DEEP Q-NETWORK BASED CRYPTOCURRENCY INVESTMENT STRATEGIES USING TRANSFORMER FUNCTION APPROXIMATOR

A Thesis

by

Tuna Alaygut

Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Computer Science

Özyeğin University
February 2024

DEEP Q-NETWORK BASED CRYPTOCURRENCY INVESTMENT STRATEGIES USING TRANSFORMER FUNCTION APPROXIMATOR

Approved by:

Asst. Prof. Dr. Emre Sefer, Advisor
Department of Computer Science
Özyeğin University

Asst. Prof. Dr. Reyhan Aydoğan
Department of Computer Science
Özyeğin University

Assoc. Prof. Dr. Arzucan Özgür
Department of Computer Engineering
Boğaziçi University

Date Approved: January 5, 2024

To my family

ABSTRACT

Cryptocurrencies have recently started to become more apparent as an alternative investment class, as they are mainly decentralized and transparent. Since cryptocurrencies have unique features and are more volatile, we need to develop novel approaches for more accurate price prediction and more profitable investment strategies. Deep learning methods including deep reinforcement learning and transformers have been attracting remarkably more attention. Here, we come up with ReinCrypt which applies Deep Q-Network (DQN) with a vision transformer (ViT) neural network function approximation, where we transform one-dimensional time-series data to 2D grayscale image-like data by using a set of important technical analysis indicators. Such transformed data will be taken as input in our framework to predict various cryptocurrency prices. To stabilize the learning process of the model and enhance its performance, we employ techniques like parameter freezing and experience replay. Our detailed findings show that the proposed method is successful in forecasting changes in cryptocurrency prices. Overall, our method yields statistically significant profit in different cryptocurrencies, not just Bitcoin. When we analyze the portfolios formed as a result of our method's output, we obtain approximately 0.5% return per transaction before considering transaction costs for the cryptocurrencies. Additionally, we can predict future cryptocurrency prices even when we train our method on a different cryptocurrency. In this case, we can train our method on highly-traded and liquid cryptocurrencies such as Bitcoin, and test its performance on relatively less liquid alternative cryptocurrencies. According to these results, deep learning-based cryptocurrency price prediction methods could be utilized for alternative cryptocurrencies, even though we have fewer data for these assets. Our code and

datasets are available at `https://github.com/seferlab/reincrypt`.

ÖZETÇE

Kripto paralar, çoğunlukla merkeziyetsiz ve şeffaf oldukları için alternatif bir yatırım sınıfı olarak son zamanlarda daha fazla öne çıkmaya başladı. Kripto paraların benzersiz özelliklere sahip olmaları ve daha oynak olmaları nedeniyle, daha doğru fiyat tahmini ve daha karlı yatırım stratejileri geliştirmek için yeni yaklaşımlar geliştirmemiz gerekiyor. Derin öğrenme yöntemleri arasında özellikle derin takviye öğrenmesi ve dönüştürücüler dikkat çekici derecede daha fazla ilgi görmeye başladı. Burada, bir Derin Q-Ağı (DQN) ile bir görüş dönüştürücü (ViT) sinir ağı fonksiyon yaklaşımını uygulayan ReinCrypt ile karşınıza çıkıyoruz. Burada, önemli teknik analiz göstergelerinden oluşan bir set kullanarak, tek boyutlu zaman serisi verilerini 2D gri tonlamalı görüntü benzeri verilere dönüştürüyoruz. Böyle dönüştürülmüş veriler, çeşitli kripto para fiyat hareketlerini tahmin etmek için çerçevemizde girdi olarak alınacak. Modelin öğrenme sürecini stabilize etmek ve performansını artırmak için parametre dondurma ve deneyim tekrarı gibi teknikler kullanıyoruz. Detaylı bulgularımız, önerilen yöntemin kripto para fiyatlarındaki değişiklikleri tahmin etmede başarılı olduğunu gösteriyor. Genel olarak, yöntemimiz sadece Bitcoin değil, farklı kripto paralarda da kar sağlıyor. Yöntemimizin çıktısı sonucunda oluşturulan portföyleri analiz ettiğimizde, işlem başına işlem maliyetlerini dikkate almadan yaklaşık %0.5 getiri elde ediyoruz. Ayrıca, yöntemimizi farklı bir kripto para üzerinde eğittiğimizde gelecekteki kripto para fiyat hareketlerini tahmin edebildiğimizi görüyoruz. Bu durumda, yöntemimizi Bitcoin gibi yüksek işlem hacmine ve likiditeye sahip kripto paralar üzerinde eğitip, performansını nispeten daha az likit alternatif kripto paralarda test edebiliriz. Bu sonuçlara göre, derin öğrenme tabanlı kripto para fiyat hareketi tahmin yöntemleri, bu varlıklar için daha az veriye sahip olsak bile alternatif kripto paralar

için kullanılabilir. Kodumuz ve veri setlerimiz şu adreste bulunmaktadır `https://github.com/seferlab/reincrypt`.

ACKNOWLEDGEMENTS

First and foremost, I extend my sincerest gratitude to my advisor, Dr. Emre Sefer, for his invaluable guidance, patience, and expert knowledge throughout my research. His mentorship was not only essential in shaping this work but also instrumental in my personal and academic growth.

I would also like to express my deepest appreciation to my family. Their unwavering support, encouragement, and love have been the cornerstone of my journey. Their belief in me has been a constant source of strength and motivation, making this achievement as much theirs as it is mine.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

The predictability of asset prices, including stocks, through artificial intelligence-based (AI-based) trading systems, has been a subject of study for nearly three decades. In contemporary financial markets, the scope of tradeable instruments extends beyond stocks to include cryptocurrencies [1], options [2], Exchange-Traded Funds (ETFs) [3], etc. According to the famous Efficient Market Hypothesis [4], stock markets are always efficient and the stock prices always incorporate the whole set of existing knowledge. According to the Efficient Market Hypothesis, technical analysis, fundamental analysis, etc. should not generate consistently above-market returns. However, many following studies [5, 6, 7, 8, 9, 10] focused on demonstrating the profitability of stock markets. These studies have mainly focused on identifying market factors and associated anomalies [8, 9, 10]. Even though the Efficient Market Hypothesis claims that these anomalies will be eliminated once transaction costs are considered, a great number of follow-up studies showed statistically significant profits not only for stocks but also for other asset classes [11, 12, 13]. For instance, [14, 15] utilizes technical analysis, which involves generating novel trading signals over historical volume and asset prices. They showed the predictability of future prices over technical analysis. Another class of work focused on generating unconventional trading signals to predict asset prices. Those studies have examined web articles [16], sentiment data [17], social network communication data [18], etc.

Deep supervised learning methods have recently exhibited superior performance in various classification and prediction tasks compared to more traditional machine learning models like Support Vector Machines (SVMs). Notably, deep learning excels in image processing tasks such as segmentation and classification, marking a significant departure from conventional methodologies [19]. A parallel trend has emerged in financial asset price prediction tasks. While more traditional time-series methods such as Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) have been employed, the application of these techniques in financial prediction lags behind their prevalence in other artificial intelligence areas such as computer vision [20, 21] and Natural Language Processing [22, 23].

In spite of the popularity of supervised techniques in deep learning for finance, these supervised approaches are limited so they end up mostly in suboptimal results [24]. On the other hand, Reinforcement Learning (RL) is a different kind of AI method where an agent learns in interactional conditions by trial and error utilizing the feedback from the agent's own experience. RL is based on reward and punishment steps to generate automated trading policy [25]. Such a time-dependent nature makes it an ideal case for Markov Decision Processes (MDP) [26] where MDPs are the central component of the RL problem solution. Markov Decision Process could define the problem's history in only the agent's existing state, which is quite important for modeling financial market data [27]. On the other hand, Deep Reinforcement Learning (DRL) combines deep learning with RL to efficiently solve more complicated decision problems [28].

Among the asset classes, cryptocurrencies are traded over blockchain where Bitcoin is the most popular cryptocurrency since its introduction. Even though crypto investments have remarkably high returns particularly during bull markets, their highly volatile nature causes the investments made to be still considered as too risky. For instance, Bitcoin price was quite volatile over time, with the maximum return as high as +1360% in 2017 [29]. Other important alternative currencies like Litecoin, XRP, and Ethereum have similar price dynamics. As a result of such price fluctuations, predicting cryptocurrency prices efficiently is important to have higher returns and low loss risk. Even though cryptocurrency price prediction can be seen as similar to stock price forecasting in terms of time-series prediction, it is currently considered one of the most difficult financial forecasting tasks [30].

Here, we propose ReinCrypt which applies Deep Q-Network (DQN) [31] with a vision transformer (ViT) neural network function approximation, where we transform one-dimensional time-series data to 2D grayscale image-like data by using a set of important technical analysis indicators. Such transformed data will be taken as input in our framework to predict various cryptocurrency price movements. Once such transformed 2D data is taken as input for each cryptocurrency, ReinCrypt selects one of Long, Short, or Neutral actions every day. Depending on this action selection and the associated price increase/decrease, a negative or a positive reward is received. Overall, through training, ReinCrypt learns to choose the action that will result in maximum cumulative rewards. We adapt stabilization techniques like experience replay and parameter freezing to stabilize the model's learning process to improve its performance. Utilizing DQN as part of ReinCrypt is important to handle the instability of the solution due to having nonlinear function approximators with Q-learning [32]. As part of this study, we also aim to extract patterns that generate profit not only for the cryptocurrency having the training data, but also across other alternative cryptocurrencies. For example, unique patterns and their relationship with price discovered over training data for a given cryptocurrency should be adaptable to other alternative cryptocurrencies as well.

Using DQN in our method instead of a traditional supervised learning technique for cryptocurrency prediction has multiple advantages. First of all, ReinCrypt is trained via rewards. In this case, considering only binary outcomes for actions is not sufficient in our price prediction problem. For instance, our method looks for receiving 5.0 reward for a 5% price increase, and 2.0 reward for a 2% price increase. Using just binary classes instead of continuous values would have decreased the performance since it wouldn't be able to differentiate between the two scenarios here. Secondly, the RL framework trains an agent over cumulative reward instead of an immediate reward. When we apply supervised learning to the asset price prediction problem, we won't be able to take into account the time steps just after the next time step. On the other hand, RL's maximization of cumulative rewards by using the knowledge from all the following time steps will be efficient for our problem. Lastly, we can obtain action values that are the expected cumulative rewards of the corresponding action as part of Q-learning's trained outcome. As a result of the training, we can also predict the profit amount which will be yielded by the action. In our case, we use a vision transformer as the function approximator where input will be 2D-like image data formed after applying technical indicators. Several previous studies take into account technical indicators while making predictions [33, 34], so conducting additional research that directly works on image-like input data from technical indicators is important.

According to the detailed experiments, our transformer-integrated DRL framework performs quite well in the highly volatile cryptocurrency domain. Across both longer and shorter testing periods, ReinCrypt is still more accurate than the simpler Buy-and-Hold Strategy and the competing CNN baseline. According to these results, deep learning-based cryptocurrency price movement prediction methods could be utilized for alternative cryptocurrencies, even though we have fewer data for these assets. While the performance of the transformer-integrated DRL approach is promising, further enhancement is achievable through more detailed hyperparameter optimization and fine-tuning. To the best of our knowledge, price movement prediction by learning over multiple technical analysis signals within DRL framework has only recently been studied [35]. In their work, the authors focus on identifying high-risk periods by using DRL where technical indicators are not processed by transformers as we do in our work. Another competing approach [36] has directly used price images without technical indicators, where they use CNN in DRL to predict international asset prices.

In our case, our function approximator Vision Transformer [37] is a state-of-the-art approach for analyzing spatiotemporal patterns in image-like data, the one formed by combining multiple technical analysis indicator signals in our case.

1.1 Related Work

Finance can be seen as one of the most focused machine learning application areas during the last 30 years. Until now, many research papers have been published. Traditional machine learning methodologies have been extensively employed for stock market forecasting, with studies focusing on applying time-series prediction techniques directly to financial datasets. Some have utilized fundamental or technical analysis techniques for accurate forecasting, as demonstrated in surveys such as [38]. Previous works, like [39, 40, 41, 42, 43], have applied Artificial Neural Networks (ANNs) to predict stock index values or forecast stock prices, integrating technical analysis indicators. [44] compared the existing common techniques for text mining which are also adapted to stock market prediction problems such as Rule-based systems, Genetic algorithms (GAs), and neural networks. Among other related research, [45] used kernel techniques to forecast stock market changes. In general, the performance of machine learning techniques is limited since they cannot infer deeper attributes in the financial dataset which is a key factor in closing the gap between machine and human trading performance.

In recent years, the surge in computational capacity has led to the emergence of newer deep learning-based approaches. Deep learning, as a specialized case of ANN with multiple layers, has demonstrated enhanced performance compared to shallower networks [46]. Various deep learning models, including Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNNs), and Transformers, have found applications across diverse domains.

CNNs, for instance, are widely used in image classification [47, 48] and video processing, as well as in natural language processing tasks such as sentence categorization [49]. LSTMs and RNNs are predominantly employed for sequential data analysis in speech processing, natural language processing, and time-series-related tasks. Transformers, initially applied to Natural Language Processing (NLP) tasks [50], have more recently demonstrated success in computer vision tasks [51, 37].

While deep CNNs and transformer-based architectures have become prevalent in the last decade, their application in financial tasks remains limited. For instance, [52] has enhanced a basic momentum-based investment strategy [7] by also integrating a feedforward neural network (FNN). The enhanced model outperformed the original momentum strategy over US stocks. In a different study, [53] has compared the performance of deep neural networks, random forest, and gradient boosting while forecasting the S&P500 index level. In a similar research with different results, [54] has found simpler Long Short Term Memory (LSTM) [55] networks outperform deeper neural networks.

Among the work that uses Reinforcement Learning (RL) in asset price prediction, [56] has come up with the RNN-based Reinforcement Learning method, which they applied to index futures markets, commodity futures markets, and stock index markets. [57] focuses on integrating LSTM and ANN techniques with Deep Q Network (DQN) to predict stock market movements. Similarly, [58] has also utilized DQN approach to infer a set of buy and sell points. Their approach has outperformed 3 traditional approaches assuming lower transaction fees. In another work, [59] integrates the BERT model with DQN to predict some Turkish market stocks. [60] formulates the trading problem as a Partially Observed Markov Decision Process model, and discusses its effectiveness for intelligent decision-making. [61] proposes a Deep RL-based automated trading system that utilizes cascades of Long Short-Term Memory by combining it with reinforcement learning. Notably, deep learning demonstrates accurate learning and generalization across buy and sell time points over extended testing horizons in these studies [62]. Another recent work [63] proposed an improved DRL-based trading strategy called State-Action-Reward-State-Action (SARSA) to increase the trading performance in stock market. As part of their DRL, they integrate Bidirectional Long Short-Term Memory (BiLSTM)-Attention technique to handle volatility in stock markets. Lastly, [64] proposed a multi-agent reinforcement learning solution which employs multi-scale CNNs and TimesNet [65] to increase the prediction performance over various stock indices.

The above-mentioned studies use only historical price datasets in their analysis. Nonetheless, several studies also integrate extra knowledge into prediction as input such as stock chart images and technical indicator-based decompositions of time-series data. For instance, [66] utilized candlestick images in predicting DJI where they used relatively simpler content-based image retrieval techniques to infer attributes from candlestick images in an automated way. In another work, [67] has instead come up with a CNN-based method to infer price movements over candlestick images. The study in [34] incorporates technical analysis indicators into prediction models, employing CNNs with two-dimensional matrix characterizations of technical analysis datasets. However, the integration of technical analysis datasets with deep learning techniques, especially the combination of CNNs with two-dimensional matrix characterizations of these datasets, is relatively rare in algorithmic trading studies [34, 68, 33]. These studies typically convert the financial time-series prediction problem into an image classification task by generating two-dimensional images from price indicators. Recent work [33] applies a single type of vision transformer and more recent convolutional mixer architecture to such image-like data for the first time. Another approach [36] has directly used price images without technical indicators, where they use CNN in DRL to predict international asset prices.

A different set of studies has utilized this extra technical indicator knowledge over cryptocurrency market. Among them, [69] has come up with a tree-based machine learning model to predict Bitcoin returns, by incorporating over 100 technical indicators. [70] used Proximal Policy Optimization (PPO) for automated Bitcoin price prediction, by designing a reward function that incorporates a subset of technical analysis indicators. However, their study has not taken the transaction costs into account, as well as their proposed approach has not outperformed simpler Buy-and-Hold baseline. On the other hand, [35] has come up with PPO-based DRL framework to differentiate between low- and high-risk periods which was an important step in enhanced cryptocurrency prediction performance. A recent review [71] has evaluated the performance of multiple algorithmic DRL trading agents on both traditional and cryptocurrency markets, where technical indicators have been found to be processed more efficiently by DRL approaches.

On the other hand, transformer-based approaches for financial market prediction are notably limited. Studies such as [72, 73, 74] propose novel transformer architectures or methods to predict stock movements, incorporating feature engineering and multi-scale Gaussian priors. These studies also integrate textual attributes from the Twitter dataset into the prediction.

CHAPTER II

PRELIMINARIES

2.1 CNN

Among deep neural networks which are used to classify significantly nonlinear patterns, Convolutional Neural Networks (CNNs) are mainly designed for image classification problems. Typically, 2D image input is first passed via multiple latent convolutional layers which is followed by pooling and nonlinear activation. After these convolutional layers, fully connected (FC) layers are used with a softmax function. In this study, CNN will be used as a baseline function approximator in the Q-learning algorithm.

2.2 ViT

A vision transformer (ViT) [37] is a transformer adapted to work with 2D image-like data. In our case, we transform the 1D price signal into multiple fundamental analysis indicators. A ViT partitions the image-like input into patches, where each patch is flattened into a vector. Afterwards, these embeddings are input into a transformer encoder similar to token embeddings where self-attention is applied. In ReinCrypt, we will use ViT as a function approximator in the Q-learning algorithm.

2.3 Q-learning

Q-learning, as one of the most frequently used Reinforcement Learning methods, focuses on an agent to learn optimal policies where the agent is trained such that it would select the action resulting in the highest cumulative rewards for a given state. Optimal policies are not directly learned via a Q-learning agent, where agent training obtains the optimal action value. In this case, given the current state, the optimal action value is each action's expected cumulative rewards. When the training is over, the optimal agent policy corresponds to a greedy policy where the maximum value action is selected by the agent. As part of learning, the Bellman Equation is used to update action value in an iterative fashion. Provided with the existing state, an agent selects the action according to policy. Afterwards, the following step and reward are observed. Q-learning generally uses ϵ-greedy policy where an agent explores a random action with probability ϵ or acts greedily.

2.4 DQN

For simpler state representations, Q-learning converges to optimal behavior. However, researchers generally use function approximation for efficient state representation when we cannot directly represent it in a table lookup. This function approximation can be any function that maps state representations to possible actions. In ReinCrypt, we use ViT to map image-like representation into three market actions: Neutral, Short, and Long. In this case, naive implementation of a nonlinear function approximation such as a neural network, makes the training procedure unstable. DQN adapts experience replay and parameter freezing to address this issue. By experience replay, we store the most recent M experiences in the memory and sample random batches at each iteration to calculate the gradient. This sampling in batches decreases the correlations in the data sequence. On the other hand, the parameter freezing step freezes the target model parameters as part of training. We maintain 2 different parameter sets and update the target network parameters with a certain frequency.

CHAPTER III

METHODS

3.1 Overview

In this chapter, we present a description of our proposed solution ReinCrypt for analyzing and predicting the performance of different cryptocurrencies. ReinCrypt is based on the collection and analysis of historical price data, including low, high, open, and close prices for a variety of cryptocurrencies. Figure 1 summarizes the reading of input and output action values for a given cryptocurrency. Here, action value means the expected cumulative rewards of an action. Figure 1 also shows the architecture of Vision Transformer (ViT) where $W \times W$ image-like input is first partitioned into patches. In our case $W = 32$, and ReinCrypt works by calculating 32 pre-defined and clustered technical indicators for 32 different time intervals, normalizing the calculated values between $[0, 255]$. Hence, creating a 32×32 grayscale image for each day for each currency. Afterwards, Vision Transformer reads this input as a 32 by 32 matrix. Figure 1c displays a 39 consecutive days sequential plot where all rows corresponding to different indicators are normalized over 32 days.

According to Figure 1, at time t, ViT of ReinCrypt generates 2 3-length vectors: ρ and η. Those vectors are important in deciding one of the three possible actions: Long, Short, Neutral. Similarly, at the next time step $t+1$, ViT is run with a different image-like data obtained via indicators where ViT decides the most possible action at time $t+1$. Among these two vectors, 3-length ρ stores action values that represent the expected cumulative reward of Long, Neutral, or Short actions respectively. They are in the range of $[0 - 1]$ and add up to 1. To illustrate, a ρ vector $[0.3, 0.6, 0.1]$ indicates that in the given state action with zero-based index 1, which happens to be

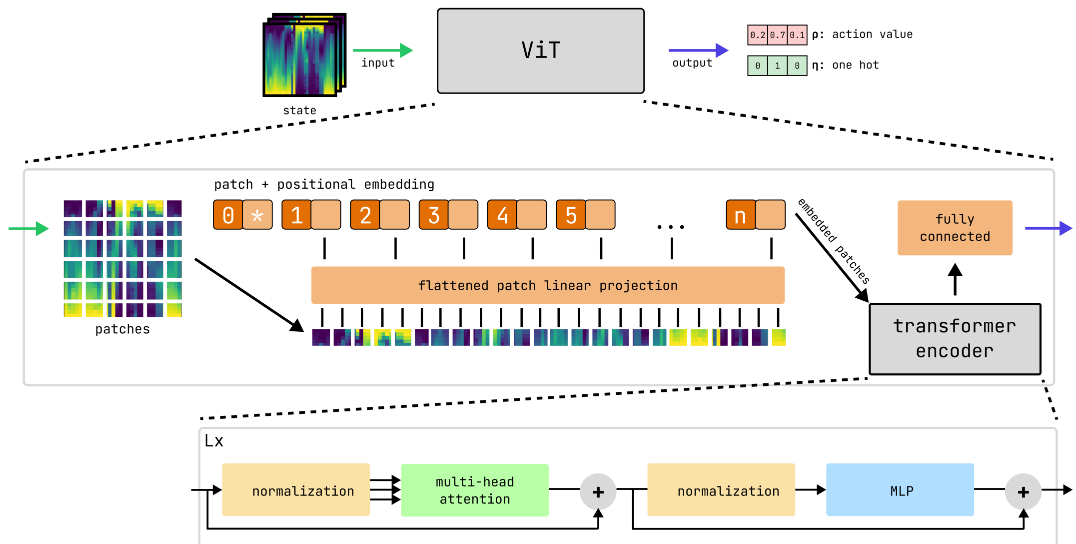


Figure 1: The illustration of ViT reading an image-like input of a single cryptocurrency at a particular time step t and outputting the two vectors ρ and τ.

neutral, is likely to yield more than other actions. 3-length one hot encoding vector η becomes 1 if the corresponding index value in ρ is the highest, and it becomes 0 for rest of the indices. For instance, $\rho[2]$ at time t stores the expected cumulative reward if our ViT takes the Neutral action at t. If ρ is $[0.3, 0.6, 0.1]$ then, η is $[0, 1, 0]$. While executing, ReinCrypt reads an image-like data at time t and selects the action with the highest action value. At the next time step $t+1$, a reward is received depending on the action at time t and the price update between t to $t+1$.

3.2 ViT Architecture

Transformers have frequently resulted in state-of-the-art outcomes for many natural language processing tasks. However, they cannot be directly applied to image-like data. A natural adaptation is to apply attention together with convolutional networks, or to modify only certain portions of convolutional networks without modifying the overall architecture. Direct self-attention application on image-like data will result in a very high complexity since each pixel should attend to every other pixel. Different solutions have been proposed to decrease this complexity [75, 76] which apply self-attention to local neighborhoods in a scalable way.

Following such adaptations, researchers came up with ViT (Vision Transformer) [37] which can be seen as a pure transformer for image-like data. ViT applies self-attention directly to sequenced image patches, where it completely removes the dependence on CNNs. Since its proposal, ViT has been integrated into many computer vision tasks including image segmentation, and image classification. Its performance has been state-of-the-art in those domains. In ViT, we have a patch and position-based embedding layer that first partitions the image into sub-images. Once images are partitioned, it is flattened and a transformer encoder which has multi-layer attention is applied as in Figure 2.

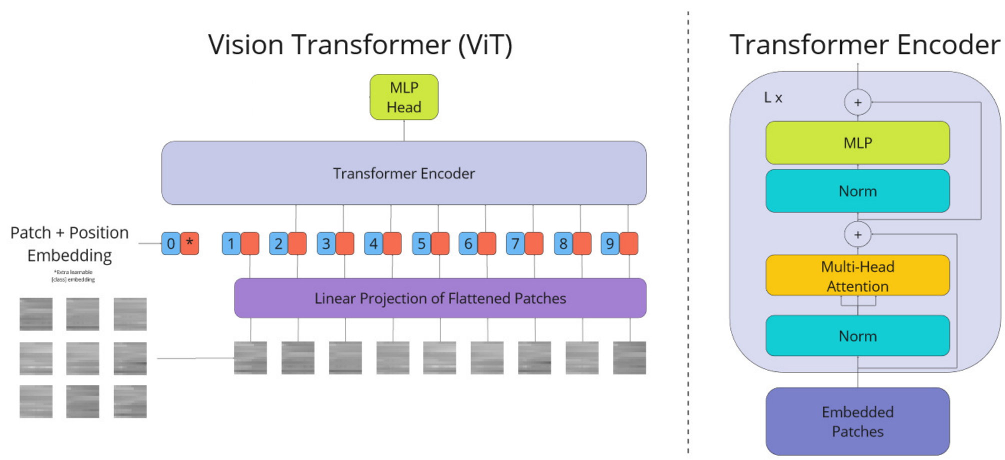


Figure 2: Overview of ViT architecture.

In more detail, given an image of $W \times H$ and a patch size B, ViT first transforms the image into $\frac{W}{B} \times \frac{H}{B}$ size components. In our case, we have a single channel so $C = 1$ and (P, P) defines every image patch's resolution. In total, we have $N = \frac{HW}{P^2}$ patches which can also be seen as the effective input sequence length for the Transformer. Once these $\frac{W}{B} \times \frac{H}{B}$ are flattened, a multi-layer encoder is applied. In each encoder layers, as defined in [37], the forward equations are follows:

$$z_0 = [x_{c_lass}; x_p^1 E; x_p^2 E; \ldots ; x_p^N E] + E_{pos},$$
$$E \in R^{(P^2 \cdot C)\times D}, E_{pos} \in R^{(N+1)\times D} \tag{1}$$

$$z'_l = MSA(LN(z_{l-1})) + z_{l-1}, \quad l = 1 \ldots L \tag{2}$$

$$z_l = MLP(LN(z'_l)) + z'_l, \quad l = 1 \ldots L \tag{3}$$

$$y = LN(z_L 0) \tag{4}$$

where MSA represents multi-headed self-attention, MLP represents multilayer perception, and LN represents layer normalization in the Transformer Encoder part. The Transformer utilizes the same hidden vector length D across all its layers, so we map flattened patches to D dimensions via a trainable linear projection. The output of this projection is referred to as patch embeddings. ViT adds positional embeddings to patch embeddings to keep positional knowledge. In the last step, we do not implement the softmax function since ViT does not output a probability distribution, but instead outputs an action value.

3.3 Dataset Preparation

Cryptocurrency price datasets were collected by using yfinance library [77]. Time-series datasets containing various daily information on each cryptocurrency such as open, close, high, and low prices as well as trading volume. We have applied two different testing schemes:

1. We downloaded the data for 291 cryptocurrencies. 283 of those cryptocurrencies, which contain 1583 days of data (between dates 2018-06-19 and 2022-10-18), are used for training and the other 8, which contain 2732 days of data (between dates 2015-04-27 and 2022-10-18), were used for validation and portfolio construction. Such dataset consists of both major bear market periods such as

the beginning of the 2020 coronavirus crisis as well as bull market periods such as the 2021 expansion.

2. We focus on the 18 cryptocurrencies with the largest market capitalization, where we focus on post-COVID era data between 2021-04-30 and 2024-01-13. In this case, we train ReinCrypt by using the data between 2021-04-30 and 2023-01-12, and test the performance for the remaining one-year period between 2023-01-13 and 2024-01-13. This post-COVID era includes both bull and bear market periods, which can be considered different than pre-COVID era bull and bear periods. Those 18 cryptocurrencies are: BTC, ETH, BNB, SOL, XRP, ADA, DOGE, DOT, TRX, SHIB, MATIC, LINK, LTC, BCH, ETC, SAND, MANA, AVAX.

We applied 32 commonly-used technical indicators [78] which focus on different time horizons on distinct categories such as volatility, momentum, volume, overlap studies, and statistics. The technical indicators used in the image creation process are chosen based on their ability to provide insights into the underlying trends and patterns in the data. We use TA-Lib (Technical Analysis Library) in Python to calculate these indicator values over time. All used indicators are summarized in Table 1.

Table 1: Technical Analysis Indicators used in our study

Category	Indicators
Statistics	LINEARREG, STDDEV, TSF, VAR
Volume	ADOSC
Volatility	ATR, NATR
Momentum	ADX, ADXR, AROONOSC, CCI, CMO, DX, MFI, MOM, PLUS_DI, PLUS_DM, ROC, ROCP, RSI, ULTOSC, WILLR
Overlap Studies	DEMA, EMA, KAMA, MA, MIDPOINT, MIDPRICE, SMA, T3, TRIMA, WMA

3.4 Turning Technical Indicators into Images

Technical indicators are pattern-based mathematical formulas that can be used to predict the direction of a financial entity given its historical data. They are commonly used in financial analysis to help traders and investors make informed decisions about buying and selling assets. There are many different types of technical indicators, each with its own set of rules and parameters for calculating its values. This 32 set of technical indicators is chosen based on their ability to provide insights into the underlying trends and patterns in the data. For each cryptocurrency, 32 technical indicators were calculated for all days (rows of data), creating 32 arrays of indicator values. Some of these arrays may be shorter than others due to the varying amount of data points required to calculate each indicator. For example, some technical indicators may need to consider data from the past 60 days to accurately predict future price movements, while others may only need data from the past 30 days. To make the arrays uniform and suitable for analysis, they are all sliced from the beginning to be the same size as the shortest array. This ensures that all the arrays have the same number of elements and can be easily compared and analyzed together.

Even though image-like matrix columns representing time steps are already ordered temporally, there is no such apparent ordering for technical indicators. The indicator ordering is a key factor in our results since we will obtain distinct images for distinct orderings. We enhance the performance of our ReinCrypt by clustering technical indicator signals. We use 4-means clustering [79] to bring signals that produce similar outcomes together. Each of the 32 technical indicators was calculated for each cryptocurrency and fed to a clustering model with 4 clusters. Afterwards, similar indicators were placed next to each other to increase spatial closeness which will be important in identifying spatiotemporal patterns via ViT. After technical indicators are calculated and clustered, images are created by scaling the values of each 32×32 array to the range $[0, 255]$ via min-max scaling. Such standardization makes the dataset more robust and consistent [80]. Once the images are created for each day, we calculate the daily return percentage by calculating the percentage difference between closing prices of two consecutive days. Since cryptocurrency is quite a volatile asset class, we make our predictions more robust by bounding returns whose absolute values are greater than 20% to 20%. Moreover, for each column of the image-like data such as close price, volume, etc., we used z-score to detect and remove outliers by replacing them with the mean of a rolling window of size 14. Figure 3 shows a sample 32×32 generated image by following the procedure above.

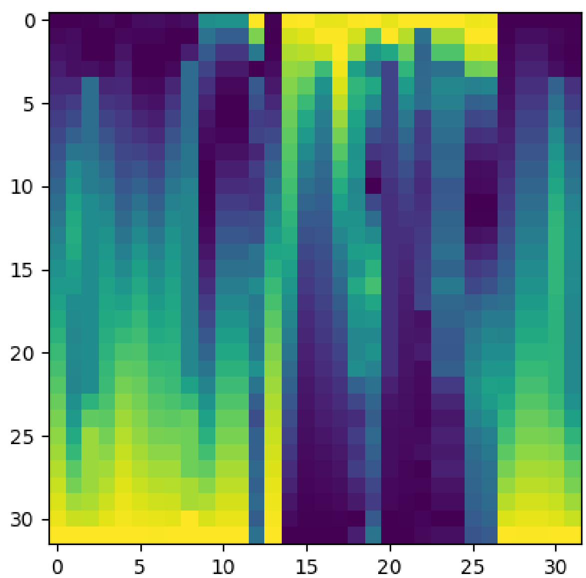


Figure 3: A sample 32×32 generated image

3.5 ReinCrypt: DQN with Transformer Function Approximator

After images are created from the financial data, they are fed into the deep reinforcement learning model, specifically a Deep Q-Network (DQN). This type of model is designed to learn to make decisions in a sequential decision-making process by learning the value of taking a particular action in a given state. To help the model learn more effectively, we used a vision transformer (ViT) function approximator for the Q-values, which is a type of neural network that is particularly well-suited for processing visual data. This allows the model to learn more effectively from the image representation of the financial time-series data.

To stabilize the learning of our model, we used two important techniques: parameter freezing and experience replay. Parameter freezing involves training two networks, called the online network and the target network, that are initialized with the same weights. The online network is updated more frequently during training, while the target network is updated less frequently and is considered to be the "truth" when training the model. Experience replay involves storing a large number of experiences (a tuple of the current state, action, next state, and reward) and applying an ϵ-greedy policy to sample a random action or a batch of experiences to learn from. This helps to break the temporal dependencies between the experiences and allows the model to learn more effectively from a diverse set of experiences. Overall, these techniques help to stabilize the learning of the model and improve its performance in predicting future price movements.

3.6 ReinCrypt Training

ReinCrypt training algorithm is summarized in Algorithm 1. We use a DQN with a transformer-based Q-value function approximator with the hyperparameters provided in Table 2. A Q-learning algorithm uses the Bellman Equation, which states the dependence between the current action value given a state $Q(s, a)$ and the following action value $Q(s', a')$, to iteratively update q-values of actions (also called action values) based on the acceptance that an optimal action value satisfies the equation. ReinCrypt train procedure integrates the 2 extra DQN techniques: Parameter freezing and experience replay. Overall, main loss $L(\theta)$ will be defined as in Eq. 5:

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n}[r + \gamma \max_{a'} Q(s', a'; \theta^*) - Q(s, a; \theta)]^2 \tag{5}$$

where L is the loss function, n is the batch size, and γ represents the discount factor. Additionally, r, s, a, s', and a' represent reward, the current state, action, subsequent state, and subsequent action, respectively. Adam optimizer [81] is executed while calculating the gradient of loss $L(\theta)$ with respect to θ. In Figure 1, state s and action a correspond to the image-like input and output action at time t respectively. Similarly, the following state s' and action a' also correspond to image-like input and output action at time $t+1$ respectively. Given that ViT is parameterized by θ, scalar $Q(s, a; \theta)$ corresponds to the action value of action a given state s. As a result, given state s, if action a corresponds to *Short*, then $Q(s, a; \theta)$ exactly corresponds to output $\rho[1]$ from our ViT. We maintain both the network parameters θ and the target network parameters θ^* during the training execution for parameter freezing implementation. At the start of training, we initialize both the network and target network parameters with identical values. In parameter freezing, the gradient step is only performed over $L(\theta)$ concerning the network parameters θ at each iteration, without being run over target network parameters θ^*. In this case, we update the target parameters θ^* at each C iteration where θ is directly copied to θ^*.

Even though training algorithm 1 is similar to default Q-learning training, we have several modifications. First of all, we need knowledge regarding the previous actions to estimate the current reward r_t^c at time t for cryptocurrency c as shown in:

$$r_t^c = a_t^c \times K_t^c - P \times |a_t^c - a_{t-1}^c| \tag{6}$$

where a_t^c, K_t^c, and P represent cryptocurrency c's action at time t, price return, and transaction penalty respectively. a_t^c is assigned -1, 0, or 1 to a_t^c for short, neutral, or long actions respectively for currency c at t. As a result, Eq. 6's leftmost term can be interpreted as the profit made by selecting the action at a given state. The next term models the transaction costs, which are activated when the model modifies the position at time t. In the training, P is not equal to 0 to prevent frequent position changes which would not have been realistic due to high transaction costs.

According to Eq. 6, we should know the action at the previous time step $t-1$ for currency c while estimating the current reward. The action at the previous time step $t-1$, a^c_{t-1} is selected via ϵ-greedy policy as well. Different than the traditional Q-learning approach, the current action does not impact the following state in our modified approach. As a result, during experience replay execution, our training approach should know the previous state, and ϵ-greedy policy is executed to obtain the previous action.

Table 2: Hyperparameters used in ReinCrypt.

Hyperparameter	Description	Value
learning rate	Learning rate	0.001
n	Batch size	32
ϵ_m	Minimum value of epsilon	0.01
max_iter	Number of iterations	1000
W	Width and height of input matrix	28
M	Memory buffer capacity	10000
B	Update frequency of θ	1000
C	Update frequency of $\theta*$	1000
γ	Discount factor	0.99
P	Transaction penalty	0.01

Algorithm 1 ReinCrypt

Initialize θ
$\theta^* = \theta$
$\epsilon = 1$
for all *b = 1, max_iter* **do**
  $c \leftarrow$ Assign a valid cryptocurrency randomly
  $t \leftarrow$ Assign a valid date index randomly
  With ϵ probability, $a^c_{t-1} \leftarrow$ random value
  Otherwise $a^c_{t-1} \leftarrow \max\limits_a Q(s^c_{t-1}, a; \theta)$
  With ϵ probability, $a^c_t \leftarrow$ random value
  Otherwise $a^c_t \leftarrow \max\limits_a Q(s^c_t, a; \theta)$
  R $\leftarrow a^c_t \times K^c_t$ - P $\times |a^c_t$ - $a^c_{t-1}|$
  S' $\leftarrow s^c_{t+1}$
  A $\leftarrow a^c_t$
  S $\leftarrow s^c_t$
  $e_b \leftarrow \{$S,A,R,S'$\}$
  Append e_b to memory buffer
  if $\epsilon > \epsilon_m$ **then**
    $\epsilon \leftarrow \epsilon \times 0.9999$
  end if
  if Memory buffer becomes full **then**
    Remove the oldest experience from the buffer
  end if
  if b % B == 0 **then**
    Randomly sample n items batch from memory buffer
    $L \leftarrow 0$
    for all *k in sampled batch* **do**
      $S_k, A_k, R_k, S'_k \leftarrow$ from e_k
      $L \leftarrow L + [R_k + \gamma \max\limits_a Q(S'_k, a; \theta^*) - Q(S_k, A_k; \theta)]^2$
    end for
    $L \leftarrow \frac{L}{n}$
    Apply gradient step for loss minimization regarding parameters θ
  end if
  if b % (B*C) == 0 **then**
    $\theta^* \leftarrow \theta$
  end if
end for

CHAPTER IV

EXPERIMENTAL SETUP

4.1 Baselines

We compared our ReinCrypt with four baselines. Among them, the simplest Random Actions approach selects buy and sell actions randomly instead of running a model. This model will be important in analyzing the statistical significance of our results. Secondly, the Buy-and-Hold strategy is a traditional trading strategy where the portfolio is not dynamically rebalanced. The investor assumes a long position and holds on to the assets for the whole period. This strategy operates under the assumption that an asset's value tends to increase over the long term despite short-term fluctuations and volatility. The remaining two baselines are similar to ReinCrypt except the two-dimensional ViT transformer is replaced with 1- 2D CNN function approximation similar to [36, 34], 2- Traditional one-dimensional transformer function approximation [82] instead of two-dimensional one where one-dimensional transformer baseline does not take technical analysis indicators into account. In the last two baseline approaches, we used Top/Bottom-k portfolio construction with multiple K values $(0.2, 0.3, 0.4)$. By using this portfolio construction method, unlike the Buy-and-Hold strategy, we saw a slight increase in the cumulative assets performance.

4.2 Detailed Portfolio Construction

Our ReinCrypt is designed to take image-like data as input and output an action for a cryptocurrency in a given time t. When designing experiments (and in real-life applications), however, we have to take into consideration more than one cryptocurrency. To achieve that, we create a vector α of length N (the number of cryptocurrencies in this case), also called a portfolio vector that satisfies $\sum_{c=1}^{N} |\alpha[c]| = 1$ given the output of our model. Constructing such a vector allows our model to make predictions regarding N cryptocurrencies at time t rather than just one. Portfolios are constructed using the outputs ρ_c and η_c, which are, as mentioned previously, action value and one-hot encoded version of the action value vectors, respectively. Also, portfolios are constructed such that $1 \leq c \leq N$. In turn, $\alpha[c]$ determines how much of the total asset should be invested into currency c. Having a value that is less than 0 in vector $\alpha[c]$ signals that a short position should be taken on currency c.

Portfolios are constructed from the central idea of maximizing return while minimizing risk, provided with the user's risk appetite. Considering our model's predictions, we ensure that assets are allocated to all different cryptocurrencies instead of just a few of them. A natural approach for achieving this would be if on obtaining the predicted returns against each cryptocurrency, they are multiplied by their absolute values and then aggregated. Ideally, since one is interested in maximizing return, it will always hold more of the first asset on a cryptocurrency with high positive predictability. To reduce risk exposure, it will always help not to become too exposed to any given asset. Let N represent the number of cryptocurrencies, and $\alpha[c]$ be the proportion of assets allocated to cryptocurrency c, then it is calculated as:

$$\alpha[c] = \frac{\rho_c}{\sum_{i=1}^{N} \rho_i} \tag{7}$$

where it is normalized the predicted return of the cryptocurrency by the sum of the predicted returns of all cryptocurrencies. This ensures that the allocation is done based on relative predicted returns. It is worth noting that the cryptocurrency market is highly dynamic. Predictions made at time t might not hold after a certain period, say $t + \delta$. Hence, a model forecast has to be constantly monitored and rebalanced occasionally to fit in with the latest model outputs and that of the investor's risk appetite.

In our experiments, we used two different portfolio construction methods as done in [36]. The first one is called the Top/Bottom-k portfolio which considers taking a position when either a buy or sell signal is robust. A Top/Bottom-k portfolio takes a long position in K assets that are expected to have the highest returns. It is a way to potentially maximize returns by selecting the assets that are expected to perform the best. Additionally, it also shorts the K assets that are expected to have the lowest returns.

The second portfolio construction technique is a market-neutral portfolio. Market-neutral portfolios are a type of investment portfolios that aim to generate positive returns regardless of market trends by systematically using both long and short positions in different assets to cancel out market risk. While these portfolios can be an attractive option for investors seeking to generate returns without being exposed to market risk, it is important to recognize that they can be complex to manage and may involve significant risks due to the inherent uncertainty of individual asset performance and market conditions. To successfully implement a market-neutral strategy, it is essential to possess a comprehensive understanding of the assets being traded and the underlying market dynamics, and to continually monitor and adjust the portfolio as needed to maintain the desired level of market neutrality.

4.3 Performance Evaluation

We evaluate the performance of ReinCrypt financially by simulating the proposed daily trades made over the test dataset. We can either buy, sell, or hold an asset. When we predict the label as "Buy", we buy the asset by using all our available capital if it has not been already bought. We assume that we start with $1 cash at the beginning. We evaluate the portfolio performance by Sharpe Ratio [83], Maximum

Drawdown (MDD) and Maximum Weight (MW) which are defined as:

$$\begin{aligned}
\text{Sharpe Ratio} &= \frac{\text{Expected Portfolio Return} - \text{Risk-Free Rate}}{\text{Portfolio Standard Deviation}} \\
&= \frac{E[R_p]}{\sigma} = \frac{E[R_p]}{\sqrt{\text{var}[R_p]}} && (8) \\
\text{Max Drawdown (MDD)} &= \frac{\text{Lowest Value} - \text{Peak Value}}{\text{Peak Value}} && (9)
\end{aligned}$$

where given the volatile nature of cryptocurrencies, we can ignore the risk-free rate, simplifying the Sharpe ratio. The Sharpe Ratio is a measure of the risk-adjusted return of a portfolio, with higher values indicating a better risk-adjusted return. The MDD represents the maximum percentage loss from a peak to a trough of the portfolio, with lower values indicating a smaller maximum loss. Lastly, MW is the maximum weight of a single asset in the portfolio, with lower values indicating a more diversified portfolio. Our code and processed datasets are available at `https://github.com/seferlab/reincrypt`.

CHAPTER V

RESULTS

5.1 *Performance under Top/Bottom-k Portfolio Construction*

While executing Top/Bottom-k portfolio, various k values and corresponding cumulative asset values obtained by running ReinCrypt can be seen in Table 3. Similarly, our ReinCrypt obtained a cumulative asset value close to 20 for a period after one year by using Top/Bottom-k portfolio construction which can be seen in Figure 4.

Table 3: Performance of Top/Bottom-k portfolios for testing cryptocurrency assets.

	Top/Bottom-k			Average
	k=0.2	k=0.3	k=0.4	
Cumulative Asset	22.43	26.36	10.62	19.80

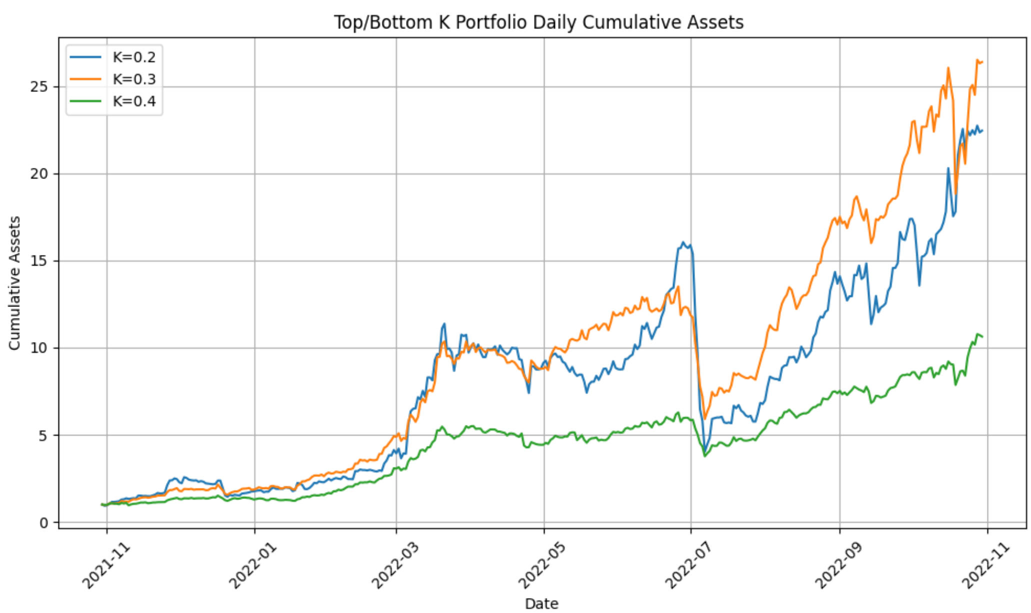


Figure 4: Cumulative asset over time by using Top/Bottom-k Portfolio.

Table 4 presents the portfolio performance for the testing period in terms of Sharpe Ratio, Maximum Drawdown (MDD), and Maximum Weight (MW).

Table 4: Performance metrics for different portfolios.

Portfolio	Sharpe Ratio	MDD	MW
Top/Bottom $k = 0.2$	2.53	-74.89%	0.25
Top/Bottom $k = 0.3$	3.49	-56.31%	0.33
Top/Bottom $k = 0.4$	3.14	-40.05%	0.50
Market Neutral	2.35	-39.25%	0.13

5.2 Performance under Market-Neutral Portfolio Construction

Again, our model achieves a cumulative asset of about 6.5 using a Market-Neutral portfolio. Cumulative assets over time for a market-neutral portfolio can be seen in Figure 5. In all experiments, we start with a cumulative asset of 1.0. Therefore, if an experiment is finished with a cumulative asset of 1.5, it means that our model was able to increase the cumulative asset by 50%.

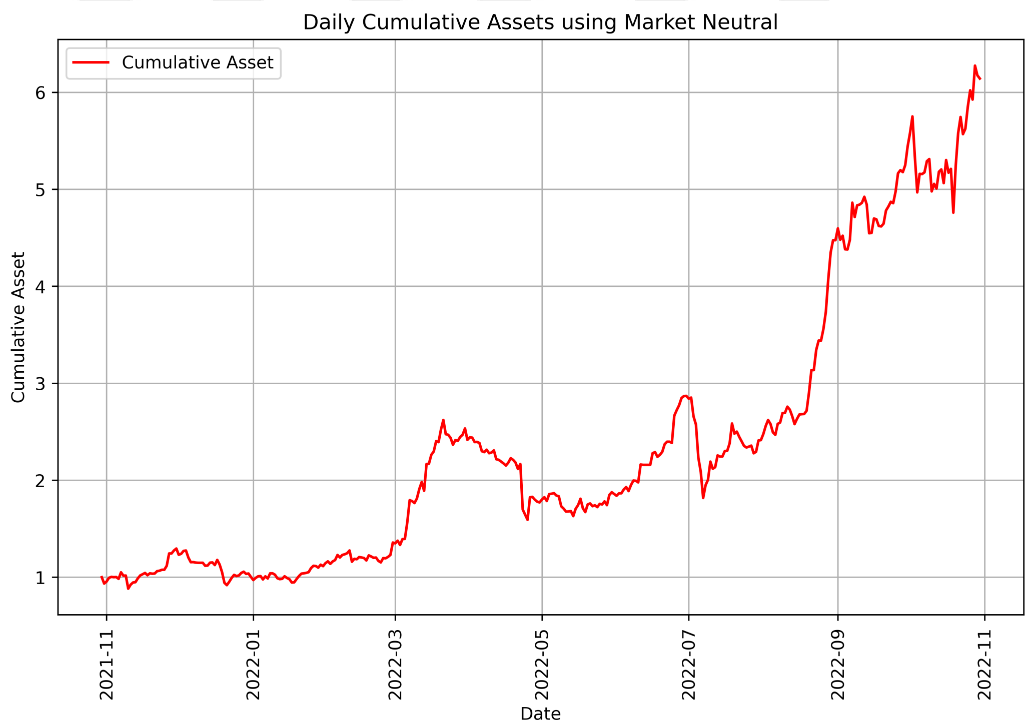


Figure 5: Cumulative asset over time using Market Neutral Portfolio.

The results in Table 5 demonstrate that distributing a larger share of assets to a smaller number of cryptocurrencies with higher stock values results in higher annual returns than the market average. The study does not take into account transaction costs, however, the data still shows that annual returns are typically significantly higher than the market average returns for most countries during the period analyzed.

Table 5: Number of transactions, gain per transaction, and annual returns for different cryptocurrencies, while executing market-neutral portfolio construction.

Cryptocurrency	# of transactions	Gain per Transaction	Annual Return
DGC-USD	65	1.4%	92%
NMC-USD	96	0.9%	87%
FTC-USD	49	1.2%	59%
PPC-USD	141	0.8%	113%
BTC-USD	46	1.2%	56%
LTC-USD	167	0.4%	67%
GLC-USD	180	0.6%	108%
TRC-USD	130	0.6%	78%

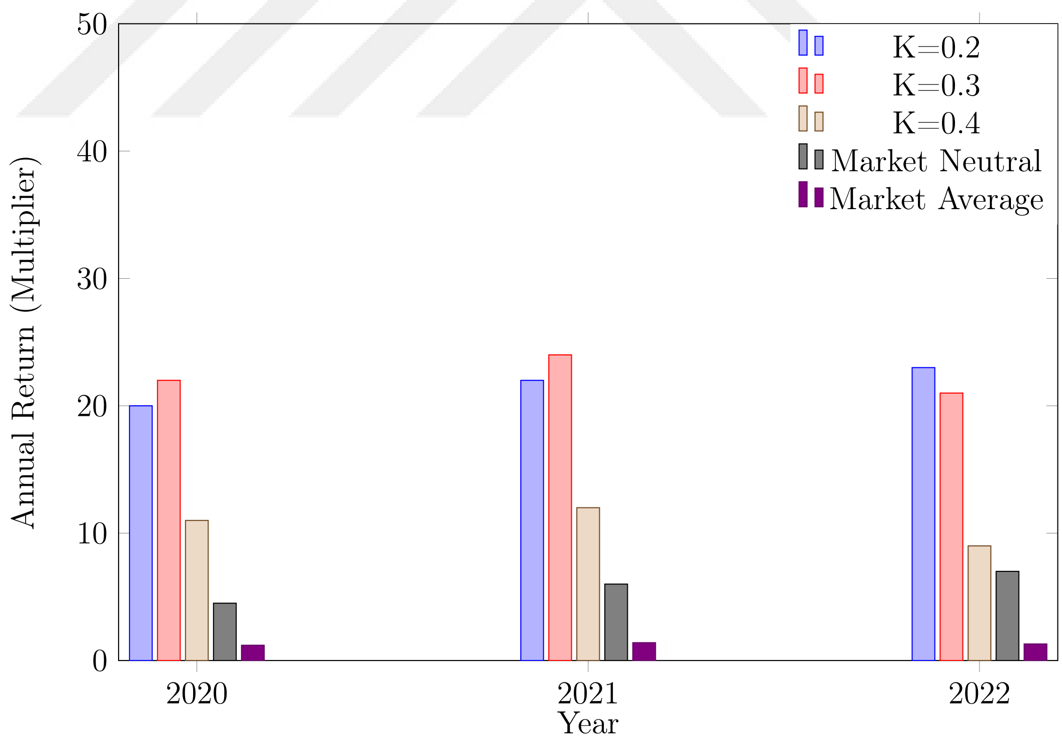


Figure 6: Annual return multiplier for different portfolios.

5.3 Statistical Significance

We compare ReinCrypt results with random portfolios to evaluate the statistical significance. We calculate the average of results over the whole test period for market-neutral and random Top/Bottom portfolios. For each case, we run 10000 simulations and report the average return μ and standard deviation σ of the annual return. While generating a random portfolio for market market-neutral case, we randomly select a value within -1 and 1, neutralize, and construct a random market-neutral portfolio α_n by normalizing the selected values. As a result, $\sum_{c=1}^{N} \alpha_n[c] = 0$ and $sum_{c=1}^{N} |\alpha_n[c]| = 1$. On the other hand, while generating a random Top/Bottom-k portfolio, we follow a similar approach. We took a long position in random $k\%$ of cryptocurrencies and a short position in the remaining random set of $k\%$ cryptocurrencies.

We use z-score for statistical evaluation since we assume portfolio returns to be distributed normally. Table 6 summarizes statistical results for different portfolio construction types. σ increases when k increases since higher k in the Top/Bottom portfolio corresponds to distributing assets to fewer cryptocurrencies. By using z-scores, we assess the performance of our ReinCrypt with distinct portfolio construction types with random portfolios.

Table 6: Statistical results of ReinCrypt for different portfolio construction types where ReinCrypt is compared with random portfolios. We calculate the average of results over the whole test period.

Portfolio Type	μ	σ	z
Top/Bottom $k = 0.2$	0.15	0.09	1.69
Top/Bottom $k = 0.3$	0.18	0.11	1.64
Top/Bottom $k = 0.4$	0.21	0.13	1.59
Market Neutral	0.12	0.07	1.73

5.4 Comparison with Baselines

We compared our ReinCrypt with four baselines as discussed in Section 4.1. In all these baseline approaches, when we use Top/Bottom-k portfolio construction, we test the performance with multiple k values $(0.2, 0.3, 0.4)$. Among these baselines, Figure 7 shows the performance in terms of cumulative asset value for a period after one year by using Top/Bottom-k portfolio construction for Random Actions case. Another baseline Buy-and-Hold baseline strategy creates a portfolio with each 8 cryptocurrency equally weighted. As shown in Figure 8, this static strategy lost about 65% of the cumulative asset in a year between 30 October 2021 and 30 October 2022.

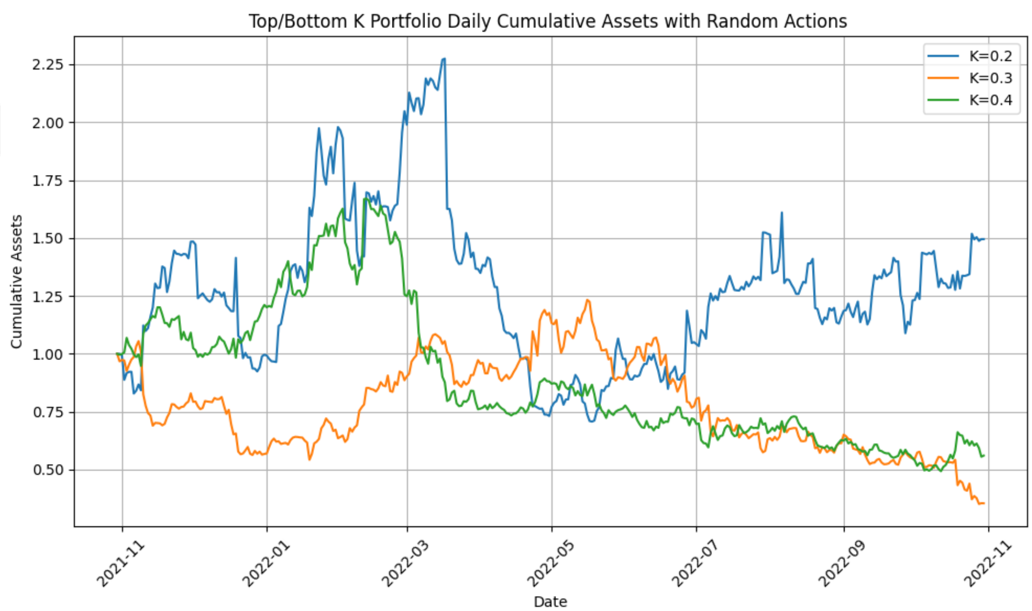


Figure 7: Cumulative asset over time by using Top/Bottom-k Portfolio for Random Actions.

We also compare our method by replacing ViT with CNN for the DQN function approximator as in [36], also similar to the one demonstrated in [34]. In this case, we see an overall positive return as seen in Table 7. Table 7 shows the resulting cumulative assets at the end of one year for different k values. By using the portfolio construction methods, unlike the Buy-and-Hold strategy, we obtained a slight increase in the cumulative assets. Among multiple k values, we obtain on average 56% return. Figure 9 shows cumulative asset change in a year between 30 October 2021 and 30 October 2022.

We also compare ReinCrypt by replacing ViT with traditional 1D transformer

Figure 8: Cumulative assets over time using Buy-and-Hold Strategy.

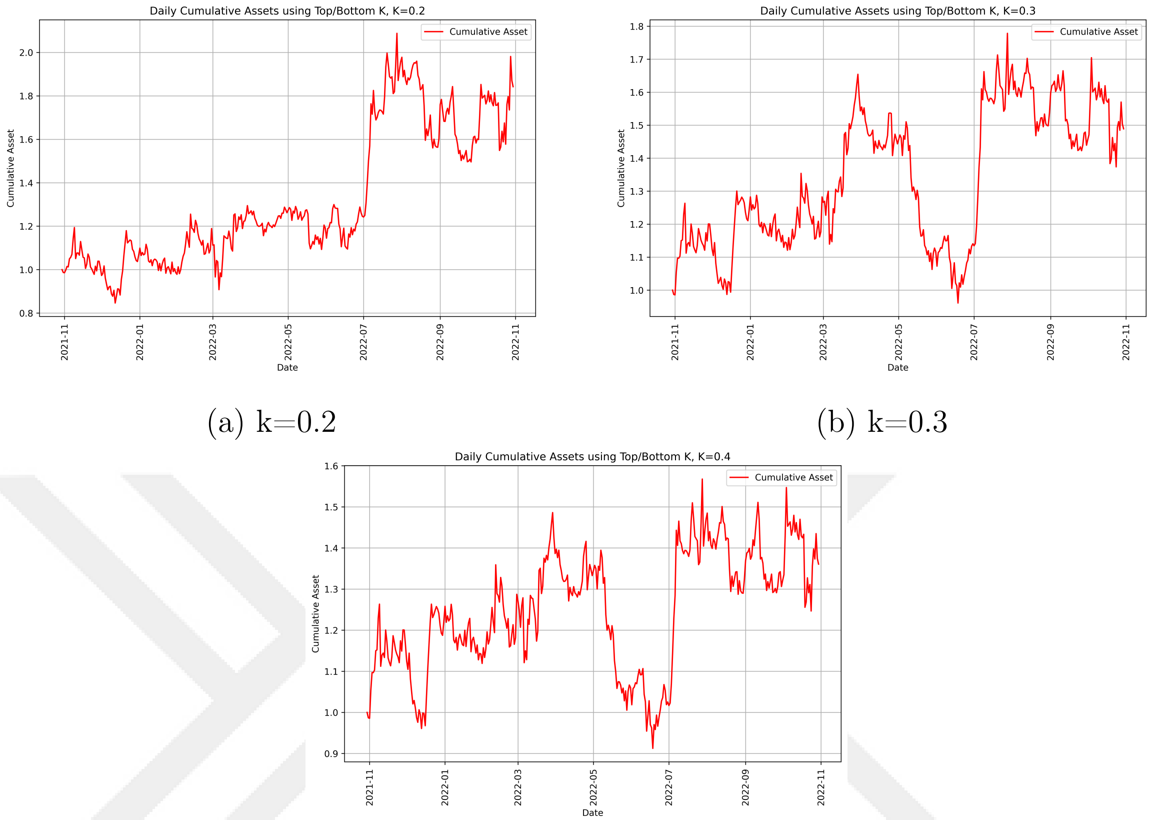
Table 7: Table of K values for Top/Bottom portfolio construction and their corresponding cumulative assets using CNN function approximator.

k	**Cumulative Asset**
0.2	1.84
0.3	1.48
0.4	1.36
Average	**1.56**

for DQN function approximator. In this case, we see an overall positive return as seen in Figure 10 for various k values in a year between 30 October 2021 and 30 October 2022. 1D transformer for DQN function approximator performs better than using CNN for DQN function approximator.

5.5 Post-COVID Performance of Large Market Capitalization Cryptocurrencies

As discussed in Section 3.3, we also focus only on the 18 cryptocurrencies with the largest market capitalization, where we focus on post-COVID era data between 2021-04-30 and 2024-01-13. While executing Top/Bottom-k portfolio, various k values and corresponding cumulative asset values for each cryptocurrency can be seen in Table 8.



(a) k=0.2

(b) k=0.3

(c) k=0.4

Figure 9: Cumulative asset over time by using CNN as DQN function approximator for various Top/Bottom portfolio construction k values.

Similarly, the detailed performance of ReinCrypt for a year period could be seen in Figure 11.

Table 8: Performance of Top/Bottom-k portfolios for training and testing on large market capitalization cryptocurrencies during Post-COVID period.

	Top/Bottom-k			Average
	k=0.2	k=0.3	k=0.4	
Cumulative Asset	0.65	0.76	0.82	0.74

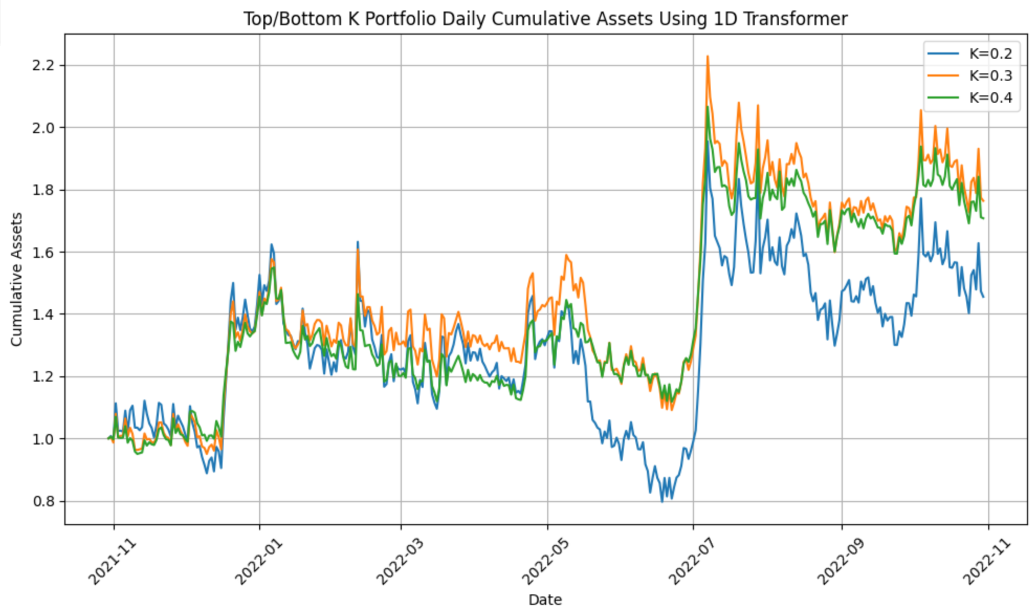
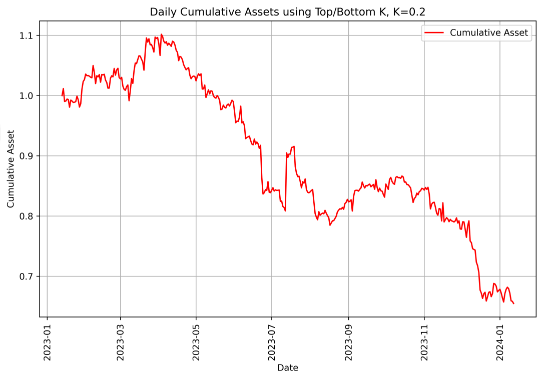
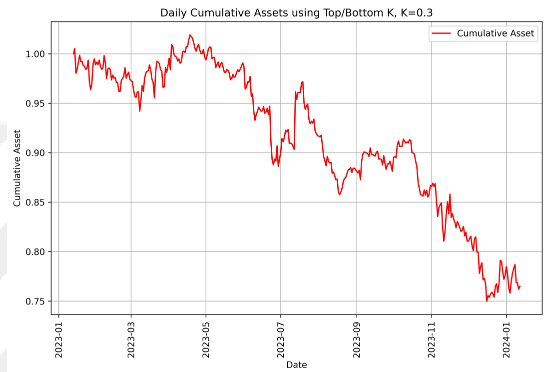


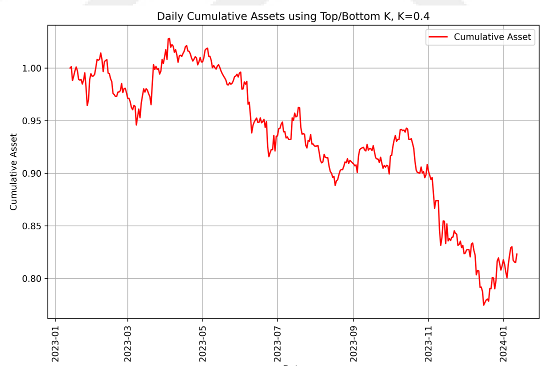
Figure 10: Cumulative asset over time by using Top/Bottom-k Portfolio for 1D transformer.



(a) k=0.2



(b) k=0.3



(c) k=0.4

Figure 11: Cumulative asset over time by ReinCrypt for various Top/Bottom portfolio construction k values when we train and test on large market capitalization cryptocurrencies during Post-COVID period.

CHAPTER VI

CONCLUSION

In this study, our DRL-based method using a vision transformer approximator has shown promising results in predicting cryptocurrency price movements and trends. Our method ReinCrypt outperformed the baseline approaches by successfully anticipating market movements with a higher degree of accuracy. Moreover, our method trained on a diverse set of cryptocurrencies could be used to generate profit on a different set of cryptocurrencies. Similarly, our method trained on highly-traded liquid cryptocurrency such as Bitcoin can be used to generate profit on other alternative cryptocurrencies as well. This result is important since price data for many alternative currencies is only available recently, but Bitcoin data has been available for over 10 years. According to these results, DRL techniques offer a lot more promise for assisting in the forecast of cryptocurrency market movements and may be a useful tool for traders and investors looking to make wise choices in a market that is continually changing.

As a future work, one can incorporate deep reinforcement learning techniques other than DQN such as Proximal Policy Optimization (PPO), Twin Delayed Deep Deterministic policy gradient algorithm (TD3), Double DQN, Asynchronous Advantage Actor-Critic (A3C), etc. into the framework. Similarly, one can also integrate transformer types other than Vision Transformer into our DRL framework. We can investigate the methods to mitigate the effects of volatility for improved forecasting performance, especially in high-volatility markets such as cryptocurrencies.

REFERENCES

[1] S. Freeda, T. C. E. Selvan, and I. Hemanandhini, "Prediction of bitcoin price using deep learning model," in *2021 5th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, pp. 1702–1706, 2021.

[2] M. Ge, S. Zhou, S. Luo, and B. Tian, "3d tensor-based deep learning models for predicting option price," 2021.

[3] W. Zheng, "Exchange-traded fund price prediction based on the deep learning model," in *2021 China Automation Congress (CAC)*, pp. 7429–7434, 2021.

[4] B. G. Malkiel and E. F. Fama, "Efficient capital markets: A review of theory and empirical work*," *The Journal of Finance*, vol. 25, no. 2, pp. 383–417, 1970.

[5] W. F. M. De BONDT and R. THALER, "Does the stock market overreact?," *The Journal of Finance*, vol. 40, no. 3, pp. 793–805, 1985.

[6] N. Jegadeesh, "Evidence of predictable behavior of security returns," *The Journal of Finance*, vol. 45, no. 3, pp. 881–898, 1990.

[7] N. JEGADEESH and S. TITMAN, "Returns to buying winners and selling losers: Implications for stock market efficiency," *The Journal of Finance*, vol. 48, no. 1, pp. 65–91, 1993.

[8] L. K. C. CHAN, N. JEGADEESH, and J. LAKONISHOK, "Momentum strategies," *The Journal of Finance*, vol. 51, no. 5, pp. 1681–1713, 1996.

[9] J. S. Abarbanell and B. J. Bushee, "Fundamental analysis, future earnings, and stock prices," *Journal of Accounting Research*, vol. 35, no. 1, pp. 1–24, 1997.

[10] G. W. Schwert, "Chapter 15 anomalies and market efficiency," in *Financial Markets and Asset Pricing*, vol. 1 of *Handbook of the Economics of Finance*, pp. 939–974, Elsevier, 2003.

[11] M. C. Jensen, "Some anomalous evidence regarding market efficiency," *Journal of Financial Economics*, vol. 6, no. 2, pp. 95–101, 1978.

[12] S. M. Phillips and C. W. Smith, "Trading costs for listed options: The implications for market efficiency," *Journal of Financial Economics*, vol. 8, no. 2, pp. 179–201, 1980.

[13] H. R. Stoll and R. E. Whaley, "Transaction costs and the small firm effect," *Journal of Financial Economics*, vol. 12, no. 1, pp. 57–79, 1983.

[14] A. W. Lo, H. Mamaysky, and J. Wang, "Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation," *The Journal of Finance*, vol. 55, no. 4, pp. 1705–1765, 2000.

[15] C.-H. Park and S. H. Irwin, "What do we know about the profitability of technical analysis?," *Journal of Economic Surveys*, vol. 21, no. 4, pp. 786–826, 2007.

[16] R. P. Schumaker and H. Chen, "Textual analysis of stock market prediction using breaking financial news: The azfin text system," *ACM Trans. Inf. Syst.*, vol. 27, pp. 1–19, mar 2009.

[17] D. Huang, F. Jiang, J. Tu, and G. Zhou, "Investor Sentiment Aligned: A Powerful Predictor of Stock Returns," *The Review of Financial Studies*, vol. 28, pp. 791–837, 10 2014.

[18] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *Journal of Computational Science*, vol. 2, no. 1, pp. 1–8, 2011.

[19] B. G. Malkiel, "The efficient market hypothesis and its critics," *Journal of Economic Perspectives*, vol. 17, pp. 59–82, March 2003.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[23] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[24] M. Lopez de Prado, "The 10 reasons most machine learning funds fail," *The Journal of Portfolio Management*, vol. 44, pp. 120–133, 06 2018.

[25] T. L. Meng and M. Khushi, "Reinforcement learning in financial markets," *Data*, vol. 4, no. 3, 2019.

[26] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming.* John Wiley & Sons, 2014.

[27] S. Chakraborty, "Capturing financial markets to apply deep reinforcement learning," *arXiv preprint arXiv:1907.04373*, 2019.

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[29] E. Bouri, S. J. H. Shahzad, and D. Roubaud, "Co-explosivity in the cryptocurrency market," *Finance Research Letters*, vol. 29, pp. 178–183, 2019.

[30] I. E. Livieris, N. Kiriakidou, S. Stavroyiannis, and P. Pintelas, "An advanced cnn-lstm model for cryptocurrency forecasting," *Electronics*, vol. 10, no. 3, 2021.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[32] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, 1992.

[33] T. Tuncer, U. Kaya, E. Sefer, O. Alacam, and T. Hoser, "Asset price and direction prediction via deep 2d transformer and convolutional neural networks," in *Proceedings of the Third ACM International Conference on AI in Finance*, ICAIF '22, (New York, NY, USA), p. 79–86, Association for Computing Machinery, 2022.

[34] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Applied Soft Computing*, vol. 70, pp. 525–538, 2018.

[35] V. Kochliaridis, E. Kouloumpris, and I. Vlahavas, "Combining deep reinforcement learning with technical analysis and trend monitoring on cryptocurrency markets," *Neural Computing and Applications*, vol. 35, no. 29, pp. 21445–21462, 2023.

[36] J. Lee, R. Kim, Y. Koh, and J. Kang, "Global stock market prediction based on stock chart images using deep q-network," *IEEE Access*, vol. 7, pp. 167260–167277, 2019.

[37] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020.

[38] R. C. Cavalcante, R. C. Brasileiro, V. L. Souza, J. P. Nobrega, and A. L. Oliveira, "Computational intelligence and financial markets: A survey and future directions," *Expert Systems with Applications*, vol. 55, pp. 194–211, 2016.

[39] J.-Z. Wang, J.-J. Wang, Z.-G. Zhang, and S.-P. Guo, "Forecasting stock indices with back propagation neural network," *Expert Systems with Applications*, vol. 38, no. 11, pp. 14346–14355, 2011.

[40] Z. Liao and J. Wang, "Forecasting model of global stock index by stochastic time effective neural network," *Expert Systems with Applications*, vol. 37, no. 1, pp. 834–841, 2010.

[41] A.-S. Chen, M. T. Leung, and H. Daouk, "Application of neural networks to an emerging financial market: forecasting and trading the taiwan stock index," *Computers & Operations Research*, vol. 30, no. 6, pp. 901–923, 2003. Operation Research in Emerging Economics.

[42] E. Guresen, G. Kayakutlu, and T. U. Daim, "Using artificial neural network models in stock market index prediction," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10389–10397, 2011.

[43] O. B. Sezer, A. M. Ozbayoglu, and E. Dogdu, "An artificial neural network-based stock trading system using technical analysis and big data framework," in *Proceedings of the SouthEast Conference*, ACM SE '17, (New York, NY, USA), p. 223–226, Association for Computing Machinery, 2017.

[44] D. Zhang and L. Zhou, "Discovering golden nuggets: data mining in financial application," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 34, no. 4, pp. 513–522, 2004.

[45] S. K. Chalup and A. Mitschele, *Kernel Methods in Finance*, pp. 655–687. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[46] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[47] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, p. 84–90, may 2017.

[48] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.

[49] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Baltimore, Maryland), pp. 655–665, Association for Computational Linguistics, June 2014.

[50] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and

A. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Online), pp. 38–45, Association for Computational Linguistics, Oct. 2020.

[51] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM Comput. Surv.*, dec 2021.

[52] L. Takeuchi, "Applying deep learning to enhance momentum trading strategies in stocks," 2013.

[53] C. Krauss, X. A. Do, and N. Huck, "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the s&p 500," *European Journal of Operational Research*, vol. 259, no. 2, pp. 689–702, 2017.

[54] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018.

[55] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, p. 1735–1780, nov 1997.

[56] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representation and trading," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 3, pp. 653–664, 2017.

[57] A. L. Awad, S. M. Elkaffas, and M. W. Fakhr, "Stock market prediction using deep reinforcement learning," *Applied System Innovation*, vol. 6, no. 6, 2023.

[58] O. Sattarov, A. Muminov, C. W. Lee, H. K. Kang, R. Oh, J. Ahn, H. J. Oh, and H. S. Jeon, "Recommending cryptocurrency trading points with deep reinforcement learning approach," *Applied Sciences*, vol. 10, no. 4, 2020.

[59] A. B. Altuner and Z. H. Kilimci, "A novel deep reinforcement learning based stock price prediction using knowledge graph and community aware sentiments," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 30, no. 4, pp. 1506–1524, 2022.

[60] T. Kabbani and E. Duman, "Deep reinforcement learning approach for trading automation in the stock market," *IEEE Access*, vol. 10, p. 93564–93574, 2022.

[61] J. Zou, J. Lou, B. Wang, and S. Liu, "A novel deep reinforcement learning based automated stock trading system using cascaded lstm networks," *Expert Systems with Applications*, vol. 242, p. 122801, 2024.

[62] S. Sun, R. Wang, and B. An, "Reinforcement learning for quantitative trading," *ACM Trans. Intell. Syst. Technol.*, vol. 14, mar 2023.

[63] Y. Huang, X. Wan, L. Zhang, and X. Lu, "A novel deep reinforcement learning framework with bilstm-attention networks for algorithmic trading," *Expert Systems with Applications*, vol. 240, p. 122581, 2024.

[64] Y. Huang, C. Zhou, K. Cui, and X. Lu, "A multi-agent reinforcement learning framework for optimizing financial trading strategies based on timesnet," *Expert Systems with Applications*, vol. 237, p. 121502, 2024.

[65] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," *arXiv preprint arXiv:2210.02186*, 2022.

[66] Z.-Y. Quan, "Stock prediction by searching similar candlestick charts," in *2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW)*, pp. 322–325, 2013.

[67] S.-J. Guo, F.-C. Hsu, and C.-C. Hung, "Deep candlestick predictor: A framework toward forecasting the price movement from candlestick charts," in *2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, pp. 219–226, 2018.

[68] N. Cohen, T. Balch, and M. Veloso, "Trading via image classification," *CoRR*, vol. abs/1907.10046, 2019.

[69] J.-Z. Huang, W. Huang, and J. Ni, "Predicting bitcoin returns using high-dimensional technical indicators," *The Journal of Finance and Data Science*, vol. 5, no. 3, pp. 140–155, 2019.

[70] D. Mahayana, E. Shan, and M. Fadhl'Abbas, "Deep reinforcement learning to automate cryptocurrency trading," in *2022 12th International Conference on System Engineering and Technology (ICSET)*, pp. 36–41, 2022.

[71] A. Guarino, L. Grilli, D. Santoro, F. Messina, and R. Zaccagnino, "To learn or not to learn? evaluating autonomous, adaptive, automated traders in cryptocurrencies financial bubbles," *Neural Computing and Applications*, vol. 34, no. 23, pp. 20715–20756, 2022.

[72] Q. Zhang, C. Qin, Y. Zhang, F. Bao, C. Zhang, and P. Liu, "Transformer-based attention network for stock movement prediction," *Expert Systems with Applications*, vol. 202, p. 117239, 2022.

[73] Q. Ding, S. Wu, H. Sun, J. Guo, and J. Guo, "Hierarchical multi-scale gaussian transformer for stock movement prediction," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20* (C. Bessiere, ed.), pp. 4640–4646, International Joint Conferences on Artificial Intelligence Organization, 7 2020. Special Track on AI in FinTech.

[74] J. Liu, H. Lin, X. Liu, B. Xu, Y. Ren, Y. Diao, and L. Yang, "Transformer-based capsule network for stock movement prediction," in *Proceedings of the First Workshop on Financial Technology and Natural Language Processing*, (Macao, China), pp. 66–73, Aug. 2019.

[75] N. Parmar, A. Vaswani, J. Uszkoreit, Łukasz Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," 2018.

[76] H. Hu, Z. Zhang, Z. Xie, and S. Lin, "Local relation networks for image recognition," 2019.

[77] B. Enkhsaikhan, "Yahoo finance asset price dataset," 2023.

[78] A. W. Lo, H. Mamaysky, and J. Wang, "Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation," *The Journal of Finance*, vol. 55, no. 4, pp. 1705–1765, 2000.

[79] E. Alpaydin, *Introduction to Machine Learning*. Adaptive Computation and Machine Learning series, MIT Press, 2014.

[80] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Applied Soft Computing*, vol. 97, p. 105524, 2020.

[81] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[82] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[83] W. F. Sharpe, "Mutual fund performance," *The Journal of Business*, vol. 39, no. 1, pp. 119–138, 1966.

VITA

Tuna Alaygut graduated in 2020 with a B.Sc. in Computer Engineering from Yaşar University in İzmir, Turkey. After graduation, he spent three years working as a Software Engineer at Vestel, a large consumer electronics company based in Turkey. In 2021, he started pursuing his M.Sc. in Computer Science at Ozyegin University under the supervision of Dr. Emre Sefer, where he is concentrating on the applications of Deep Q-Networks (DQNs) in the cryptocurrency market.