**Project Proposal**

**Deep Learning Big Project (50.039)**

1. **Topic, problem to be investigated**
   1. **Stock Returns Prediction**: Every year, millions of investors, traders and market participants invest into assets in the stock market, with the hope of generating profits. The idea is simple: they believe that they can leverage their understanding of current affairs, along with lots of data, in order to predict whether a particular asset will go up or down in the future, and thus make investment decisions today that they expect will provide them with profits.
   2. **Components of Stock Returns**: In reality, there are many factors that affect stock prices, such as economic factors, geopolitical conditions, daily news (about each company), interest rates and more. But the component we are most interested in, is the technical component: prices, volume of stock traded, moving averages, highest price in a day, lowest price in a day, etc. The idea is that there is a component of stock price that can be analyzed technically (purely in terms of numerical data). As mentioned above, these technical factors cannot account for the entirety of the stock price, however we are not trying to build a model that can explain the entirety of the stock price: we merely require the model to learn just enough, so that it can gain an edge that is sufficient, so that if an amount of money is invested into the trading strategy (simulated), it will generate overall positive returns over time.
   3. **Why Deep Learning:** Deep Learning has been successfully applied to many difficult problems. We know that deep learning is more effective in situations where we have lots of data and are tackling a complex problem. We have also conducted a literature review that suggests that certain deep learning architectures are able to extract out predictive value from technical indicators, and thus generate positive returns on investment[1][2][3].
   4. **Why CNN:** Although it seems counterintuitive to use CNNs, since stock prices are generally lists of numbers, if we generate rows of data that include data from multiple days, for multiple technical indicators, we will end up with a 2D representation of a single datapoint – we can treat this as an image, and use convolutional layers that will extract out time-dependent features as well. If we choose to extend the above data (different technical indicators, for different days) for different assets (lets say we get the data for different companies in a particular industry), we can obtain 3D datapoints and use 3D convolutional layers.
2. **Expected Inputs and Outputs**

**Input**: 1 Datapoint: multiple 2D arrays (refer to the model architecture in Section 4, part C), each having data for different time-lagged features, for different days. For example: a 3x3 2D array containing the 3 day price moving average, 5 day price moving average, and 7 day price moving average, for the past 3 days.

**Output**: 0 indicates the stock price will drop next day,

1 indicates the stock price will rise next day

1. **What Dataset to be Used**

We will use the stock market data that is published on Yahoo Finance. The dataset link for obtaining the S&P500 Futures Data (the first index which we will train and test our architecture on) is: **https://finance.yahoo.com/quote/ES%3DF?p=ES%3DF**

Yahoo Finance API is the API that Yahoo provides to fetch financial information, it is a media platform that provides financial news, data about stock quotes, press releases, and financial reports. We can access to more than five years of daily OHLC price data and get minutes OHLC data for recent days with Yahoo Finance.

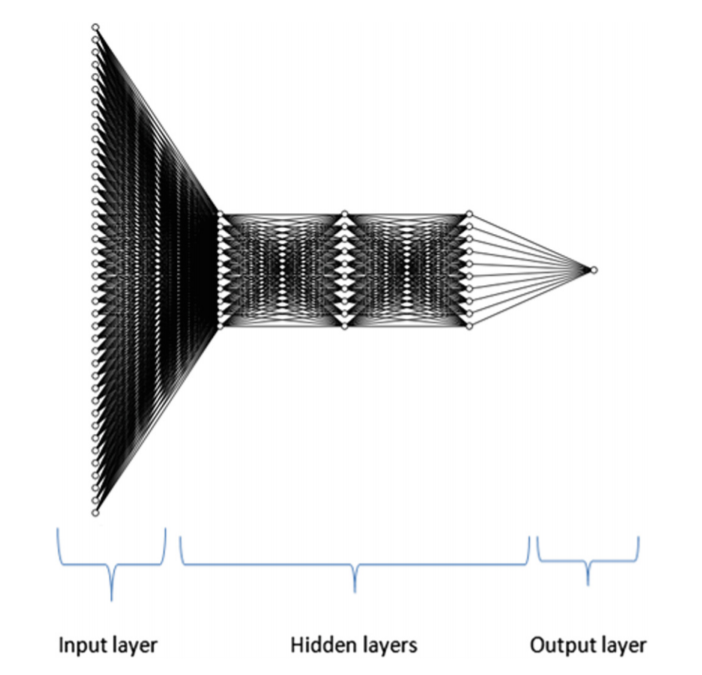
1. **Architecture Draft**

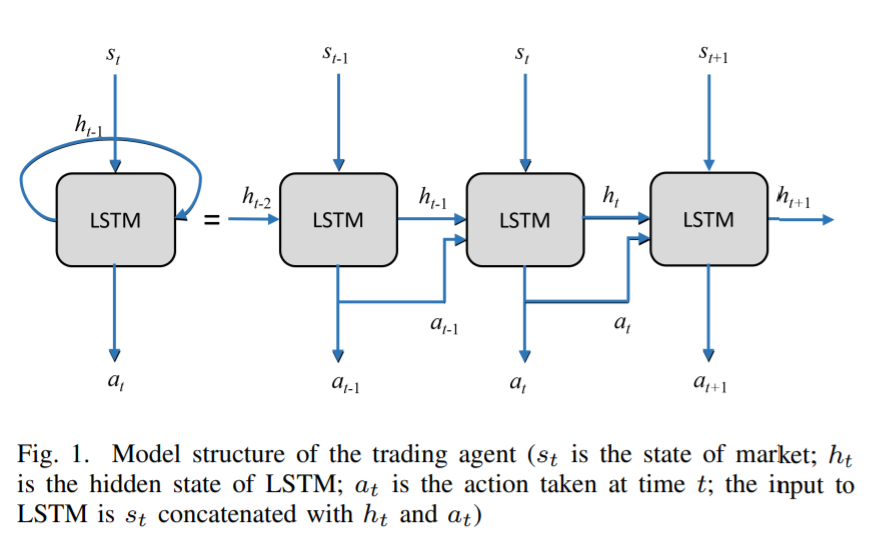
In order to understand how we might utilize CNNs for multiple time series data, we conducted a literature review on the utility of CNNs in this field, to understand current approaches and gather some state-of-the-art results for comparison purposes.

One criteria / high-level intuition that we kept in mind as we evaluated the different approaches, was that the approach should not be completely black-boxed, and should make sense prior to training the model, in order to avoid over-fitting. This will help us avoid spurious correlations.

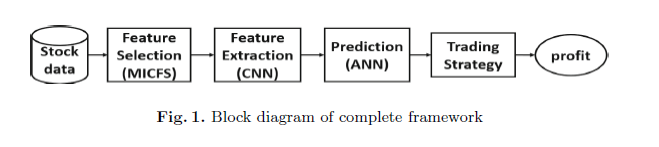
We list three papers we read, and we will start our project by using a rough architecture that takes inspiration from paper [3] which we feel makes the most sense, and we will expand on it further through further feature engineering (utilizing principal components), changing the number of convolutional layers, adjusting the number of days of data for each datapoint, hyper-parameter tuning, etc.

1. **A Stock Market Trading System using Deep Neural Network** [1]: In this paper they train a Neural Network on a dataset, that uses the opening, closing, minimum and maximum prices for the past 10 days(4x10 = 40 input features), for STI(the Straits Times Index). They train multiple such models, changing the output variable to the price of the n+tth day. They then create a rules based trading system based on the output of the multiple models and the last available price. They achieve a 70% profitable trade ratio, and a Sharpe ratio of 5.34.
   1. Evalution: This is a good precedent, but it is mostly black-boxed. The rules based system makes sense, and can be adapted for our project, but there’s no feature engineering here.
   2. Architecture used:

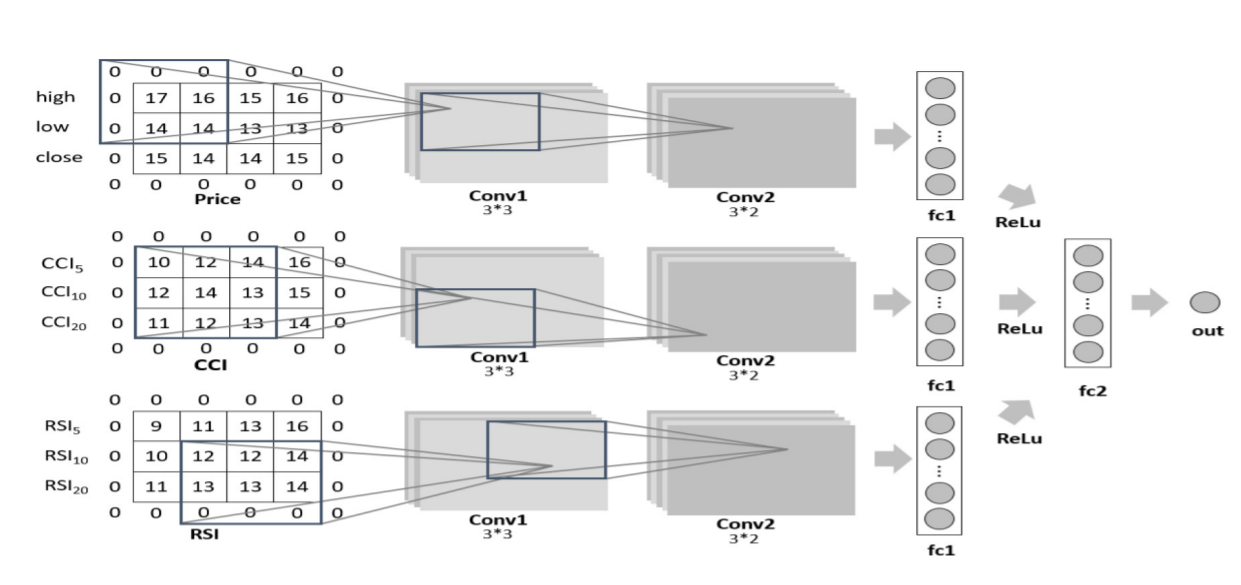


1. **Quantitative Trading on Stock Market based on Deep Reinforcement Learning** [2]: In this paper, the authors use deep reinforcement learning, use input data consisting of multiple technical indicators along with noisy price data for 3 assets, and directly predict trading decisions(states here). They achieve a good return rate of 1.8-2% on average, achieving profits on two assets, but suffering a huge drawdown on the third.
   1. Evaluation: They use 18 different technical indicators, which is good. Reinforcement learning is a method that intuitively makes sense, but it is not very transparent in its decision making.
   2. Architecture used:
   3. 
2. **(our favourite precedent)** **A Multi-Indicator Feature Selection for CNN-Driven Stock Prediction** [3]: Here the authors identify the biggest problem with using CNNs on time series technical indicators – theres a lot of overlap, lot of correlation between these indicators. Thus there is a need for a method that can extract out the unique variations in the input dataset, before using them as input data for the CNNs. Thus they develop a Maximal information Coefficient based feature selection mechanism, to determine which features would contribute the most to the binary output(positive or negative returns). They then train a unique CNN architecture – using separate convolutional filter pipelines for each set of features (to ensure no mixing between the features until the very last layer). They also use L2 regularization in order to prevent overfitting. They achieve positive returns over 2016-end of 2017, ranging between 27 and 36%, for 8 different indexes, showing the robustness of their method.
   1. Evaluation: Very logical approach, supported by feature selection, and time series convolutions on the features. They utilize a rules-based trading strategy that relies on the signals output by 2 models. Perhaps the rules system can be expanded further by training more models, like paper [1].

System Architecture:



Model Architecture:



1. **Team Members**

* Rahul Bhattacharjee, 1003719
* Chen Yan, 1003620
* Pang Luying, 1003631
* Gladys Chua Shi Ying, 1003585

1. **What we will deliver**

* Dataset and Dataloaders
* Trained model for stock prediction
* Visualization of the dataset and model’s performance
* a small GUI runnable on user-selected data (at least a few examples)
* A Comprehensive Report including:
  + Observations
  + Accuracy Metrics (F1, Accuracy, ROC AUC, Training Loss, Validation Loss, etc)
  + Model Architectures Used
  + System Architecture

**References / Citations**

**[1]** Yong, B.X.; Rahim, M.R.A.; Abdullah, A.S. A stock market trading system using deep neural network. In Proceedings of the Asian Simulation Conference, Melaka, Malaysia, 27–29 August 2017; Springer: Berlin/Heidelberg, Germany, 2017.

**[2]** Wu, J.; Wang, C.; Xiong, L.; Sun, H. Quantitative trading on stock market based on deep reinforcement learning. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019

**[3]** Yang, H.; Zhu, Y.; Huang, Q. A multi-indicator feature selection for CNN-driven stock index prediction. In Proceedings of the International Conference on Neural Information Processing, Siem Reap, Cambodia, 13–16 December 2018; Springer: Berlin/Heidelberg, Germany, 2018.