CNN-DDQN based, Self Learning, Automated, Trading System

MSc Research Project
MSc. Data Analytics

Rushabh Parmar
Student ID: X21187240

School of Computing
National College of Ireland

Supervisor: Dr. Ahmed Makki

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Rushabh Parmar
Student ID:	X21187240
Programme:	MSc. Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Dr. Ahmed Makki
Submission Due Date:	14/12/2018
Project Title:	CNN-DDQN based, Self Learning, Automated, Trading System
Word Count:	7952
Page Count:	25

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard speci ed in the report template. To use other author s written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	14th August 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not su cient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator o ce must be placed into the assignment box located outside the o ce.

O ce Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

CNN-DDQN based, Self Learning, Automated, Trading System

Rushabh Parmar
X21187240

Abstract

Trading is a widely studied field in terms of psychology as well as technology. While there are ample trading systems that provide an automated trading environment based on calculative trends, there are still minimal algorithms that capture the market sentiments to bet their decisions. Recent advancements in Convolutional Neural Networks (CNN) have helped us understand the important features of a candlestick chart and capture the market trends to output a trading decision. At the same time, Deep Reinforcement Learning algorithms like Double Deep Q-Network (DDQN) have proven successful in creating a self-learning trading system. This research demonstrates a CNN-DDQN-based model, where CNN outputs a trading decision given 24-day Gramian Angular Field (GAF) images as input, acting as an agent in DDQN, which incorporates the trading environment. The GAF is what is being tested in the research with a motive to enhance the performance of the CNN-DDQN model to make better trading decisions. CNN helps us acknowledge the psychological perspective of the market, whereas we fine-tune the decisions made over time by using the Q-Network at the core. The GAF image, with 5 channels stacked vertically as features, outperforms all the other data representations with the highest weighted F1 score of 0.813, by CNN to classify the images correctly.

1 Introduction

Trading has been one of the most researched fields, where technology has always acted as a helping hand to understand market sentiments. Short-term trading is usually based on technical analysis, comprising of visually comprehending the charts to capture a trend. Still, there have been ample theories that repel the idea of understanding the pattern any asset reveals. Eugene Fama in his journal 'Efficient Capital Markets: A Review of Theory and Empirical Work' puts forward his theory on the Efficient Market, now renowned as 'The Efficient Market Hypothesis (EMH), stating that the price of any asset at any point of time only reflects the current information about that asset, showing no evidence of the historic prices. Technical analysis does little help in predicting the price, as markets are 'Information Efficient'[1]. As with every revolution comes counterrevolution, numerous economists, econometrics, and psychologists presented their critics of the theory, naming that behavioral science as a whole, when interpreted correctly does hold the power to understand the relation between the price-action movement and supply-demand balance, opening the door for the prediction of stock prices. On the other hand, Burton G. Malkiel in his paper 'The Efficient Market Hypothesis and its Critis' challenges the validity of EMH, he presented the works explaining the other way around, he states that the prices of any asset, if not fully then are partially predictable and with right tools and information, one can overcome the Efficiency Market Trait [2]. Considering this, many algorithms have been developed to predict future prices, or direction, whereas following Machine Learning, Deep Learning has seen its fame in absorbing huge amounts of historic data, only to reveal patterns, a human eye would miss.

Convolutional Neural Networks (CNN), part of the Neural Networks family are well known for image processing, where they capture the important features of an image and perform various operations to either output a scalar value, as part of regression, classify the image, as part of classification or give a multi-class probability as an output to further process the decision, based on its application. A 2-D CNN takes an input as a 2-D matrix, which can be an RGB or Grayscale image, and image processing is done in a hierarchical manner, where it learns the low-level features, which are edges and lines, and further, it learns high-level features as shapes and objects. CNN acts as a powerful tool in learning different types of features of an image, and further classifying them.

Whereas Gramian Angular Field (GAF) is an image encoding technique, where univariate time-series data is transformed into a Gram Matrix and represented as an image, to store the temporal correlation between the data points. Open-High-Low-Close (OHLC), which is profoundly used to understand the demand-supply balance of the market, reveals the market sentiments of that current time, and the possibility of short-term profit space exploration. There are various ways to plot the OHLC and one of the most famous charts is the candlestick chart, invented in the 18th century by a rice trader [RIC Brief 4]. These are box plots arranged in a sequential manner, where OHLC, all the data points are comprised in the figure for that particular time-stamp. While candlestick patterns have been a well-formed representation of the time-series data, GAF images are known to capture the temporal correlation between the data points at different time intervals. GAF tends to store the particular time series in the primary diagonal, as the time increases as we go from top-left to bottom-right, which makes it easier for CNN to capture the seasonality and patterns in the data. Jun-Hao Chen et al. [5] presented a study where they used GAF with CNN for classifying eight critical candlestick images, the GAF-CNN model provided a 90.7 % success rate in classifying the patterns, proving its applicability in capturing the vital features of a patter. Whereas the study did not express its motivation to predict the future price-action movement based on the patterns, which drives the inspiration behind our objective to use GAF along with CNN to help contribute to the vast field of stock price prediction.

Deep Reinforcement Learning (DRL) has been quite equipped to show its performance in various fields like robotics, aviation, gas and energy, and many more, it is used to simulate an environment and introduce an agent to take automatic actions based on a policy function, the actions taken on a particular state forms a state-action pair, whose potential is optimized using the loss and reward function. Andrew Brim et al. (2022) [7], operated an experiment where they used a DRL algorithm, Double Deep Q-Network (DDQN), with CNN as a function approximator to trade the top 30 stocks of the S&P 500 index, the resulting CNN-DDQN model outperformed the index for Jan 1st, 2020 till June 31st, 2020, with 17.2 %, also capturing the ability of CNN to shift its attention from all features to the most recent ones. This proves the privilege that comes with using the DRL with CNN, to optimize and simulate the trading environment to better the decision-making. The images used as input were candlestick patterns for 28 days, which leaves space to question the performance of the model on GAF images, which capture the temporal dependency at its most. This acts as a backbone for our research to test the performance of the CNN-DDQN model with GAF images, on top-10 Nifty-50 stocks, where the model will be evaluated based on weighted F1 score, attained by the trained CNN models.

2 Related Work

Literature Review: We have divided the literature review into three subsections for creating a broader picture of the background and enhancing the flow of our research. The first section focuses on the need to choose the area of research as trading, it highlights the Captial Market Theory and further critics on, second section dives into the use of Gramian Angular Summation Field (GASF) images for better representation of time-series data and applicability of Convolutional Neural Networks (CNN) to process the images, lastly, we shed some light on advancement of Reinforcement Learning (RL), Deep Reinforcement Learning (DRL) and its merger with CNN to direct the way Deep Learning models are used for trading.

Finance

Information about an asset, plays a vital role in deciding its face value, and so of the share price. Even though the information is cheap, partly because of companies' fundamental disclosure and partly because of technology, it doesn't reach every trader as a whole but in pieces at varied times. Following this, Eugene Fama's most renowned theory about capital markets 'The Efficient Market Hypothesis' came into publicity stating that the Markets are Efficient, meaning the price of any asset reflects the first available information and not the other way around while predicting the next price-action movement is nearly impossible as the future information is still unknown (Eugene Famas, 1970). It states that there are three types of markets, weak-form efficient market, where predicting the future returns based on past returns of a stock is not feasible, and none of the technical indicators adds any value to this, then is a semi-strong-form efficient market, here the publicly available information of the company reflects the price of the stock and it requires rigorous fundamental analysis to find the relation, while lastly, we have a strong-form efficient market, where the price of the stock depends upon the public as well as private information of the company, thus making this form very rare, as the private information is only accessible to the few. As the markets' volatility depends upon the Information Trait, any randomly selected portfolio would give as much return as a carefully selected one by professionals, as the news is unpredictable, so are the returns. Even though the theory formed a strong statement to support Markets Efficiency, it was attacked by numerous psychologists and economists with a strong interest in understanding the investor's behavior to predict the price movement.

In April 2003, economist Burton G. Malkiel released a study titled 'The Efficient Market Hypothesis and its Critics,' aiming to examine the credibility of the 'Efficient Market Hypothesis', to enlighten the idea of using behavior science to predict the market, and if not fully then partially.

Every revolution is followed by its counterrevolution, and so did 'The Efficient Market Hypothesis theory. Burton F. Malkiel in his work, 'The Efficient Market Hypothesis and its Critics' has presented previous empirical studies that prove the predictability trait of an asset's price. He puts weight on the applicability of behavior science to understand market sentiments for taking appropriate decisions, as the saying goes 'History doesn't repeat itself, but rhymes', the patterns don't repeat themselves, but there is always some clue about the future in them if one pays enough attention. The study provides a section where various forms of predictability patterns were examined to estimate their after-effects (Burton F. Malkiel, 2003). It shows that past dividend yields, reveal a 40% variance in future returns, stating that the participants behave a certain way after noticing the dividends paid by the company, forming a pattern. Also, the October 1987 crash was mentioned, stating that there were signs of future huge declines, in news and sentiments, which started affecting the market in the first few weeks of October, followed by the mass losing its confidence, causing a market crash. It happened in bits and pieces, then all at once, only if one knows how to understand the balance between the demand and supply, one would be able to partially estimate a profit space. Sentiments are subjective, but their relation and effect on the price-action movement is not, as every category of participant would react in its way to a particular information, Dong Lou in "Term Structure of Sentiment Effect on Investor Trading Behavior" has categorized those participants in three types, explained the term structure of their sentiment and its effect on the trading behavior and how it stands as an important factor in decision making.

In the paper "Term structure of sentiment effect on investor trading behavior," it is noted that the stock market consists mainly of three types of investors: Domestic individuals, Domestic institutions, and Foreign investors. The study reveals that the duration of an investor's sentiment towards the market, known as the "term structure," varies depending on the type of investor (Kim Karam et. al, 2021). In the short term, all categories of investors show changes in their trading behavior influenced by market sentiment. However, the impact of sentiment on each group differs. As the study suggests, Domestic Individual Investors are highly influenced by sentiment, both in the short term and the long run. This means that their trading behavior is strongly affected by emotions and market sentiment, and this effect can last for an extended period, while Foreign Investors on the other hand are less influenced by market sentiment compared to domestic individual investors. The effect of sentiment on their trading behavior diminishes over the long run, suggesting that they rely more on other factors when making investment decisions. Lastly, the Domestic Institutions are the least sentimental group and are likely the most informed investors in the short term. They show the smallest reaction to changes in sentiment (ΔSent) when using data on a one-day interval. This implies that they are less affected by emotions and sentiment, and they may have access to better information, making them more informed investors. Overall, all types of investors are affected by market sentiment in the short term, which is reflected in their trading behavior.

This finding presents opportunities for traders to aim for short-term gains by predicting market sentiments and the resulting trading patterns.
One common method used to understand market sentiments and predict future movements is by observing the open-high-low-close price of a stock over a specific period. As these data points collectively reflect the change in the balance between supply and demand, and thus the pattern.

Convolutional Neural Network:

Gramian Angular Field (GAF), is a type of univariate time series encoding technique, used to represent the time-series data into images. The time-series data, represented in cartesian coordinates doesn't capture the correlation of each data point with one another, but only with time-stamps, thus a need for spatial representation of the data, which captures the temporal correlation arises. Wang and Oates, 2015, contribute a study, where two types of representation of data namely GAF and Markov Transition Field (MTF), along with their combination are tested onto a Tiled-Convolution Neural Network. A Tiled-CNN is a Deep Learning architecture, belonging to the family of CNN, which is designed for tile-like structured data which are in sequential, graph, or any other structured form. Tiled CNN, captures the high-level features of every image of the 12 datasets used and the model is evaluated based on error rate while using different representations of the data. The experiment presented highly competitive results compared to the state-of-the-art techniques, the GAF images, have less error rate for 8/12 datasets compared to MTF. The study overall concludes that Tiled CNN performs better when Gramian Matrix representing the temporal dependency is given as an input. (Write more about, on what basis were the models evaluated). Such contributions have led many researchers to use the GAF representation of time-series data in the world of finance. Further to support the assumptions, we brief on a GAF-CNN model introduced by Jun-Hao Chen et al. 2020, to classify candlestick patterns.

Convolutional Neural Networks have been highly profound in the field of finance for their predictability power, in terms of 1-D, 2-D as well as 3-D data. CNN can learn simple features like lines, edges, and corners to complex features like objects and shapes, being provided the right set of architecture along with the right representation of data in use. Trading decisions are mostly based on price charts, and the most commonly used is a candlestick chart, which elaborates the demand-supply balance, comprising the Open-High_low-Close data of each time-stamp in boxplots. The candlestick charts have several repetitive patterns, which help traders estimate a smooth entry or exit space. There are overall 103 candlestick patterns according to Thomas Bulkowski's Encyclopedia of Candlestick Charts, and the research by Jun-Hao Chen et al. (2020), helps classify eight of the most critical ones using a GAF-CNN model. This model encodes the time-series data of stock price into GAF images, they create two different datasets, based on features comprised, of Open-High-Low-Close (OHLC) and Closing, Upper shadow, Lower shadow, Real Body (CURL), these two features are compared against each other, where CURL has shown to outperform the former one, the GAF-CNN model was also compared with LSTM, on the real-world as well as simulated data, where it outperformed LSTM with 90.7 % accuracy on real-world as well as 92.4% accuracy on simulated data. This explains that GAF-CNN combined model is an excellent choice for trading decisions support, further directing us toward learning about the different architectures of CNN in detail.

Deep Learning has caught the eyes of many businesses and researchers recently while competing as well as beating human performance in some well know tasks. While there are colossal numbers of papers that shed some light on the recent advancement, Laith Alzubaid et al. (2021) contributes to the field by publishing in-depth research on Deep Learning algorithm, Convolutional Neural Networks' evolution, architecture, challenges, and applications. The paper talks about each layer in detail, from the Convolutional layer to the Fully Connected Layer, where each activation function such as ReLU, Leaky ReLU, Tanh, sigmoid, and more are explained with their usability. It explains the types of loss functions and required optimizers to reduce the loss during the training phase. Further numerous architectures are explained, from AlexNet, GoogleNet, Xception, HRNet, CapsuleNet, and many more, with their modifications and use cases.

Reinforcement Learning

Neural Network has been outperforming all the traditional machine learning methods, in countless applications, mostly because of its ability to comprehend and process enormous amounts of data. While the results are promising, the black box nature of the model acts as a limitation, where the understanding of the hidden layer, which maps the input to the output is still not stated. Andrew Brim et al. 2022, undertake the challenge to explore the ability of the Convolutional Neural Networks (CNN) Feature Map Visualization technique with an aim to interpret the working of the model. In the research, they experiment with the Deep Reinforcement Learning algorithm, Double Deep Q-Networks' (DDQN) adaptability with CNN to outperform the S&P 500 index, as well as capture the shift of neuron excitement from all the features to the most recent ones during the covid phase. The shift in neurons' effect on the model's ability to make better decisions was captured using the feature map visualizations and measured using logistic regression, while for the CNN-DDQN models returns were compared directly for t-consistency to that of the S&P 500 index, and results were in the favor, showing a 13.2 % geometric returns in 124 days compared to -4 % by S&P 500 index.

3 Research Methodology

The research follows a CRISP-DM methodology, which is well known for its iterative approach that helps us reassure the fulfillment of the business objective. The method starts with clearly defining the business objectives, and as we proceed further it gives us the opportunity to drive back, and sense if the steps are in-line with the motive. The methodology comprises of five main steps, which are discussed in detail below:

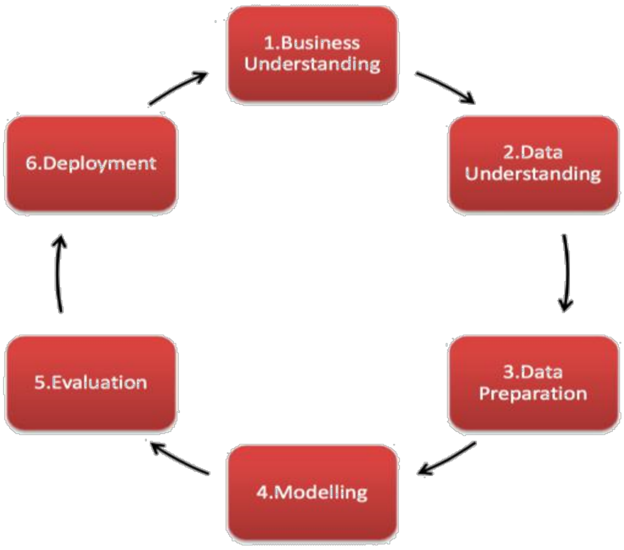

Fig 1. CRISP-DM cycle

1. Business Understanding:

Recent advancements in the trading model using Machine Learning and Deep Learning have proven their success, but they're still lying colossal opportunities to bridge the gap between finance and technology, especially when one chooses to use algorithms to trade based on market psychology. We use a Convolutional Neural Network-based Double Deep Q-Network model to estimate a trading opportunity, by understanding the patterns revealed by the historic prices. The research motives to contribute to the academic research on using Deep Reinforcement Learning along with other Deep Neural Networks, to self-learn the trading decisions. While the study also aims at putting forth a baseline model, which can be used by industrial experts as an assistant or an add-on to their current algorithms. The model will be trained and tested on Nifty-50 top-10 stocks, from the year 2016 January to 2021 December.

2. Data Understanding:

Data acts as a crucial element in terms of completing our objective, and it is extremely essential to verify the quantity and assure the quality of the data extracted. Two steps involved in this section, Data Extraction, and Exploratory Data Analysis are explained below:

2.1 Data Extraction:

The dataset we will be using for this experiment comprises share prices for the top 10 companies of Nifty 50, which is India's National Stock Exchange (NSE) Index. The companies are, Reliance, TCS, Infosys, Hindustan Unilever, ICICI Bank, HDFC Bank, Bharti Airtel, Kotak Bank, Wipro, LT, and the historic data involves daily movement of their stock prices from the date 01-01-2016 till 31-12-2021. We extract the data

from Yahoo Finance, using its Python Library 'finance', the library uses Yahoo Finance API internally to extract and process the request. The data is extracted in a Pandas data frame format.
As shown below in Fig 2, (Try to add labels of the company), the Dataframe has Dated as an index, and seven columns namely, Open, High, Low, Close, Adjacent Close, Volume, and Tickers. When extracted through the finance library, it comes as an individual ticker, which we merge and adds another column as 'Tickers' for our better interpretation.

```
stock_data
```

Date	Open	High	Low	Close	Adj Close	Volume
2016-01-01	500.158997	504.666260	499.366516	502.907928	483.540588	2499742
2016-01-04	497.781525	502.140198	488.717438	492.977081	473.992157	13923887
2016-01-05	495.453583	500.258057	493.819092	497.855835	478.683044	6897687
2016-01-06	499.069336	514.324707	495.503113	511.253815	491.565033	12349673
2016-01-07	505.731171	509.173553	499.292206	501.867798	482.540497	9109980
...	...	...	...	...	...	...
2021-12-24	697.750000	702.400024	690.150024	698.450012	689.858337	6591213
2021-12-27	699.849976	701.000000	691.150024	699.599976	690.994141	5094328
2021-12-28	703.000000	707.500000	701.549988	705.450012	696.772217	5079022
2021-12-29	705.250000	710.000000	700.049988	703.700012	695.043762	3949208
2021-12-30	705.099976	719.900024	704.000000	714.099976	705.315735	9043675

13329 rows × 6 columns

Fig 2 Pandas DataFrame

2.2 Exploratory Data Analysis:

Exploratory Data Analysis (EDA) stands as an essential step in data understanding, giving us the opportunity to explore our dataset in all aspects. We will start by analyzing the high-level information and further explore the individual features in detail.
Fig 3 below shows the number of rows in each data frame when they were extracted from the library. As it is a common fact, every stock is open for trading, and every day the market is open, unless the company is about to be delisted, thus we have 1481 rows for each Ticker.

Ticker	row_count
RELIANCE.NS	1481
TCS.NS	1481
INFY.NS	1481
HINDUNILVR.NS	1481
ICICIBANK.NS	1481
HDFCBANK.NS	1481
BHARTIARTL.NS	1481
KOTAKBANK.NS	1481
WIPRO.NS	1481

Fig 3 Row Count

```
stock_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 13329 entries, 2016-01-01 to 2021-12-30
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       13329 non-null  float64
 1   High       13329 non-null  float64
 2   Low        13329 non-null  float64
 3   Close      13329 non-null  float64
 4   Adj Close  13329 non-null  float64
 5   Volume     13329 non-null  int64
dtypes: float64(5), int64(1)
memory usage: 728.9 KB
```

Fig 4 Data Frame Info

While Fig 4 shows the cumulative data frame information, as shown in the final data frame after concatenation. There are no null values, the 'Volume' column has all the values as integers, the 'Tickers' has string values, while all the other column has float values. Whereas the overall number of rows is 13329. Below is Fig 5, which portrays the trend of each stock, represented using its closing price. As can be observed, there is no seasonality in any stock, and this can be explained as because of volatility and numerous external factors which affect the price at any moment. As can be seen, TCS tends to be the most volatile stock, followed by Reliance, while Wipro seems to be the most stable stock, followed by ICICI Bank. We have our set of high volatile and low volatile stocks, as it will help train the model in a much broader environment.

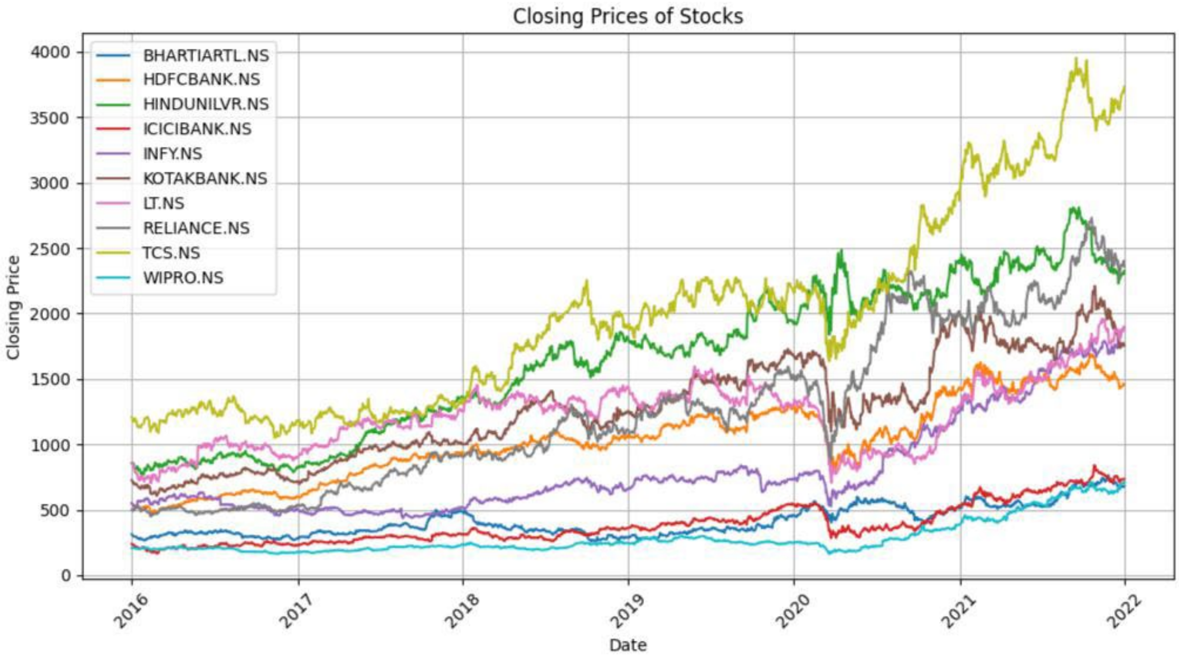

Fig 5 Trend of price-action

Next we check for the spread of data in terms of central tendency and outliers. We plot the boxplot of these 10 stocks below. Fig 6 below shows the boxplot.

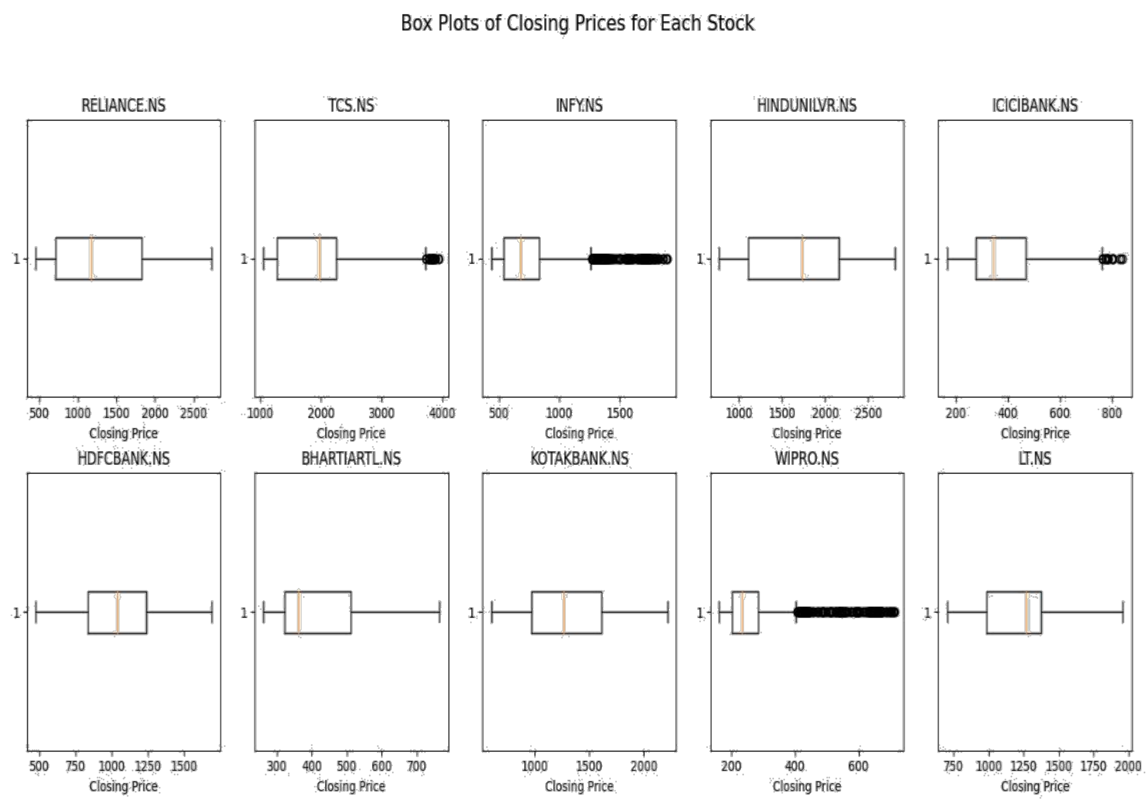

Fig 5 Boxplot of each stock

As seen in the above figure, we can say that the stocks Wipro and Infosys, tend to have many outliers, expressing many of such events where prices were highly volatile, whereas Reliance, HindUniliver, ICIC Bank, and Kotak Bank are among the stocks with least outliers, and where the data is distributed around the median in a balanced way.

3. Data Preparation:

After analyzing the data, and understanding its all dimensions, in every needed aspect, we move forward to data preparation. This stage enhances our practices to create the final data needed for our model. It has 2 substeps as Data preprocessing and Data transformation.

In Data preprocessing, we first clean the data, in this part, we check for null values using the ISNA () function of Python. As seen in Fig 7 below, there are no null values. We further move to data selection where we select the columns we need for our model training.

```
stock_data.isna().sum()
✓ 0.0s

Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
Ticker       0
dtype: int64
```

Fig 7 Null Values in Data Frame

In Data selection, initially we use the columns Open, High, Low, and Close, and drop the columns for Volume and Adjacent Volume, the column Ticker will be used to segregate the different data for different companies.

After normalization, we perform Gramain Angular Difference Field transformation on the data. Gramian Angular Field is a way to represent the data points in a Gram Matrix, which encapsulates the temporal correlation of the time series. These matrices are converted to images for giving them as input to our models

4. Data Modeling:

Data Modeling stands as an essential process, where we choose the models which we will be using to achieve our objective. The models are usually chosen based on previous research done in similar or different domains, applying the models. As we are using a visual representation of the time series data using GASF images, Convolutional Neural Network lies as a powerful model for interpreting images. Jun Hao Chen et al. 2020, explored the potential of CNN with LeNet architecture, with GASF images to identify eight critical candlestick patterns, where they achieved an accuracy of 90.7 percent in identifying eight candlestick patterns. Further Andrew Brim et al. 2022, used CNN along with the Deep Reinforcement Learning algorithm, Double Deep Q-Network (DDQN), on candlestick images, to trade. The CNN-DDQN models' returns were compared with that of the S&P 500 index, and the results were positive. We in this research choose CNN along with DDQN and use combined GASF images, which are constructed by overlapping the Open, High, Low, and Close GASF images. We simulate the trading environment using the DDQN, where CNN acts as a function approximator to optimize the decision-making.

5. Evaluation:

Evaluation is a crucial step, as it helps us understand the performance of the model and the areas in which it needs improvement. The CNN will be evaluated based on the F-1 score, to measure biases in decision-making, in terms of Buy, Sell, and Hold. We use the weighted F1 score, which considers the imbalances in classes, which in our case is the Hold class with the highest amount of data. The model labels the images based on the Bollinger Bands (calc. using mean + std_deviation*1.5), thus labeling a huge number of images as Hold, as it is safer to not trade, as most of the time price is in the bands' range.

The data is being split into Train, Valid, and Test kinds, where train comprises data ranging from 02-01-2019 to 31-12-2019, Validation from 02-01-2020 to 31-12-2021, and test from 02-01-2021 to 31-12-2022. Where the models trained on the train set are saved, validated on the Validation set to tune the parameters, and tested on the Test set to generate the results.

4 Design Specification

4.1 CNN model:

Convolutional Neural Network (CNN) acts as a replica of the visual cortex in humans, where the main activities involve feature extraction and interpretation of the features, meaning referencing it to something known. These functions are operated by a combination of layers when it comes to CNN. The layers are namely the Convolutional layer, Pooling layer, and Fully Connected layer.

The convolutional layer is the most important layer, which contains filters(kernels). The kernel is a grid of a pre-defined shape, with random numbers as values, which act as weights. These are used to perform convolution on the input data.

The convolutional layer accepts the image, which is nothing but a matrix of pixels, of a shape as height X weight, on which we perform inner multiplication (so-called convolve), with the kernel of the specified size and we have multiple filters, where each filter has different weights to capture a different set of features. Thus the output of the Conv layer has as many feature maps, which is depth as the number of filters, and the height and width is calculated using the below formula:

Filter size: F (both height and width of the filter)

Padding: P

Stride: S

$H\_out = (H\_in - F + 2P) / S + 1$

$W\_out = (W\_in - F + 2P) / S + 1$

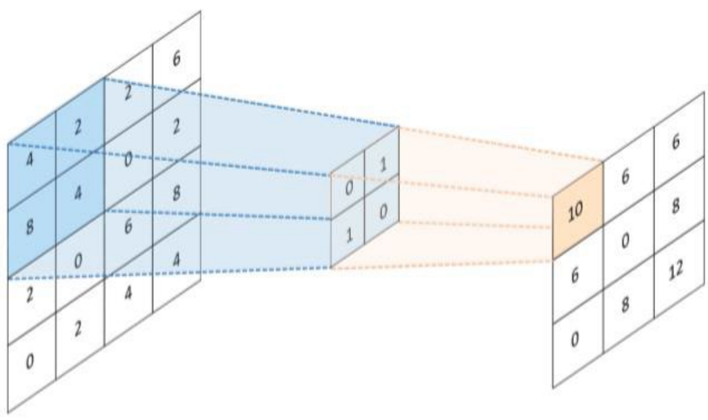

Fig 8 Convolve Operation

Fig 8 above represents the convolution applied on 4X4 image, by a 2X2 kernel.

Padding and stride are two terms used above, where padding enables adding extra pixels to the border of the feature maps, to have the same spatial dimension of the output as the input, while stride represents the movement of the kernel of the image, in horizontal and vertical direction.
We have activation functions applied over the feature maps, to enhance the non-linearity in the feature values captured.
Next, we have a pooling layer, the main purpose of the pooling layer is the sub-sampling of the feature maps. There are different types of pooling layers used for different applications, such as Max Pooling, Min Pooling, Global Average Pooling, Tree Pooling, and Global Max Pooling. Whereas Max Pooling, Average Pooling, and Min Pooling are the most commonly used pooling techniques.

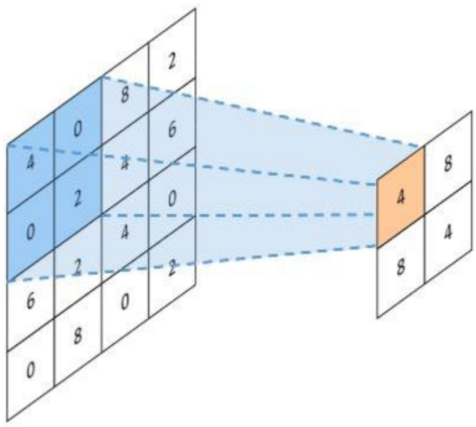

Fig 9 Max Pooling Layer

Lastly we have a Fully Connected(FC) or Dense layer. The input of the FC layer is a flattened 1-dimensional vector, thus the 3-dimensional feature maps from the Pooling layer go through flattening, where they are transformed to an appropriate sequential 1-D vector. In this layer, each neuron is connected to every other neuron from the previous layer, thus the name. So each element from the vector is passed to the neuron, where there is the corresponding weight associated for every element/feature, and a weighted sum is performed on these elements. Then an activation function is applied to the sum to introduce non-linearity. The single scalar output is then given as an input to neurons of the next layer along with other outputs, forming a vector.

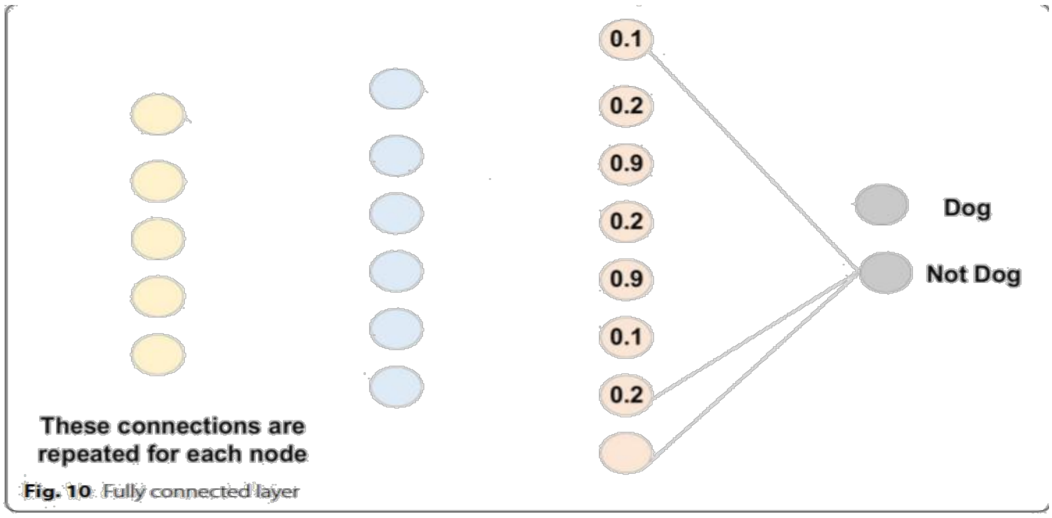

Fig 10 (Fully connected layer)

4.2 DDQN model:

Reinforcement learning is a branch of machine learning, which aims at developing algorithms, that learn by interactions. In Reinforcement Learning, we have an agent, that interacts with the environment through states, where on given each state, the agent has to take some action. The actions are further evaluated by the rewards that the action attained. Rewards act as a feedback loop, which helps the agents optimize their decision-making, to maximize the future cumulative reward. There are various types of Reinforcement Learning algorithms, which differs in the reward system, feedback loop, or function approximation.
One such type of Reinforcement Learning algorithm is a Deep Q-network (DQN), this arises by combing Deep Learning with Q-Netork, Q-Network is the most basic type of Reinforcement Learning which stores the state-action values, which are Q-values in a tabular form. The DQN uses a single neural network, which is a Policy Network to estimate the Q-values, which further leads to overestimation bias of the values. To tackle this issue, Double Deep Q-Network (DDQN) is used, which uses two separate networks, namely Policy Network and Target Network, where Policy Network selects the action and Target Network estimates the maximum future Q-values.
Fig 11 below shows the DDQN architecture

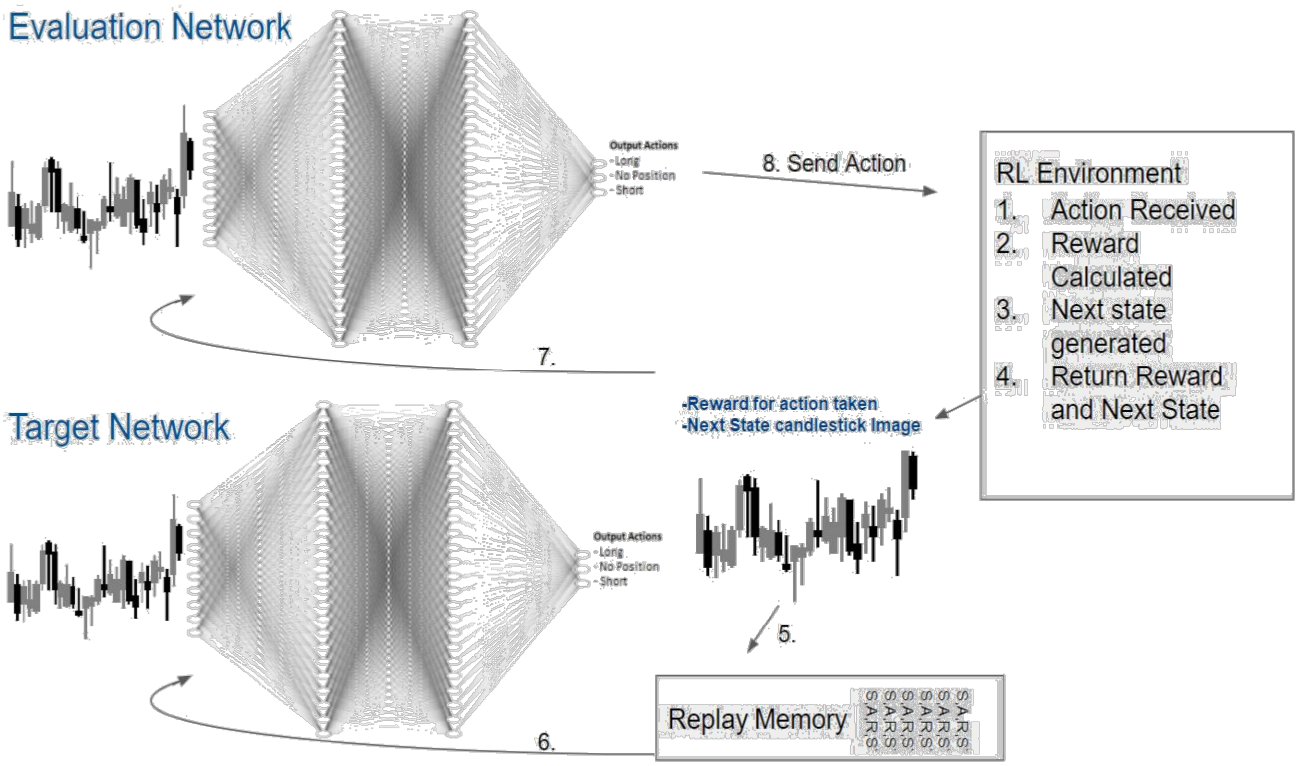

Fig 11 (DDQN architecture)

The main components of a DDQN model are as follows:

1. Policy Network: The policy network (Evaluation Network) is the main neural network responsible for approximating the action-value function (Q-values) for different state-action pairs. It takes the state representation as input and outputs Q-values for all possible actions.

2. Target Network: The target network is a copy of the policy network with frozen weights. It is used to estimate the Q-values for the next state during the Q-learning update process. By using the target network to estimate Q-values, DDQN can tackle the overestimation bias seen in the standard Q-learning algorithm.

3. Replay Memory: The replay memory stores the agent's experiences in the form of state-action-reward-nextState-nextAction (SARSA) tuples. These experiences are used for training the policy network in batches, providing a more stable learning experience.

5 Implementation

In the above sections we discussed about the steps taken to define our objective, understand our data and even the architectures of the models were discussed, while in this section we explain the data flow of our

research experiment. We discuss the technical aspects, from data extraction to image creation, model implementation, and optimization.
Fig 12 below shows the data-flow diagram

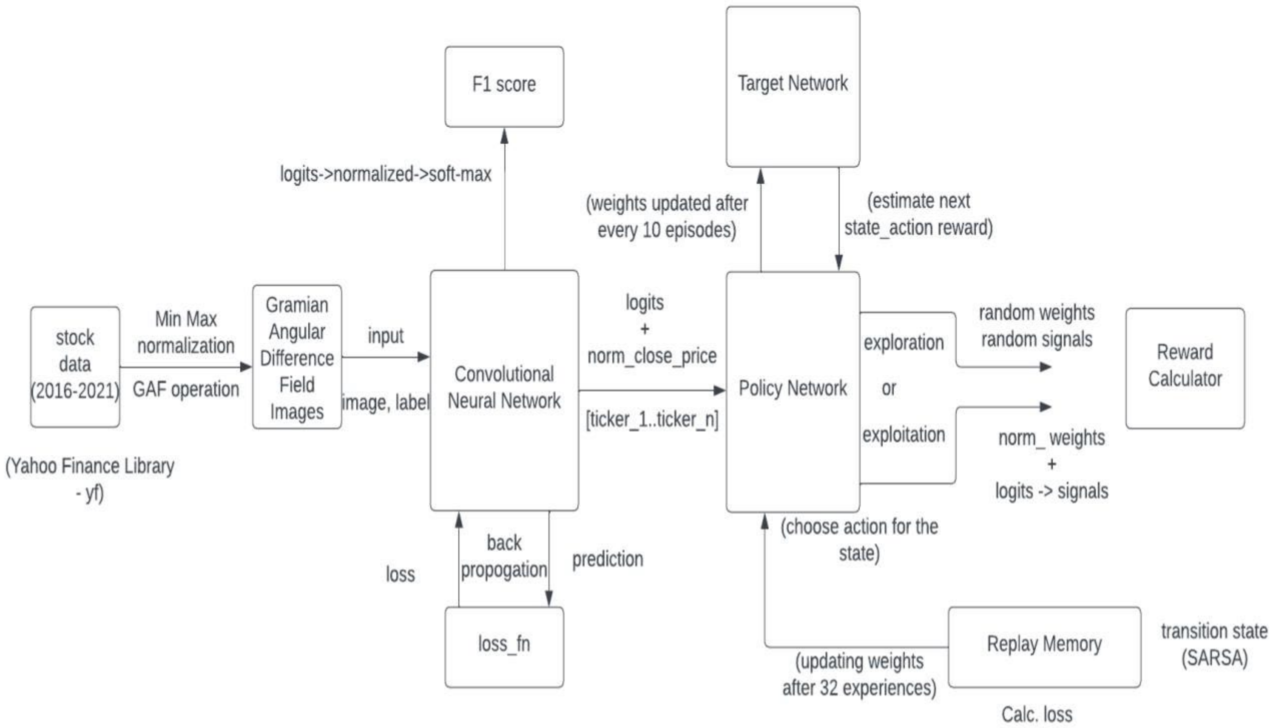

Fig 12 Data-flow architecture

5.1 Data Extraction and Exploration: The data is extracted using the Yahoo Finance Python library yf. The data is from the year 2016 to 2021, for the top 10 Nifty 50 stocks, the data frame is understood by performing various EDA techniques as mentioned in Methodology

5.2 Data Pre-processing and Construction: The data is first normalized using Min Max Normalization, here we take Minimum and Maximum values based on each quarter. The Gramian Angular Field function requires normalized data as input. We perform Gramian Angular Difference Field (GADF) operation, where a trigonometric difference is applied on each pair or data, to capture the temporal dependencies. The GADF images are created using the Matplotlib library, which is then labeled using the Bollinger Band strategy. The images are labeled as either Buy, Sell, or Hold, based on where the next_close price is based on Mean + Std_deviation*1.5. These images with labels and ticker name are split into train (2016-2019), validation(2019-2020), and test(2020-2021) set. The data is saved in Pickle files, based on the sets and their respective ticker names.

5.3 Convolutional Neural Network: We implement Convolutional Neural Network(CNN) using Pytorch, which is a framework for implementing Deep Learning models. Class 'BollinerClassifier' is responsible for creating a CNN model, which inherits the nn.Module class of Pytorch.

Below Fig 13 shows the architecture of our model

```
(baseThesis) C:\Users\hp\Dropbox\1AAAAThesis\codeBase>python helper_models.py
BollingerClassifier(
  (conv1): Conv2d(1, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu1): ReLU()
  (pool1): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv2): Conv2d(64, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu2): ReLU()
  (pool2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv3): Conv2d(32, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu3): ReLU()
  (fc1): Linear(in_features=576, out_features=128, bias=True)
  (relu4): ReLU()
  (dropout1): Dropout(p=0, inplace=False)
  (fc2): Linear(in_features=128, out_features=32, bias=True)
  (relu5): ReLU()
  (dropout2): Dropout(p=0, inplace=False)
  (fc3): Linear(in_features=32, out_features=3, bias=True)
)

(baseThesis) C:\Users\hp\Dropbox\1AAAAThesis\codeBase>
```

Fig 13 CNN architecture

Input (24*24*1), Output: Number of classes (3)

Architecture:

Convolutional Layer1: 64 output channels, kernel size 3x3, ReLU activation, followed by MaxPooling (2x2)

Convolutional Layer2: 32 output channels, kernel size 3x3, ReLU activation, followed by MaxPooling (2x2)

Convolutional Layer3: 16 output channels, kernel size 3x3, ReLU activation

Fully Connected (FC1) Layer: Input size 16x6x6 (computed from the previous layer's output size), Output size 128, ReLU activation, Dropout

Fully Connected (FC2) Layer: Input size 128, Output size 32, ReLU activation, Dropout

Fully Connected (FC3) Layer: Input size 32, Output size Num_classes 3

The input image size is 24 X 24 X 1, as it has 24-days time-stamp and 1 channel, while the output is 3 logits, representing Buy, Hold and Sell prediction raw values.

During the training we normalize the logits and apply soft-max to represent them as probabilities, to calculate the F1-score.

Further, the logits are used by the Policy Network, during the Greedy approach to calculate the trading_signals.

5.4 Double Deep-Q Network: The Deep Reinforcement Learning, Double Deep Q-Network comprises two neural networks namely, Policy Network and Target Network, which are deployed using the Pytorch class, nn.Module. The Policy Network along with CNN acts as the Q-network, which is responsible for taking action.

Select action: First the Policy Network, outputs weights, whereas the logits from CNN are used to calculate the trading signals. The action selection follows an epsilon-greedy strategy, where based on the epsilon value, the model either chooses a random action with random weights or a greedy approach to choose action and weights.

Exploration: Here the weights and randomly distributed, summing up to 1, and random actions are chosen

Exploitation: Here the weights are calculated based on the Policy Network parameters, which outputs the weight for each asset, and the trading signal is calculated based on the raw output from CNN, which are logits.

Target Network: The Target Network is responsible for estimating the target Q-values, which are then stored in the transition and help to reduce the over-estimation bias of the Deep Q-Learning algorithm. Whereas weights of the Target Network are updated from the Policy Network after every 10 episodes.

Reward calculation: Each interaction leads to a new portfolio value, which is compared to the previous value and the reward is calculated

Replay memory: We hold a replay memory, which stores state, norm_state, images, labels, weights, next_state, next_norm_state, next_images, next_labels, reward and done. The experiences are stored for a batch size of 32, and when it reaches the batch size, we use the transitions, to calculate the loss and update the weights of the Policy Network based on the loss.

5.5 Saving models and results:

Every data representation is being fed to the model, and the trained CNN model is saved in the model's directory, as mentioned in the party. As we change range [0,1], [-1,1], number of features, OHLC, OHLCV, data stacking, and averaging, whereas raw time-series data and GADF images, we tend to create 18 data representations. The models for each representation are saved in the directory, where we calculate the F1 score of each model, to evaluate them based on their ability to classify correctly.
Every model, as being saved after training, the results, meaning the F1 score, produced by the model is saved alongside. The results are saved as a pickled file and further used for comparison and evaluation.

The model's performance is evaluated based on certain factors and tests. These tests are mentioned below in the Evaluation and Results section.

5 Results & Evaluation

This section focuses on evaluating the models' performance by conducting different experiments, and observing the results to either accept or reject the null hypothesis.

We evaluate the effectiveness of the data format based on the F1 score generated by the model.

The F1 score is a common evaluation metric used to assess the balance between precision and recall in classification tasks. It takes into account both the false positives and false negatives, providing a single value that indicates the model's performance in terms of correctly identifying positive elements and reducing misclassification.

Here we use a weighted F1 score, which is a variant of the F1 score, which takes into account the number of data in each class. As we have more data which labels as hold, compared to buy and sell, it draws class imbalance, where weighted average to calculate the F1 score plays a significant role.

We conduct two tests to evaluate our model:

5.1 Experiment 1

Effectiveness of the Gramian Angular Difference Field (GADF) images to capture the significant temporal correlation.

Null Hypothesis (H0): The model's F1 score on GADF images is not different from that on raw time-series data.

Alternative Hypothesis (H1): The model has a higher F1 score on GADF images, compared to raw time-series data given as input

5.2 Experiment 2

Effectiveness of adding more variables as features.

Null Hypothesis (H0): The model's F1 score on the data with additional features is not different from that on the dataset without considering the volume.

Alternative Hypothesis (H1): The model has a better F1 score on the data consisting of Open, High, Low, Close, and Volume, compared to the dataset without considering the volume.

Test 1 - In the first test, we construct three types of datasets.

1. Raw time-series data, which includes the Open, High, Low, and Close (OHLC) in a data frame.

2. Single channel, OHLC, GADF images, formed by averaging the OHLC values. The images have a dimension of 1 X 24 X 24.
3. Multiple channels, OHLC, and GADF images, formed by stacking the OHLC values vertically. The images have a dimension of 4 X 24 X 24, where Open, High, Low, and Close are represented as 4 channels. These datasets have 2 formations each, by trying different sample ranges as [0,1] and [-1,1]
Fig 14 below shows the test results.

Window =	24 days		
Normalized =	quarterly	**Input size**	
		(OHLC)	(OHLCV)
Gasf images	Stack channels	4 x 24 x 24	5 x 24 x 24
Gasf images:	Avg of correlations	24 x 24	24 x 24
Raw timeseries:	Stack channels	4 x 24	5 x 24

f1-score of validation dataset
OHLC

	Gasf stack [0, 1]	Gasf stack [-1, 1]	Gasf [0, 1]	Gasf [-1, 1]	Raw [0, 1]	Raw [-1, 1]
2DCNN	0.81776	0.81645	**0.81261**	0.80876		
1DCNN			0.80885	**0.81077**	0.68496	0.75706

Fig 14 Test 1 results

As seen above the GADF single-channel images, with sample range [0,1] outperform all the other data formats with an F1 score of 0.81261.
Thus we reject the null hypothesis and accept the alternate hypothesis stating that GADF representation captures more significant temporal dependencies details compared to raw-time series data.

Test 2 - In the second test, we modify the dataset created for the first test. Here the dataset for the first test comprises OHLC variables, we create another set with OHLCV, adding Volume as an extra feature. So we have 12 formats of data representation, with 2 main categories one with OHLC and one with OHLCV as features.

Window =	24 days		
Normalized =	quarterly	Input size	
		(OHLC)	(OHLCV)
Gasf images	Stack channels	4 x 24 x 24	5 x 24 x 24
Gasf images:	Avg of correlations	24 x 24	24 x 24
Raw timeseries:	Stack channels	4 x 24	5 x 24

f1-score of validation dataset

OHLC

	Gasf stack [0, 1]	Gasf stack [-1, 1]	Gasf [0, 1]	Gasf [-1, 1]	Raw [0, 1]	Raw [-1, 1]
2DCNN	0.81776	0.81645	**0.81261**	0.80876		
1DCNN			0.80885	**0.81077**	0.68496	0.75706

OHLCV

			Gasf [0, 1]	Gasf [-1, 1]	Raw [0, 1]	Raw [-1, 1]
2DCNN	**0.81836**	0.81754	0.79442	0.79840		
1DCNN			0.79197	0.79512	0.69644	**0.76225**

Fig 15 Test 2 results

Fig. 15 above shows the test results.
As seen above the GADF images, with a single channel, sample range of [0,1], having features OHLCV, outperforms all the other data formats, with an F1 score of 0.81836.
Thus we reject the null hypothesis and accept the alternative hypothesis stating that adding an extra feature, Volume, created a difference in the models' performance to classify the images.

6 Discussion & Conclusion

The research was determined to define and test an optimal way to represent time-series data. A Convolutional Neural Network (CNN), as a complement to the Policy Network of a Double Deep Q-Network has been used as a deciding agent for different data representations. The data of the top 10 stocks of Nifty 50, was represented in different variations of the Gramian Angular Difference Field (GADF). The variations were created by adding extra features, stacking the images vertically, using averaging, as well as changing the sample range. We could successfully create 18 different ways to represent the data. Where a 2D CNN as well a 1D CNN were used to test images and raw-time series as a way to perform the comparative evaluation.
The results suggest that GADF images, with Open, High, Low, and Close as well as Volume, with a sample range of [0,1], where the images are vertically stacked over each other, having 5 X 24 X 24 dimensions each, outperforms all the other representations, with a weighted F1 score of 0.81836. GADF images with OHLC features averaged to form one channel, having 1 X 24 X 24 as the dimension of each image had a slightly lower score of 0.81261. The difference can be assumed to be caused by averaging, which fades away the values of each feature individually and combines them as a whole average value. Whereas the addition of the Volume feature, adds an extra channel, making the images more distinct, and creating more spatial features in the image for CNN to classify them correctly.
1D CNN model being trained on raw time-series data, with a sample range of [0,1], had the least performance with an F1 score of 0.68496. This can be due to the reason that 1D CNN highly relies on local

patterns to perform classification and raw-time series data, and does not always contain any type of repetitive local pattern, whereas a GADF image is built upon temporal dependencies, which does capture the necessary information to make distinct images.

Thus GADF images capture the necessary temporal information suitable for financial analysis, making a way to be used for trading.

The outputs of the CNN are inputs to the DDQN, which leads to better Q-value calculation by the model. Our aim to optimize the logits from CNN, as a way to give as precise inputs as possible, as a way to determine the right data representation has been achieved.

The major limitation of our work stands in having a higher number of features. More features would make the images more distinct, making them more feasible to be classified. Order book, buyers, sellers, and RSI would have acted impactful in producing more accurate logits.

7 Future Work

As we have only compared the performance of various GADF images among each other and raw time-series data, a comparative study with candlestick images constructed using Open, High, Low, and Close, along with Bollinger Bands, would draw more insights into this work, as we might assess the representation in a much broader concept. Even adding extra technical indicators, such as features like Resistance and Support Index (RSI), No. of Buyers and Sellers, and Moving Averages, would create more variations of the GADF images, giving us more dimensions to perform a comparative study. Hence adding comparing the performance with candlesticks images and adding technical indicators as extra features, can be a scope of future work.

References

1. Fama, E.(1970) *Efficient Capital Markets: A review of theory and empirical work - JSTOR, 1. Efficient Capital Markets: A Review of Theory and Empirical Work:* Available at: https://www.jstor.org/stable/2325486 (Accessed: 10 July 2023).
2. Malkiel, B. (2003) *The efficient market hypothesis and its critics - Princeton University*. Available at: https://www.princeton.edu/~ceps/workingpapers/91malkiel.pdf (Accessed: 13 July 2023).
3. Author links open overlay panelKaram Kim *et al.* (2021) *Term structure of sentiment effect on investor trading behavior*, *Finance Research Letters*. Available at: https://www.sciencedirect.com/science/article/pii/S1544612321000866 (Accessed: 14 July 2023).
4. Wang *Encoding time series as images for visual inspection and classification ...* Available at: https://www.semanticscholar.org/paper/Encoding-Time-Series-as-Images-for-Visual-and-U sing-Wang-Oates/e90666552aaaa056bc6465019632bf06917c842c (Accessed: 18 July 2023).

5. Chen, J.-H. and Tsai, Y.-C. (2020) *Encoding Candlesticks as images for patterns classification using Convolutional Neural Networks*, *arXiv.org*. Available at: https://arxiv.org/abs/1901.05237 (Accessed: 01 July 2023).
6. Alzubaidi L;Zhang J;Humaidi AJ;Al-Dujaili A;Duan Y;Al-Shamma O;Santamaría J;Fadhel MA;Al-Amidie M;Farhan L;(2021) *Review of Deep Learning: Concepts, CNN*

Architectures, challenges, applications, Future Directions, *Journal of big data*. Available at: https://pubmed.ncbi.nlm.nih.gov/33816053/ (Accessed: 07 July 2023).

7. Author links open overlay panelThibaut Théate *et al.* (2021) *An application of deep reinforcement learning to algorithmic trading*, *Expert Systems with Applications*. Available at: https://www.sciencedirect.com/science/article/pii/S0957417421000737 (Accessed: 14 June 2023).
8. Brim, A. and Flann, N.S. (2022) *Deep Reinforcement Learning Stock Market Trading, utilizing a CNN with Candlestick Images*, *PLOS ONE*. Available at: https://journals.plos.org/plosone/article?id=10.1371%2Fjournal.pone.0263181 (Accessed: 05 June 2023).
9. *Using CNN for stock prediction* (2022) *Data Platform and Machine Learning*. Available at: https://dwbi1.wordpress.com/2022/02/10/using-cnn-for-stock-prediction/ (Accessed: 22 June 2023).
10. Peng, N.Y. (2020) *Reading charts with Convolutional Neural Networks*, *Medium*. Available at: https://towardsdatascience.com/reading-charts-with-convolutional-neural-networks-cbaabd d5f478 (Accessed: 01 August 2023).

11. Das, A. (2021) *Convolution neural network for image processing-using Keras*, *Medium*. Available at: https://towardsdatascience.com/convolution-neural-network-for-image-processing-using-ke ras-dc3429056306 (Accessed: 02 August 2023).

12. Sá Sanchez, J. (2022) *Build your first CNN with tensorflow*, *Medium*. Available at: https://towardsdatascience.com/build-your-first-cnn-with-tensorflow-a9d7394eaa2e (Accessed: 25 July 2023).
13. Gu, J. *et al.* (2017) *Recent advances in Convolutional Neural Networks*, *arXiv.org*. Available at: https://arxiv.org/abs/1512.07108 (Accessed: 03 July 2023).
14. Taye, M.M. (2023) *Theoretical understanding of Convolutional Neural Network: Concepts, architectures, applications, Future Directions*, *MDPI*. Available at: https://www.mdpi.com/2079-3197/11/3/52 (Accessed: 30 June 2023).
15. van Hasselt, H., Guez, A. and Silver, D. (2015) *Deep reinforcement learning with double Q-learning*, *arXiv.org*. Available at: https://arxiv.org/abs/1509.06461 (Accessed: 16 July 2023).