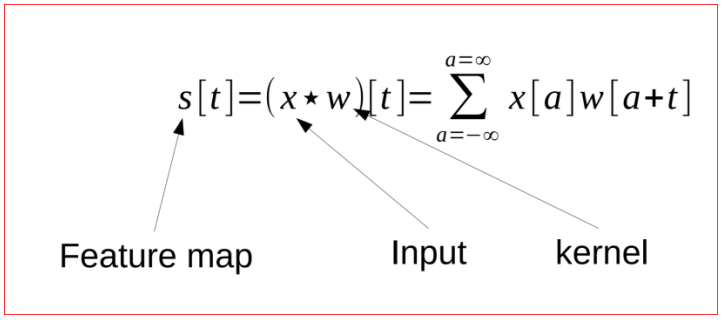


Figure: Convolution function

The mathematical operation helps compute similarity of two signals.

We may have a feature detector or filter for identifying edges in the image, so convolution operation will help us identify the edges in the image when we use such a filter on the image.

We usually assume that convolution functions are zero everywhere but the finite set of points for which we store the values. This means that in practice we can implement the infinite summation as a summation over a finite number of array elements.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

Figure: I is 2D array and K is kernel-Convolution function

Since convolution is commutative, we can rewrite the equation pictured above as shown below. We do this for ease of implementation in Machine Learning, as there is less variation in range of valid values for m and n. This is cross correlation function which most neural networks use.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

Figure: Cross Correlation function

Pooling Layers

A pooling layer is another building block of a CNN.

Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. Pooling layer operates on each feature map independently.

The most common approach used in pooling is max pooling.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
char_input_layer (InputLayer (None, 1024, 92)          0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 1018, 128)         82560
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 203, 128)          0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 199, 128)          82048
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 39, 128)           0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 35, 128)           82048
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 33, 128)           49280
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 6, 128)            0
_________________________________________________________________
flatten_1 (Flatten)          (None, 768)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              787456
_________________________________________________________________
dropout_2 (Dropout)          (None, 1024)              0
_________________________________________________________________
output (Dense)               (None, 2)                 2050
=================================================================
Total params: 1,085,442
Trainable params: 1,085,442
Non-trainable params: 0
_________________________________________________________________
None
Train on 7687 samples, validate on 855 samples
Epoch 1/1
7687/7687 [==============================] - 181s 24ms/step - loss: 0.6939 - acc: 0.4950 - recall: 0.4942 - precision: 0.4962 -
val_loss: 0.6948 - val_acc: 0.4901 - val_recall: 0.4901 - val_precision: 0.4901

Epoch 00001: saving model to cnn_model.hdf5
2848/2848 [==============================] - 28s 10ms/step
```

RNN/LSTM+CNN

The CNN LSTM architecture involves using Convolutional Neural Network (CNN) layers for feature extraction on input data combined with LSTMs to support sequence prediction.

CNN LSTMs were developed for visual time series prediction problems and the application of generating textual descriptions from sequences of images (e.g. videos). Specifically, the problems of:

- Activity Recognition: Generating a textual description of an activity demonstrated in a sequence of images.
- Image Description: Generating a textual description of a single image.
- Video Description: Generating a textual description of a sequence of images.

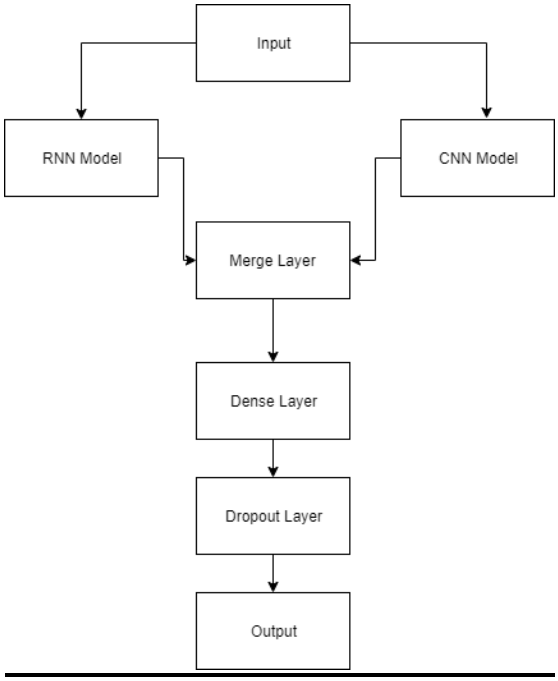
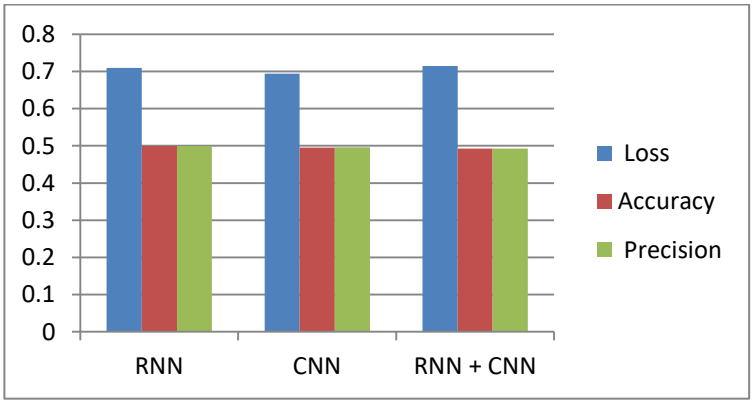
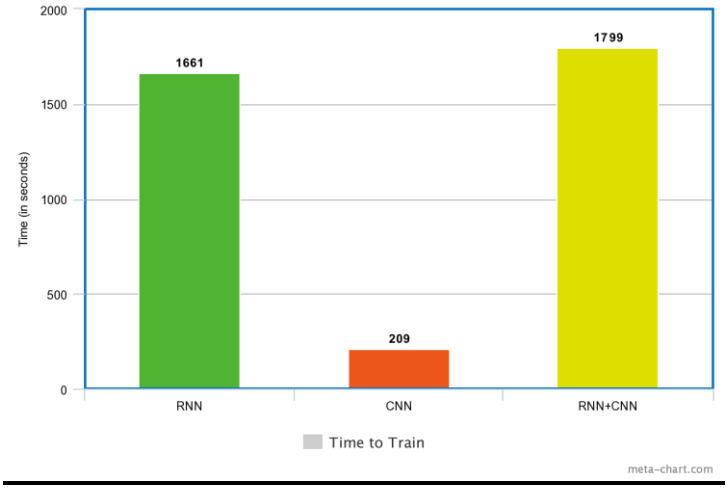


Figure: RNN + CNN Architecture

Graphs





Expected Output

In this project we are going to use the stock market dataset for training the model, sentiment analysis on tweets and risk calculation to overcome the existing barrier and making the riches of stock market investment available for all.

Result

RNN →

```
Stats for RNN model
Examples of correct predictions:

Stock movement was positive
News info:
to defend Alstom power unit buy at EU hearing

Stock movement was positive
News info:
Fitch Affirms Barclays Bank Taipei Branch at 'AA+(twn)'; Withdraws Ratings

Stock movement was positive
News info:
Iran reassures India over development rights of gas field

Examples of INcorrect predictions:

Stock movement was negative
News info:
Taiwan's UMC orders machinery equipment from Tokyo Electron

Stock movement was negative
News info:
Forward Calendar - United States corporate bond new issues

Stock movement was negative
News info:
Apple seeks to dismiss lawsuit filed by electric battery maker
```

```
Top 3 Most Positive Probability:
      Actual  PositiveProb
1559     1.0      0.545421
1585     1.0      0.545650
405      1.0      0.548100
```

CNN →

```
Stats for CNN model
Examples of correct predictions:

Stock movement was positive
News info:
to defend Alstom power unit buy at EU hearing

Stock movement was positive
News info:
Fitch Affirms Barclays Bank Taipei Branch at 'AA+(twn)'; Withdraws Ratings

Stock movement was positive
News info:
Iran reassures India over development rights of gas field

Examples of INcorrect predictions:

Stock movement was negative
News info:
Taiwan's UMC orders machinery equipment from Tokyo Electron

Stock movement was negative
News info:
Forward Calendar - United States corporate bond new issues

Stock movement was negative
News info:
Apple seeks to dismiss lawsuit filed by electric battery maker


Top 3 Most Positive Probability:
      Actual  PositiveProb
1703     0.0      0.576799
2504     0.0      0.585518
699      1.0      0.597684
```

RNN+CNN →

```
Stats for CNN_RNN model
Examples of correct predictions:

Stock movement was positive
News info:
to defend Alstom power unit buy at EU hearing

Stock movement was positive
News info:
Fitch Affirms Barclays Bank Taipei Branch at 'AA+(twn)'; Withdraws Ratings

Stock movement was positive
News info:
Iran reassures India over development rights of gas field

Examples of INcorrect predictions:

Stock movement was negative
News info:
Taiwan's UMC orders machinery equipment from Tokyo Electron

Stock movement was negative
News info:
Forward Calendar - United States corporate bond new issues

Stock movement was negative
News info:
Apple seeks to dismiss lawsuit filed by electric battery maker


Top 3 Most Positive Probability:
      Actual  PositiveProb
187      0.0      0.600560
2529     0.0      0.605467
1559     1.0      0.605598
```

Accuracy

-	Accuracy	Recall	Precision
CNN	0.500000	0.500000	0.500000
RNN	0.498947	0.498947	0.498947
CNN + RNN	0. 498244	0. 498244	0. 498244

Conclusion

We propose a deep learning based formalization for stock price prediction. It is seen that, deep neural network architectures are capable of capturing hidden dynamics and are able to make predictions. Whether you use RNN or CNN or hybrid models for time series forecasting really depends on the data and the problem you try to solve. I would go with a simple model if it serves the purpose and does not risk to overfit.

Reference

[1] https://keras.io/

[2] https://nlp.stanford.edu/projects/glove/

[3] https://gist.github.com/aparrish/2f562e3737544cf29aaf1af30362f469

[4]https://www.researchgate.net/publication/321503983_Stock_price_prediction_using_LSTM_RNN_and_CNN-sliding_window_model

[5] Antonina Kloptchenko, Tomas Eklund, Barbro Back, Jonas Karlsson, Hannu Vanharanta and Ari Visa, Combining Data And Text Mining Techniques For Analyzing Financial Reports: Eighth Americas Conference on Information Systems, 2002, pp. 20-28.

[6] Marc-André Mittermayer and Gerhard F. Knolmayer, Text Mining Systems for Market Response to News: A Survey: Working Paper No 184, Institute of Information Systems, University of Bern, August 2006.

[7] Robert P. Schumaker and Hsinchun Chen, Textual Analysis of Stock Market Prediction Using Financial News Articles: 12th Americas Conference on Information Systems (AMCIS), 2006.

[8] Nan Li and Desheng Dash Wu, Using text mining and sentiment analysis for online forums hotspot detection and forecast: Decision Support Systems 48, 2010, pp. 354–368.

[9] Xiangyu Tang and Chunyu Yang, Jie Zhou Stock Price Forecasting by Combining News Mining and Time Series Analysis: 2009 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology – Workshops, ISSN 978-0-7695-3801-3/09, pp. 279-282.