Animating Stock Markets[*]

Tomasz Kaczmarek[†] Kuntara Pukthuanthong[‡]

August 31, 2023

Abstract

We use Variational Recurrent Neural Networks (VRNNs) to predict stock price trends by translating daily price changes into graphical representations and training the model to forecast future trajectories. We utilize the data from the S&P500 index constituents from 1993 to 2021, with the model exhibiting superior performance in predicting price changes, particularly for large, less predictable stocks. This presents implications for constructing high-liquidity portfolios. While the computationally intensive approach and market conditions specificity present limitations, our study uncovers intriguing opportunities for utilizing advanced machine learning in finance, pushing the frontier in market predictability and portfolio management research.

Keywords: Variational Recurrent Neural Networks, Graphs, Predictability, SP500

JEL Codes: G00, G11, G12

[*]This work was supported in part by the National Science Foundation under Grant No. OAC-1919789.

[†]Tomasz is from the Department of Investment and Financial Markets, Institute of Finance, Poznan University of Economics and Business, Poland. Email: tomasz.kaczmarek@phd.ue.poznan.pl. Polish National Agency partly funded Tomasz's research for Academic Exchange within Bekker Programme, grant number BPN/BEK/2021/1/00404/U/DRAFT/00001, and National Science Centre, Poland, grant number 2021/41/N/HS4/02344. For Open Access, the author has applied a CC-BY public copyright license to any Author Accepted Manuscript (AAM) version arising from this submission.

[‡]Kuntara is from Trulaske College of Business at the University of Missouri, Columbia, MO, USA. Email: pukthuanthongk@missouri.edu

1 Introduction

The endeavor to accurately predict future stock price movements has perennially been at the heart of financial economics. This pursuit, foundational to both trading strategies and risk management, is rendered intricate by the multifaceted and dynamic nature of financial markets. Traditional models, which predominantly rely on numerical time-series data, often grapple with the market's evolving patterns, its pronounced volatility, and its susceptibility to a myriad of external influences. Such challenges underscore the pressing need for a more innovative and adaptable approach to forecasting.

Financial markets are not mere static entities; they pulsate with life, evolving and reacting to a plethora of stimuli. This dynamism is reminiscent of frames in a cinematic reel, where each frame, though a standalone snapshot, is intrinsically linked to its predecessor, painting a broader narrative. Similarly, the stock market of today is a reflection of its yesterdays, and to forecast its tomorrows, one must grasp this sequential relationship. It is here that our research introduces a paradigm shift. By harnessing the power of Variational Recurrent Neural Networks (VRNNs), we aim to predict stock price trends, translating daily price changes into graphical representations and training the model to forecast future trajectories.

To elucidate the significance of this approach, consider the dot-com bubble of the late 1990s and early 2000s. A static snapshot during the bubble's zenith would portray tech stocks as the golden geese of the era. However, a more "animated" perspective reveals a sequence: the mid-1990s rise of the internet, the late 1990s' exponential growth in tech valuations, the bubble's peak around 2000, and its eventual burst in the early 2000s. This sequence, akin to movie frames, provides a holistic understanding of the buildup, climax, and denouement of the bubble.

In finance, researchers have been using machine learning to predict asset returns and measure risk premiums. Such studies include those conducted by Feng, He, and Polson, 2018, Chen, Pelger, and Zhu, 2023, and Gu, Kelly, and Xiu, 2020. However, many of these studies have relied on traditional feed-forward neural networks that only consider features of fixed dimensions and do not consider the longer-term sequential dependency of asset prices and returns. There is a limitation because asset returns are known to follow both short-term and long-term patterns, such as return momentum and reversal patterns. To address this issue, it is important to model the sequential dependence of asset returns when designing strategies that rely on such data. Our study contributes to this literature by exploring the mechanisms and performance of various deep sequence modeling techniques, specifically in estimating the risk premiums of U.S. equities. The same analogy applies to weather forecasts, using frames instead of single graphs (Bi et al., 2023)

In this way, frames form a recurrence (or sequence) in the model that is also a characteristic part of financial markets. That is why predicting financial markets with their dynamics as frames is better than comparing two static graphs (for example, a graph with observed and predicted prices).

Notably, our predicting future graphs are related to predicting prices instead of returns, and our model is very good at predicting prices characterized by high autocorrelation. Figure 2 shows that the next frame prediction task relies on high autocorrelation. Moves of pixels must be strongly autocorrelated to predict the future path, and prices demonstrate the desired characteristics for this model. Predicting prices in financial markets has its distinct advantages and drawbacks. The primary advantage is the inherent autocorrelation present in prices, which is a statistical characteristic meaning that a variable's current value is highly influenced by its past values.

Regarding stock prices, today's price is generally close to yesterday's price, and this trend tends to continue. This autocorrelation creates a certain level of predictability that predictive models can capitalize on, making them potentially more accurate. Moreover, price prediction can provide direct and tangible information to investors and traders who base their decisions on price levels and changes.

On the other hand, predicting prices also has some challenges. One of the main challenges is that price series are generally non-stationary, meaning their statistical properties, such as mean and variance, can change over time. This can make the task of modeling and predicting prices more complex. Furthermore, prices are absolute values that can be influenced by factors such as inflation or company size, which can cause distortions and make predictions less reliable. In contrast, returns, which represent the relative changes in price, are often preferred in financial modeling for their desirable statistical properties. They are typically stationary and normally distributed, making them easier to model and predict. However, return series often exhibit little to no autocorrelation, making them less predictable and harder to forecast accurately.

Predicting prices in the context of our research is significant for many reasons. Firstly, our model's design is deeply rooted in the conceptualization of prices as visual data. By mapping prices onto video frames, our methodology leverages the graphical trajectory of price movements to make future predictions. Hence, focusing on prices is integral to fully capitalizing on the model's predictive potential and the autocorrelation inherent in price data.

Secondly, by predicting prices directly, we capture the direction and magnitude of potential changes. This level of detail can give investors a more granular understanding of future market conditions, facilitating more precise investment decisions. Thirdly, price prediction allows our research to contribute to an important financial forecasting conversation – whether price or return prediction yields more valuable insights. By providing reliable price predictions, we offer fresh evidence of the utility of price-based models, pushing the boundaries of existing financial forecasting methodologies.

Finally, price predictions can be particularly useful in certain investment contexts. For example, options pricing heavily relies on future price forecasts. Likewise, stop-loss orders, limit orders, and similar trading strategies are contingent on price thresholds.

We have calculated our Sharpe ratio (SR) from weekly returns to be 2.94 for equally weighted portfolios and 2.47 for value-weighted portfolios.[1] In comparison, Jiang, Kelly, and

[1]Please note that we are currently in the process of tabulating our results, and they will be available for review in one month.

Xiu, 2020 (2020) report a 1.74 value-weighted SR for all decile stocks, while we achieve a 2.47 Sharpe ratio for S&P500 quintiles. It is important to note that their sample includes publicly traded stocks from NYSE, Nasdaq, and Amex, whereas ours only includes the constituents of the S&P500. We will explain the rationale behind the decision to use this sample next.

Jiang et al., 2020 show higher SR for equally weighted portfolios. However, we are cautious about the practicality of this strategy. Equally weighting a portfolio means investing the same amount of money in each stock, regardless of its market value or size. While this approach can lead to high theoretical returns, it can be challenging to implement in practice, particularly with frequent (weekly) rebalancing. There are two primary challenges associated with this strategy. The first is transaction costs, which can significantly erode returns in practice. The second is liquidity constraints, as smaller companies (included in 'all stocks' in Jiang et al., 2020) may have lower liquidity, making buying or selling the required number of shares difficult.

In contrast, our value-weighted strategy weights investments by the market value of the companies. This approach favors larger, more liquid companies, which can be more easily traded. Therefore, the higher SR achieved by our strategy using S&P500 stocks (large-cap, highly liquid stocks) for quintiles is likely more practical and "investable" in the real world.

Our study introduces a state-of-the-art approach to predicting future US stock prices as video sequences using Variational Recurrent Neural Networks (VRNNs). We apply our model to generate dynamic future frames forecasting stock prices based on a rich data set from the Center for Research in Security Prices (CRSP) for companies listed on the S&P500 index between 1993 and 2021. This methodology harnesses the synergistic strengths of Recurrent Neural Networks (RNNs), Variational Autoencoders (VAEs), and Convolutional Neural Networks (CNNs) to capture both temporal and spatial relationships between price movements.

Complexity in financial market dynamics often cannot be sufficiently captured through simple numerical forecasts. Our methodology's ability to predict more than mere price movements - such as volatility trends and trading volumes - presents a much more holistic view of the market's future. This depth of insight is invaluable in strategic decision-making processes, aiding investors to manage risk and potential returns better.

As the financial industry becomes increasingly data-driven, investment in advanced machine learning capabilities like VRNNs is not merely a cost but an essential strategic asset. These technologies offer a competitive advantage through enhanced forecasting capabilities, enabling firms to anticipate market movements more accurately and act decisively.

Our results demonstrate the VRNN model's capacity to accurately forecast the trajectory of market data changes for the next ten trading days. These forecasts cover closing prices, maximum and minimum prices, the direction of a 20-day moving average, and volumes. This multi-dimensional output offers an enriched prediction of market trends and arms investors with a holistic understanding of the anticipated market performance, empowering them to strategize their investment decisions effectively.

In the intricate world of financial markets, understanding stock price reactions to company announcements is paramount. To deepen our understanding about our approach, we compare ours with the traditional prediction models that employ an expanded window approach, meticulously considering all past stock reactions to predict future movements. For instance, when predicting a stock's reaction on the 30th day, these models would analyze the stock's behavior from the 1st day to the 29th day. While comprehensive, this method treats each day's reaction as an isolated event, potentially overlooking the evolving nature of stock reactions over time.

Imagine a tech company that frequently makes product announcements. Over several months, they unveil new products, update existing ones, and announce various business developments. On the 32nd day, the company drops a bombshell: a merger with another tech giant. Traditional models, with their expanded window approach, might struggle here. They'd look at all past reactions, but without a prior merger announcement as reference, the prediction could be based merely on the average of all past reactions.

Enter the Variational Recurrent Neural Network (VRNN) approach, which maintains a dynamic "memory" of stock reactions. This memory captures evolving patterns, such as the stock's growing positive response to partnership announcements. When faced with the unexpected merger news, the VRNN, informed by its memory, might recognize the merger as a mega-partnership of sorts. Drawing parallels with the stock's strengthening reactions to past partnership announcements, the VRNN could predict a more robust reaction to the merger news.

As another example, Consider a tech stock that, over a span of 10 days, reacts to a series of product announcements with the following percentage changes: [1%, 2%, -1%, 0.5%, 1.5%, 2%, -1.5%, 0.5%, 1.5%, 2.5%]. A discernible pattern emerges: modest gains often precede a slight dip. Using a traditional expanded window approach, to predict the stock's reaction on the 11th day, the model would analyze all previous percentage changes, treating each as an independent data point. The prediction might be a simple average, suggesting a modest gain.

However, a VRNN approach, with its dynamic memory, would recognize the sequence's rhythm. It would "remember" the stock's tendency to dip after a series of gains. Given that the last three days before the 11th showed gains, the VRNN might predict a dip for the 11th day, aligning more closely with the stock's observed behavior. This illustrative example underscores the VRNN's potential to capture intricate temporal patterns in stock price reactions, offering a nuanced lens for financial forecasting.

In this research, we delve into the VRNN's potential to capture the sequential and evolving nature of stock price reactions, offering a fresh perspective in the ever-complex realm of financial forecasting.

In short, comparaing to the other approaches, VRNN stands out in the realm of financial forecasting due to several distinctive features. First and foremost among these is its efficiency. Unlike traditional models that become increasingly computationally intensive as they process more data, VRNNs handle each new data point in the context of a continuously updated hidden state, ensuring streamlined operations even as data sequences grow. Second, this hidden state, often referred to as the model's "dynamic memory," captures the intricate temporal patterns in data, allowing VRNNs to recognize and respond to evolving trends in stock price movements. Third, instead of equally weigh each return, VRNNs inherently prioritize recent data, giving more weight to short-term patterns and relationships. This ensures that the model remains adaptive and responsive to the latest market dynamics. In essence, the VRNN's blend of efficiency, dynamic memory, and emphasis on recent data positions it as a formidable tool for capturing the nuanced rhythms of financial markets.

Our study's contribution to financial economics is multifold. Not only does it introduce a novel VRNN-based methodology for predicting future stock prices, but it also paves the way for the application of advanced machine learning techniques in financial forecasting. The multi-dimensional output of our model aids in making informed investment decisions by offering a rich and nuanced understanding of market trends.

Numerous studies have explored the influence of images, graphs, and colors on investment decisions and stock prices. However, examining graphs and machine learning and their impact on stock prices is still in its early stages. The first paper to utilize machine learning on images was conducted by Obaid and Pukthuanthong (2022). They use CNN to convert images into a sentiment score and demonstrated that photo sentiment is more predictive than text sentiment.

Jiang et al. (2020) is the first paper that uses graphs to predict returns; this, naturally their concept is most closest to ours, though they are not the same. To illustrate, they transform historical price and trading data into two-dimensional images and use CNN to analyze graphs and predict prices based on patterns. We differ in several aspects from their approach.

Our approach involves utilizing VRNNs to predict stock market price movements. We emphasize visual representations in video frames while utilizing image-based machine-learning models. Our VRNN models combine RNNs, VAEs, and CNNs to handle complex sequential data efficiently. We focus on predicting price movements as sequential frames or a "video," instead of using CNN, which only uses static, individual 2D images for prediction as they do. CNN cannot capture the sequential dependencies inherent in financial time series data.

Our predictions are based on the S&P 500 index constituents and prioritize short-term forecasts and high liquidity. In comparison, they utilize data from all companies listed on NYSE, AMEX, and NASDAQ. We do not specify a specific frequency for rebalancing but note that their highest Sharpe ratio is achieved with a five-day rebalancing. Conversely, results for 20-day and 60-day rebalancing are considerably lower.

Our methodology is ideal for investors who frequently rebalance their portfolios and engage actively with their investments. This approach suits those in trading roles that require higher turnover or investors seeking more active involvement. The methodology employed by Jiang et al., 2020 is more suitable for investors interested in medium to long-term strategies as they analyze static 2D images that represent longer-term trends. However, this approach may not provide insights for short-term, quick decision-making.

Regarding liquidity, we focus on S&P 500 index companies, ensuring high liquidity and lower transaction costs. This is particularly advantageous for institutional investors who need to make significant transactions without impacting the market price significantly. On the other hand, their study covers all companies listed on NYSE, AMEX, and NASDAQ, providing broader market coverage and potentially diversifying investment options. However, this comes at the cost of lower liquidity and potentially higher transaction costs, from small stocks.

Our model excels in forecasting accuracy by predicting the direction and scale of price changes, closing prices, maximum and minimum prices, the direction of a 20-day moving average, and volumes. This level of detail provides investors with a comprehensive understanding of future stock performance. The video frames generated by our model depict stock price movements over time, creating a dynamic and informative time-series-like visual output. Our approach gives practitioners a flexibility to consider short-term and medium-term investment strategies based on predicted movements for the next ten trading days.

In comparison, their model predicts the probability of returns to go up and down, lacking the detailed information and accuracy of our dynamic, time-inclusive approach. However, their simpler prediction may be easier for practitioners to interpret and apply, making it more suited for investors focused on a binary decision-making process (buy/sell) based on the predicted price change direction. Their approach may also benefit those implementing high-frequency trading strategies, where the price movement direction is more important than the magnitude of change.

Finally, the main difference lies in the output. They rely on images to forecast the *probability* of returns increasing or decreasing. Their approach involves arranging portfolios according to the return data image from the previous day and evaluating the returns of these portfolios in the following period. Our study, on the other hand, utilizes images to predict future prices up to 10 days in advance.

While both studies use graphical representations in their methodology, the key difference lies in the type of financial variable being predicted (returns vs. prices) and how images are used in the prediction. Jiang et al. used past images of returns to sort portfolios. In contrast, we generate future images of price trends and then decoding these images to predict future prices.

Predicting future prices has the advantage of high autocorrelation, a property not found in returns. This allows your model to make more accurate and detailed predictions potentially. This could have significant implications for market participants interested in short-term price movements. It's a fundamentally different approach with its unique contributions and applications.

It's worth noting that, like all predictive algorithms, our model has certain limitations. However, despite these limitations, it offers a detailed and multi-dimensional view of future market dynamics. The interpretation of the output may be more complex than simpler binary predictions, but our approach is highly effective. While some other methods, such as Jiang et al. (2020)'s methodology which utilizes static 2D images, may be less computationally intensive, our results are focused on large and highly liquid stocks. While our study may not be directly applicable to smaller stocks, the high predictability threshold of these large stocks suggests that our model may perform well with smaller stocks as well. Further research can explore this possibility, but it would require significant computing power. Overall, our model is a reliable and effective tool for predicting future market dynamics.

Despite these constraints, we firmly believe that our work heralds a significant step forward in financial forecasting. It underscores the potential of innovative machine learning methodologies in generating nuanced, visually intuitive, and comprehensive market forecasts, thus adding a robust tool to the arsenal of investors and financial analysts. Our study is a foundation for future research in this exciting and promising arena of graph-based financial predictions.

2 Animated Market Data

The advent of machine learning in finance has opened a gateway for sophisticated and efficient forecasting methods. However, the complexity of financial markets and the multiplicity of factors affecting them raise significant challenges for these methods. Here, we present a novel approach to address this complexity by leveraging animated market data. While stock price changes are typically viewed statically, they exhibit a dynamic nature, akin to frames in a movie, where each frame is closely related to its predecessor. In the same vein, current stock market figures reflect their past performances. Thus, forecasting future price movements hinges significantly on understanding this sequential relationship, which can be effectively captured through animation.

2.1 Multivariate Graphical Data Input

The incredible information capacity of visuals serves as the bedrock of our research. Single graphs can convey a multitude of information simultaneously, making interpretation more accessible. Firstly, a single chart can depict an entire data series related to a specific variable, such as a stock's historical prices over a specified period. Secondly, one chart can encapsulate multiple time-series data simultaneously, like overlaying stock prices, moving averages, and trading volumes. Thirdly, in addition to portraying price data on the day of observation, graphs can present information about their variability, broadening the data spectrum while maintaining clarity. Lastly, the ability to use different colors in graphics makes it easier to discern differences between observation points for each series.

In our study, we harness the power of visuals to capture past closing prices, maximum and minimum prices, 20-day moving averages, and trading volume. We independently generate each image for full control over its content and the optimal arrangement of all elements in the chart. Jiang, Kelly, and Xiu (2020) inspire our chart construction. To our knowledge, we are the first study to generate future graphs.[2] Instead of processing the image to predict the return direction, we incorporate several modifications to enhance the image clarity for machine learning algorithms that will process them later. Figure 1 visualizes the comparison between graphs from this study and those used by Jiang et al. (2020).The left graph proposed by us displays 20 days of historical daily observations and predicts the market data for up to 10 days in the future. Meanwhile, the graph on the right, constructed by Jiang et al., 2020, also uses 20 historical daily observations but predicts the direction of the price for 5, 20, or 60 days. It is important to note that their output is the probability of return direction, while our predictions are prices up to 10 days ahead.

Primarily, closing prices are the most critical elements in charts, whether the predictive task is forecasting returns, price changes, or prices themselves. Hence, we increase the visibility of prices in the chart by modifying the typical OHLC (Open, High, Low, Close)

[2]Jiang et al., 2020predict the probability of stocks to go up and down.

chart, where the closing price is merely a small dot, to a line chart connecting closing prices. This line running through the entire chart is easily perceptible to the human eye and machine learning algorithms.

Secondly, we omit to present opening prices, which do not provide significant predictive value, and in the situation of drawing closing prices as a line, are challenging to overlay. Thirdly, we color-code each of the utilized series. The closing prices are white, the trading volume is light grey, the high-low (HL) bars are darker grey, and the moving average is the darkest grey against a black background. This color differentiation makes it easier to distinguish, for example, between closing prices and the moving average. Importantly, closing prices are always overlaid last, so they remain visible irrespective of other data. The following section discusses further modifications in chart creation that tailor our images to the main objective of our study - capturing market dynamics through image animation.

2.2 Market Dynamics as Video Frames

Stock price changes on day t are conditional on the market information from the n preceding days, much like the content of the last video frame depends on the information in the previous frames. Figure 2 illustrates the analogy between consecutive video frames in the popular testing database "Bair Push Dataset" (Ebert, Finn, Lee, & Levine, 2017) used to evaluate the predictive abilities of machine learning models dedicated to forecasting subsequent video frames and video frames generated by us to predict future stock prices. The changes in the content of each frame are minor, as most of the information repeats in each of them. The difference between the frames induces movement, which becomes apparent when frames are composited into a single image and displayed sequentially over defined time intervals. In the case of frames demonstrating the direction of stock prices, the content change only involves two observations. With each subsequent frame, a new observation appears on the right side of the chart, and the oldest observation disappears. In this manner, consecutive frames demonstrate the dynamics of market price changes.

Figure 3 presents the base chart used to create a video for a single observation. We then divide this chart into frames, each being an image of 64x64 pixels in size, which is the standard dimension in video analysis algorithms. The same figure demonstrates how the base chart is divided into frames, where the first few frames form the context for movement creation, and the subsequent frames become the subject of the forecast.

Three elements of the base chart's construction become essential in using it to create a video for forecasting future stock prices. Firstly, the chart must have an appropriate width. Our single frame with a width of 64 pixels must demonstrate data for specific days, where the number of pixels for each day must be equal. We dedicate four pixels for a single daily observation. This size ensures a smooth line formation connecting individual observations for closing prices and moving averages. The lines would be very jagged with two pixels, three pixels are indivisible by 64, and a larger pixel count than four reduces the range of information on a single frame without providing noticeable benefits in chart smoothness. In this case, a chart with a width of 64 pixels accommodates 16 observations.

When each observation is conditioned on the n preceding ones, the frames must differ by one observation. Hence, in the context of the base chart, expanding the number of displayed observations from 16 to 17 means expanding the chart from 64 to 68 pixels. However, Jiang et al. (2020) demonstrates that 20-day market data predicts future stock returns. To extend the range of input data for the model and to build a sufficient number of input frames with movement context, we use the first five video frames as input data for the model (observed period). In this approach, the first frame provides information within the scope of the first 16 observations (16 observations x four pixels = 64 pixels). The subsequent four frames extend the number of observations to 20 and increase the base chart width to 80 pixels (16 observations x 4 pixels + four observations x four pixels = 80 pixels).

When modeling financial data, each observation is conditioned on the preceding ones, meaning that each frame in our representation must differ by one observation. In the context of our base chart, expanding the number of displayed observations from 16 to 17 corresponds to expanding the chart from 64 to 68 pixels, as each observation is represented by four pixels.

However, Jiang et al. (2020) demonstrate that a 20-day market data window is predictive of future stock returns. To align with this finding and extend the range of input data for our model, we construct our input frames with a movement context in mind.

In our approach, we use the first five video frames as input data for the model during the observed period. The first frame provides information within the scope of the first 16 observations, translating to 64 pixels (16 observations multiplied by four pixels each).

The subsequent four frames do not add four complete sets of 16 observations; instead, they add one observation per frame, extending the total number of observations to 20. This increases the base chart width to 80 pixels, calculated as the original 16 observations multiplied by four pixels each, plus the additional four observations multiplied by four pixels each (16 observations x 4 pixels + 4 observations x 4 pixels = 80 pixels).

This methodology allows us to capture the temporal dynamics of the market, representing the data in a way that reflects both the sequential nature of financial observations and the insights from existing research on market prediction

Our model's design includes a sequence of frames, where each frame represents a specific set of observations (e.g., price observations). The first frame encapsulates the initial 16 observations, represented by 64 pixels. Four subsequent frames each add one observation, extending the total to 80 pixels. These frames correspond to observed data. Additional predictive frames are used for forecasting future observations, with each forecast requiring four pixels on the base chart. The model forecasts ten frames, expanding the base chart to a total width of 120 pixels. This structure allows for both the representation of observed data and sequential forecasting.

The second critical aspect of creating the base chart is the process of input data scaling. First, to eliminate the impact of price jumps associated with stock operations such as splits, we recalculate all chart elements based on the daily rate of returns, which are appropriately corrected by the CRSP (Jiang et al., 2020). Second, how the base chart elements are scaled significantly influences the potential look-ahead bias in the forecasting process.

In our model, we implement a method of chart scaling that considers both price and volume data, based solely on observations from a specific period. This scaling method involves mapping the maximum and minimum prices and the top-level volume to specific pixel levels on the chart.

For illustration, let's assume that the scaling method sets the minimum price at a pixel level of 42 and the maximum at a pixel level of six.[3] If the maximum price during the observed period only reaches a pixel level of 12 instead of six, the algorithm can learn from this pattern. Since the chart must touch both the 42 and six-pixel points in each observation, the algorithm can infer that there should be a price increase to the six-pixel level in the forecast period.

Figure 4 in the document illustrates this scaling method, showing how the data is scaled on the base graph.[4]

The top-right chart is the pure input used to create frames, with dotted lines on the other three graphs marking the extreme values for prices (upper part of the chart) or volume (bottom part).

The potential issue is if we incorporate data from the forecast segment of the chart for scaling purposes, the algorithm could swiftly identify specific patterns. For instance, if the price maximum were to be within the forecast period instead of the observed period, the chart in the observed period would not reach the value typically necessary to achieve the maximum. Such a phenomenon would indicate to the algorithm that the price in the forecast period should attain the maximum value, thereby leading to the look-ahead bias.

[3]In this context, lower pixel levels correspond to higher prices, and higher pixel levels correspond to lower prices. This mapping allows the algorithm to represent price movements within a defined pixel range, facilitating the learning and prediction process.

[4]In our model, we represent financial data as a sequence of frames, with each observation mapped to a specific pixel level on a chart. To facilitate our analysis, we have chosen to map higher prices to lower pixel levels and lower prices to higher pixel levels. This choice allows us to represent price movements within a defined range, aligning with our specific modeling and visualization approach. It is important to note that this mapping is not a conventional practice and is specific to our research methodology. The relationship between pixel levels and price levels is an arbitrary choice that serves our analytical needs, and it is not related to the conventional understanding of pixel resolution in digital images.

In our model, we scale both prices and volumes based on the extremes observed in the input period. This scaling method has an important implication: it can lead to situations where prices or volumes in the forecast period exceed the acceptable range of the chart.

To illustrate, consider the chart area for prices, which is represented between the first and 48th pixel. Within this range, the extremes for prices in the observed period are set at the 6th and 42nd pixels.[5][6] This means that the price can rise or fall by six pixels relative to the extreme in the input period before it reaches its maximum or minimum permissible level on the chart.

However, we have implemented a safeguard to handle situations where the data exceeds this range. If a price or volume does exceed the acceptable range, we represent it by drawing a single pixel at the appropriate extreme on the chart. This approach ensures that the data remains within the defined bounds of the chart, while still allowing some room for additional changes.

Figure 4, specifically the bottom-right panel, illustrates this situation. It shows how the scaling process accommodates extremes in the data, ensuring that the representation remains consistent and meaningful, even when unexpected fluctuations occur.

By scaling prices and volumes according to the observed extremes and implementing this safeguard, we create a flexible yet controlled representation of the data. This approach supports our analysis while accommodating the dynamic and sometimes unpredictable nature

[5]We choose to represent the highest and lowest stock prices using the 6th and 42nd pixel levels, respectively. This choice is based on a careful analysis of the observed range of prices in our dataset, where we map the historical extremes to these specific pixel values. By doing so, we ensure a standardized representation that accommodates the inherent variability of stock prices while maintaining a consistent scale across different stocks and time periods. The choice of these particular pixel levels is not arbitrary but reflects the underlying distribution of prices in our data. It allows us to translate pixel levels back into real-world price levels by using the same scaling factors. This approach supports our analysis by providing a controlled yet flexible way to represent price movements, and it can be adapted to different datasets by recalibrating the scaling factors based on the observed price range.

[6]To determine the range of prices, first we can identify the minimum and maximum prices in the dataset. For example, let's say the minimum price is \$10, and the maximum price is \$100. Next, we determine the range of pixels. In our case, the 6th pixel represents the maximum price, and the 42nd pixel represents the minimum price. Third, we can calculate the scaling factor by Dividing the range of prices by the range of pixels. In this example, the scaling factor would be (\$100 - \$10)/(6 - 42) = \$2.27 per pixel. Lastly, we can convert pixel levels to prices by multiplying the pixel levels by the scaling factor and add the minimum price. In this example, the 6th pixel would correspond to \$100, and the 42nd pixel would correspond to \$10.

of financial markets.

In summary, creating an image as a carrier of information about market states is an exact task where individual elements can significantly impact how the machine learning algorithm learns the dynamics of image changes. In the next section, we demonstrate the requirements behind the machine learning algorithm capable of understanding the market dynamics visualized as a video.

3 Model Selection

This section focuses on the Variational Recurrent Neural Networks (VRNN) model we employ for stock price forecasting. This model is pivotal in our research and serves several simultaneous functions we will elaborate on. This discourse aims to delineate the elements constituting the VRNN model and elucidate why a cohesive understanding of this model is imperative for accomplishing our research task.

Our model needs to fulfill several critical functions concurrently. Firstly, it must analyze data in a time series context. Secondly, it must analyze the image, correctly identifying correlations among its elements. Thirdly, it must be capable of capturing highly uncertain correlations between the stock price at time t and market data from the preceding n observation days.

A vital aspect of our model is its capacity to forecast stock price levels rather than returns. Stock prices exhibit high autocorrelation, a highly desired phenomenon when forecasting subsequent video frames. Consequently, our model deviates from traditional financial models centered around forecasting returns and directly forecasts stock price levels.

To capture the high levels of uncertainty associated with the factors influencing stock price fluctuations, our model incorporates several advanced concepts known in artificial intelligence research. This section succinctly delineates the main tasks and capabilities of each applied algorithm.

At the outset of this section, a crucial point worth mentioning is that we delve into the task of video prediction, a specific manifestation of self-supervision where generative models learn to predict future frames in a video (Devlin, Chang, Lee, & Toutanova, 2018; Gidaris, Singh, & Komodakis, 2018). By undertaking this approach, we aim to establish a firm ground in the predictive modeling landscape.

3.1 Convolutional Neural Network

Convolutional Neural Networks (CNN) form our model's backbone of image analysis, providing an effective means to identify dependencies among individual pixels on a chart. CNNs leverage a mathematical operation known as convolution to scan and process input data, allowing the model to identify and extract significant features from an image.

The fundamental idea behind CNNs involves using multiple layers of convolution, applying a set of learnable filters to the raw pixel data of an image. The convolution operation for a single 2D filter F of size $a \times a$ applied to a part of the image I is defined as:

$$C_{ij} = \sum_{m=0}^{a-1} \sum_{n=0}^{a-1} I_{i+m,j+n} F_{m,n} \tag{1}$$

where $C_{i,j}$ is the convolved feature (or feature map), $I_{i+m,j+n}$ represents a portion of the image I, with the same size as the filter, that pixels' are located at $i + m$ and $j + n$. Finally, $F_{m,n}$ represents the elements of the filer F with m and n indicating the location of the elements within the filter. This process is repeated across all the image regions, effectively sliding the filter across the image, enabling the network to learn spatial hierarchies or patterns.

3.2 Recurrent Neural Networks

Stock price forecasting at time t based on market data from the previous n observations inherently entails time series analysis. Predicting sequences is a natural task for Recurrent Neural Networks (RNNs), a class of artificial neural networks designed to recognize patterns in data sequences.

RNNs can uniquely retain information from prior inputs in their hidden states, thereby modeling temporal dependencies. A basic form of an RNN can be represented as:

$$h_t = \sigma \left(W_{hh} h_{t-1} + W_{xh} x_t + b_h \right) \tag{2}$$

where h_t is the hidden state at time t, x_t is the input at time t, W_{xh} , and W_{hh} are weights, b_h is a bias, and σ is an activation function.

While models based on the combination of CNNs with a recurrent model, such as the Convolutional LSTM Network (Shi et al., 2015), are applicable for analyzing sequences of image frames, traditional RNNs fall short when accounting for forecast uncertainty due to their deterministic nature. Their only source of variability resides in the conditional output probability model, which is insufficient for capturing the randomness intrinsic to the data (Babaeizadeh, Finn, Erhan, Campbell, & Levine, 2017).

As demonstrated in Figure 5, typical probabilistic models, after applying the uncertainty, generate an average or "shadow" of various potential scenarios when forecasting movements for phenomena laden with high uncertainty. While indicating the most probable trajectory, these predictions are imprecise and thus inadequate for forecasting stock prices. This shortcoming necessitates the deployment of algorithms capable of predicting phenomena with high levels of uncertainty. The following sections discuss the implementation of such models, focusing on integrating Variational Autoencoders into RNN structures.

3.3 Variational Autoencoders

To enhance the efficacy of image series analysis, employing data compression of the images using autoencoders is practical. An autoencoder is a neural network architecture that consists of two main components: an encoder and a decoder. The encoder compresses high-dimensional image data into a compact space known as a bottleneck, and the decoder reconstructs the original data from this compressed representation (refer to Figure 6).

The compression is achieved by minimizing the reconstruction loss, which is the difference between the original input image and the output image generated by the decoder. A simple autoencoder utilizes fully connected layers of a neural network, but autoencoder layers can also encompass Convolutional Neural Network (CNN) cells, facilitating the compression of image or sound data.

However, traditional autoencoders map input data to a single vector, rendering them deterministic. This can be a limitation when modeling data with inherent variability. Non-deterministic models that map data onto a distribution are employed to address this. These models, known as Variational Autoencoders (VAEs), utilize two vectors—one for the mean and another for the standard deviation of the feature distribution (Kingma & Welling, 2013; Rezende, Mohamed, & Wierstra, 2014).

VAEs are a potent example of deep generational probabilistic graphical models adept at capturing input data variability and generating a distribution that summarizes this variability (refer to Figure 7). They offer a harmonious blend of flexible non-linear mapping between latent random states, observed outputs, and effective approximate inference. This combination enables VAEs to model complex multimodal distributions, which are beneficial when the underlying true data distribution comprises multimodal conditional distributions.

The underlying principle of a VAE is rooted in Bayesian inference and can be expressed as follows:

$$q_\phi(z \mid x) = \mathcal{N}\left(z \mid \mu_\phi(x), \sigma^2_\phi(x) I\right) \tag{3}$$

where $q_\phi(z \mid x)$ is the approximate posterior (encoder), ϕ are the parameters of the encoder, z is the latent variable, and x is the observed data. The parameters $\mu_\phi(x)$ and $\sigma^2_\phi(x)$ of the Gaussian distribution are outputs of the encoder. The variational part of the VAE concerns minimizing the divergence between the true and approximate posterior.

Thanks to these modifications, VAEs can non-deterministically compress data and estimate various future states for each sample of their latent variables. The properties of VAEs will be leveraged in the next section, which discusses the incorporation of VAEs into the architecture of the Variational Recurrent Neural Network (VRNN).

3.4 Variational Recurrent Neural Network

The architecture of the Variational Recurrent Neural Network (VRNN) combines the temporal modeling capabilities of RNNs with the probabilistic latent variable modeling capabilities of VAEs, thereby enabling the creation of a powerful predictive model for complex and uncertain processes (Chung et al., 2015). At its core, the VRNN model embeds a VAE in each of the recurrent cells of the RNN.

This design allows the VRNN to handle the high uncertainty inherent in video frame prediction tasks. The variational component of the VRNN enables the generation of multiple plausible trajectories for object movement depicted in video frames, as opposed to a single deterministic trajectory.

VRNN is formulated with an architecture that closely intertwines the hidden states of the RNN (h_t) with the latent variables of the VAE (z_t) as shown by the following equations: The prior:

$$p_\theta\left(z_t \mid h_{t-1}\right) = \mathcal{N}\left(z_t \mid \mu_\theta^{(p)}\left(h_{t-1}\right), \sigma_\theta^{(p)}\left(h_{t-1}\right)\right) \tag{4}$$

The approximate posterior:

$$q_\phi\left(z_t \mid x_t, h_{t-1}\right) = \mathcal{N}\left(z_t \mid \mu_\phi^{(q)}\left(x_t, h_{t-1}\right), \sigma_\phi^{(q)}\left(x_t, h_{t-1}\right)\right) \tag{5}$$

The observation model:

$$p_\theta\left(x_t \mid h_t\right) = \mathcal{N}\left(x_t \mid \mu_\theta^{(x)}\left(h_t\right), \sigma_\theta^{(x)}\left(h_t\right)\right) \tag{6}$$

The hidden state update:

$$h_t = f_\theta\left(h_{t-1}, z_t, x_t\right) \tag{7}$$

In the equations above, x_t denotes the input at time t, z_t is the latent variable, and h_t is the hidden state. The parameters θ and ϕ represent the network parameters for the generative and inference model, respectively. The function f_θ corresponds to the deterministic transition function of the RNN, which is usually a non-linear function such as a Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) or Gated Recurrent Unit (GRU) (Cho et al., 2014).

Figure 8 presents a single cell of the VRNN model encapsulated within a dark red rectangle. This VRNN cell operates on the principles of a recurrent network, transforming the prior hidden state h_{t-1} into the current hidden state h_t. This h_t can be used in the following cell, establishing a temporal link across cells.

Inside the VRNN cell is a VAE that operates on the combined data of the prior hidden state h_{t-1} and the new frame x_t. It compresses this high-dimensional information using an encoder and a decoder that are equipped with CNN layers, facilitating pattern recognition in the input images.

In the VAE, the data is condensed into a latent vector z_t through a process based on two key elements: the mean vector (μ) and the variance vector (σ). This process can be seen in the Equation 5, where the input data x_t and the prior hidden state h_{t-1} are used to estimate the true distribution of the latent variable z_t.

The decoder part of the VAE relates to the Equation 4 and Equation 6, generating new frame data x_t from the hidden state h_t, and computing the distribution of the actual data. The deviation between this generated data and the actual data is termed the reconstruction error, which is minimized during training.

Finally, the VRNN cell uses a recursive function that integrates the information from the prior hidden state h_{t-1}, the current latent state z_t, and the new frame x_t. This process is represented by the Equation 7, which updates the hidden state h_t based on the previous hidden state, the current latent state, and the new input frame.

In conclusion, the diagram exhibits how the VRNN model leverages the strengths of

RNNs, VAEs, and CNNs to efficiently handle complex, sequential data. It illustrates how each component - prior, approximate posterior, observation model, and hidden state update - contributes to the overall functioning of the model.

In conclusion, we demonstrate the connection between the RNN and VAE in the VRNN architecture, presenting a comprehensive understanding of how the non-deterministic nature of the VAE is integrated into the RNN, thereby enhancing its capacity to handle high-uncertainty scenarios. The next section demonstrates the application of VRNN to finance.

4 Generating Future US Stock Prices as Video

In this chapter, we delve into our methodology, starting with a description of the data used in the study. Then we explain the procedure for training the VRNN model for generating future frames. Further, we clarify how to read and interpret the data on the generated video frames. Finally, we present our results, beginning with the probability of determining the right direction of price changes, to showcase our model's economic applications in portfolio construction.

4.1 Data

In our research, we utilize daily data from the Center for Research in Security Prices (CRSP) for constituents of the S&P500 index. Historical compositions of companies included in the index are acquired from Refinitiv. Our research sample pertains to the period from 1993 to 2021. Market data concerning the highest and lowest prices are available in CRSP since June 1992.

We focus on S&P500 index companies for several crucial reasons. First, our research involves forecasting daily stock price changes in the short term. Utilizing short-term forecasts to construct an investment portfolio necessitates frequent rebalancing and concurrent sufficient liquidity of analyzed companies. Concentrating on the largest U.S. companies in the S&P 500 ensures high liquidity and low transaction costs. This makes the results of our research applicable to investors with high assets. See Table 1 for the number of firms in our sample in each year.

Second, a well-documented phenomenon is the decidedly lower predictability of the largest companies stock prices.[7] Hence, by concentrating on the largest stocks, we demonstrate that our strategy achieves solid results amongst companies for which predicting the direction of price changes is most challenging.

Finally, the machine learning task of the next frame forecasting requires significant computational resources. A smaller number of companies in the analysis considerably accelerates calculations, making our study feasible even with commonly available computing equipment.

4.2 Training VRNN

Our task is to forecast the trajectory of market data changes for the next ten trading days, using a method that involves both motion context and video frames. As mentioned in Section 2, we utilize five 'context frames,' which define the input data and signal the direction of object movement found on individual frames. These context frames encapsulate information from 20 trading days of historical market data, approximating one month (from t-19 to t), and predict 10 trading days of future market data that approximate two weeks (from $t+1$ to $t+10$). These changes encompass closing prices, maximum and minimum prices, the direction of a 20-day moving average, and volumes. By utilizing data other than closing prices, we aim to enhance the forecast capabilities related to future changes in the closing price.

To train the model, we generate a total of 15 frames, comprising five context frames and ten forecast frames. This allows the model to independently learn the relationships between the context and forecast frames.

[7] Jiang et al. (2020) in their research use all companies listed on NYSE, AMEX, and NASDAQ. They show that the out-of-sample Sharpe ratio of strategies based on H-L decile spread portfolios sorted by image-based return is highest with a five-day rebalancing. Results for 20-day and 60-day rebalancing are significantly lower, and a Sharpe ratio above one is achievable only for equally-weighted strategies. Value-weighted strategies reach a Sharpe ratio above one only with a weekly (five-day) rebalancing.

Our model's design involves a base chart with a 64x120 pixel resolution, representing different dimensions or features of the market data, such as price and volume. We divide this chart into 15 parts, corresponding to five context frames (capturing 20 trading days of historical data) and ten forecast frames (predicting the next ten trading days). The choice of 15 parts and the specific resolution aligns with our model's architecture and the nature of the data.

We employ a sliding window method with a 64x64 resolution, moving the window by four pixels at a time across the base chart. This technique allows us to analyze sequential segments of the data, capturing temporal patterns and relationships.

Each observation in our model corresponds to the last trading day of a calendar week. While our data includes daily closing prices, we may have chosen this weekly reference point to align with other data sources, reduce noise, or capture specific market patterns relevant to our study.

Generative models, such as VRNNs, utilize specific measures to determine the degree of fit between the generated image and the actual one. One such measure is the "loss," often called the reconstruction loss (loss_rec). This loss quantifies the discrepancy between the actual data and the reconstructed data generated by the model. The smaller the reconstruction loss, the more closely the generated data matches the actual data, indicating a better-performing model.

The second measure, "prior loss" (loss_prior), deals with regularizing the latent variables in the VRNN. In simple terms, it ensures that the distribution of the latent variables follows a specific pattern, often a standard Gaussian distribution. By minimizing the prior loss, we encourage the latent variables to conform to this pattern, which can improve the model's generative abilities.

Finally, the total "loss" of the VRNN model is usually a combination of these two components, the reconstruction, and prior losses. By minimizing these two components jointly, we aim to train a model that can effectively generate data similar to the training set while maintaining a regular structure in the latent space. Therefore, t his loss function plays a crucial role in training our VRNN model and, subsequently, the quality of the predictions it generates.

Our study divides the data into a training period (1992-2000) and a testing period (2001-2021). We train the model once on the training data, setting aside several hundred of the last observations from the training set as a validation sample to measure the model's training level. Based on the total "loss" in the validation sample and the early stopping approach, we determine the number of epochs necessary to train the model.

All of our calculations are conducted using Python. We use PyTorch package and perform demanding computations at the Foundry and Hellbender high-power computing environments at the University of Missouri.

4.3 Reading the Market Data from Video Frames

The output of our model consists of predicted frames. It is crucial to accurately read the data from these frames to conduct accuracy tests and apply these forecasts to portfolio construction. In this process, we examine each generated frame and read pixels from the relevant columns of the images, which correspond to the stock prices. We only search for white pixels, as closing prices are denoted in white. For instance, on the first of the generated frames, the first column of pixels on the right contains information about the forecasted closing price on day $t+1$.

Subsequently, on the second of the generated frames, the fifth column of pixels from the right contains data for the same observation as the previous frame from day $t+1$ but also includes an additional forecast in the first column from the right with the closing price from day $t+2$. Proceeding this manner, we arrive at the tenth generated frame, which contains forecasted prices for days $t+1$ to $t+10$. We average all readings for the same days to eliminate potential noise or vagueness in the forecasts. The information read on the pixel's position forecasting the price on day $t+n$ can be compared with the reading of information about the pixel on day t. This comparison provides the simplest yet effective way to verify whether the model has predicted a price increase, stability, or decrease for a specified number of days ahead. However, not only the direction of changes can be significant, but also its scale. Therefore, forecasts differing by a larger (or smaller) number of pixels can be more (or less) important.

Finally, the number of days n utilized in forecasting weekly price changes is also important. The maximum number of session days in a single week is five. Nevertheless, some weeks have fewer days. For this purpose, for each day t, representing the last session date in a given calendar week, we count the number of session days in the following calendar week. This calculation determines the number of session days in the next week. It serves as a reference point for choosing the appropriate length of the forecast from the data generated by the VRNN.

4.4 Accuracy of Forecasted Price Direction

Table 2 serves as a critical component of our analysis, providing a detailed breakdown of the model's predictive performance across different horizons. The table is divided into two main sections: the top part, which includes the unclear rate, and the bottom part, which excludes it. Both sections are further categorized into "Correction," "Uncorrection," and "Unclear."

Correction represents the accuracy of the model's predictions, where the model correctly predicted the market direction. For example, the correction rate for direction1 is 53.33%, indicating a moderate level of accuracy for one-day-ahead predictions. Uncorrection includes instances where the model's predictions were incorrect. The uncorrection rates range from 26.67% for direction1 to 33.33% for direction10, reflecting the model's errors. Unclear encompasses situations where the model's predictions were ambiguous or inconclusive. The unclear rates range from 20% for direction1 to a significant jump to 40% for direction10. "Unclear" could arise from the model's inability to discern a clear pattern or trend in the data, possibly due to noise, volatility, or other complex factors. The top part of the table, including the unclear rate, provides a comprehensive view of the model's performance, considering all possible outcomes. It helps us understand not only where the model is correct or incorrect but also where it struggles to make a definitive prediction.

The bottom part of the table, excluding the unclear rate, offers a more focused view of the model's performance by considering only the clear predictions. It allows us to assess the model's accuracy when it is confident in its predictions.

The results in Table 2 reveal some key insights. The correct and incorrect rates are similar for all across the length of the predicted period in days. Our correction rate is comparable with that of Jiang et al., 2020.

While the correction rates for shorter horizons are promising, the decrease in accuracy and increase in unclear predictions for longer horizons highlight areas for improvement. The results call for further investigation into the model's architecture, data preprocessing, and feature engineering to enhance its ability to generalize across different prediction horizons.

In conclusion, Table 2 serves as a valuable tool in assessing the model's predictive capabilities, strengths, and weaknesses. By presenting both the top and bottom parts of the table, we offer a nuanced view of the model's performance, considering both clear and unclear predictions. The insights gained from this analysis will guide our ongoing efforts to refine the model, with the ultimate goal of achieving robust and reliable predictions for various market scenarios.

There are some limitations of this table. We do not report whether there is an asymmetric effect between negative and positive prediction. The correction presents the same direction of prediction for the actual and predicted value. Second, we do not consider the magnitude of the difference between actual and prediction. Although the model presents the correction that is greater than the incorrect one, the margin is narrow between 12% to 16%.

4.5 Portfolio Performance

We construct value-weighted (VW) and equal-weighted (EW) portfolios for the five quintiles. The returns and Sharpe ratios increase across quintiles. The long-short portfolios of our strategy (VRM) generate the best performance. See Table 3. We compare our performance with that of non-machine learning strategies including momentum (MOM), short-term reversal (ST_REV) and long-term reversal (LT_REV). Our Sharpe ratio and Calmar ratios for both equal-weighted and value-weighted portfolios are higher than those of non-machine learning based trading strategies. VRM has 2.47 and 2.94 for value-weighted and equal-weighted portfolios; while, MOM has SR of 0.05 and 0.01, ST_REV of 0.24 and 0.51, and LT_REV of 0.05 and 0.30.

5 Any title 5

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

6 Any title 6

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

7 Conclusion

In conclusion, this research presents a novel approach to the prediction of stock price movements by using advanced machine learning techniques, particularly Variational Recurrent Neural Networks (VRNNs). By transforming daily price changes into visual representations and forecasting future trajectories, we've shown that our model delivers superior performance, especially for large, less predictable stocks. This approach holds significant implications for portfolio construction, particularly regarding liquidity and rebalancing strategies.

Our research demonstrates the potential of machine learning in augmenting the understanding of financial market dynamics. However, it also acknowledges the limitations of this approach, including its computationally intensive nature and potential specificity to the market conditions of the training dataset.

Moving forward, it's clear that the fusion of machine learning and finance research can lead to significant advancements in understanding and predicting market trends. As technology evolves and computational resources become more accessible, adopting and refining such methods in finance are likely to continue and provide further insights into market dynamics. This study's results indicate promising future research directions, particularly in exploring the intricacies of market behavior using innovative computational methods.

We hope that this study sparks further interest and research in this field, enabling more sophisticated models that can better capture the intricacies of financial markets and provide investors with superior tools for navigating the complexities of the market.

References

Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R., & Levine, S. (2017). Stochastic Variational Video Prediction. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*. arXiv: 1710.11252

Bi, K., Xie, L., Zhang, H., Chen, X., Gu, X., & Tian, Q. (2023). Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 1–6.

Chen, L., Pelger, M., & Zhu, J. (2023). Deep learning in asset pricing. *Management Science*.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Emnlp 2014 - 2014 conference on empirical methods in natural language processing, proceedings of the conference* (pp. 1724–1734). doi:10.3115/v1/d14-1179. arXiv: 1406.1078

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., & Bengio, Y. (2015). A Recurrent Latent Variable Model for Sequential Data. *Advances in Neural Information Processing Systems*, *28*

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*, 4171–4186. arXiv: 1810.04805

Ebert, F., Finn, C., Lee, A. X., & Levine, S. (2017). Self-supervised visual planning with temporal skip connections. arXiv: 1710.05268 `[cs.RO]`

Feng, G., He, J., & Polson, N. G. (2018). Deep learning for predicting asset returns. *arXiv preprint arXiv:1804.09314*.

Gidaris, S., Singh, P., & Komodakis, N. (2018). Unsupervised Representation Learning by Predicting Image Rotations. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*. arXiv: 1803.07728

Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, *33*(5), 2223–2273.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. doi:10.1162/neco.1997.9.8.1735

Jiang, J., Kelly, B. T., & Xiu, D. (2020). (re-)imag(in)ing price trends. *Capital Markets: Asset Pricing & Valuation eJournal*.

Kingma, D. P., & Welling, M. (2013). Auto-Encoding Variational Bayes. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*. arXiv: 1312.6114

Obaid, K., & Pukthuanthong, K. (2022). A picture is worth a thousand words: Measuring investor sentiment by combining machine learning and photos from news. *Journal of Financial Economics*, *144*(1), 273–297.

Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *31st International Conference on Machine Learning, ICML 2014*, *4*, 3057–3070. arXiv: 1401.4082

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., Woo, W.-C., & Kong Observatory, H. (2015). Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. *Advances in Neural Information Processing Systems*, *28*.

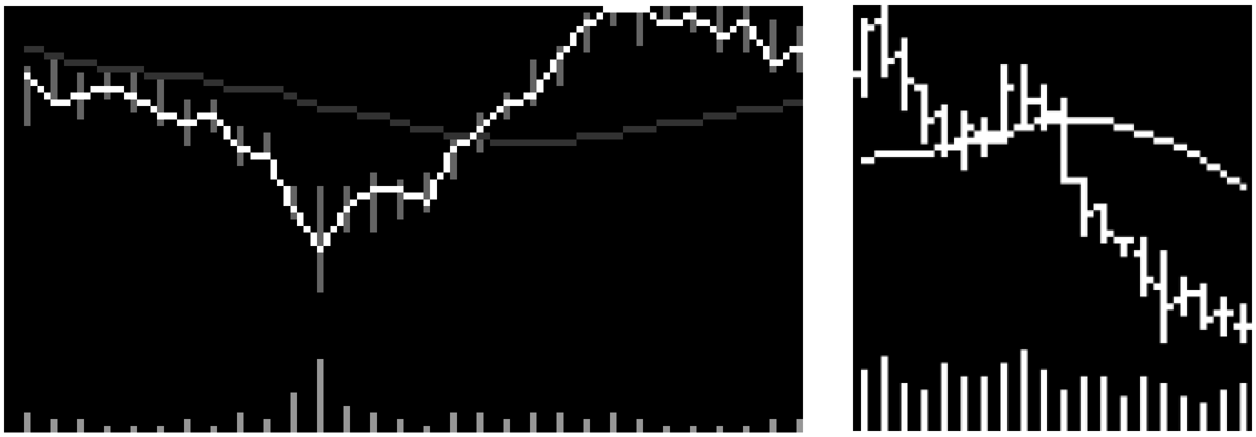

Figure 1: Graphs comparison

In this figure, we show a comparison between two graphs. The graph on the left is the base graph that we use to create frames for animations that, with 20 days historical daily observation predict up to 10 days of future market data. On the right, we show a graph prepared by Jiang, Kelly, and Xiu (2020) where 20 historical daily observations are used to predict the price direction in 5, 20, or 60 days.

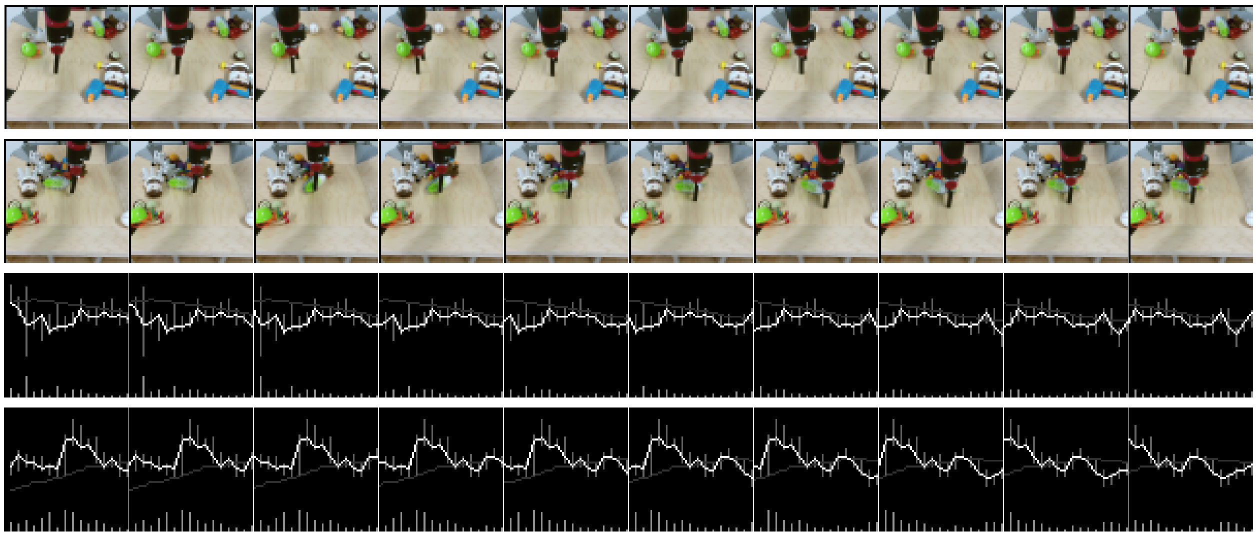

Figure 2: Sample video frames

This figure presents sample frames of four videos. Each video consists of ten frames. Two upper rows demonstrate frames from video database BAIR Robot Pushing frequently used by researchers to test algorithms dedicated to next frame forecasting task (Ebert, Finn, Lee, & Levine, 2017). Two bottom rows show frames from graphs created in this research.

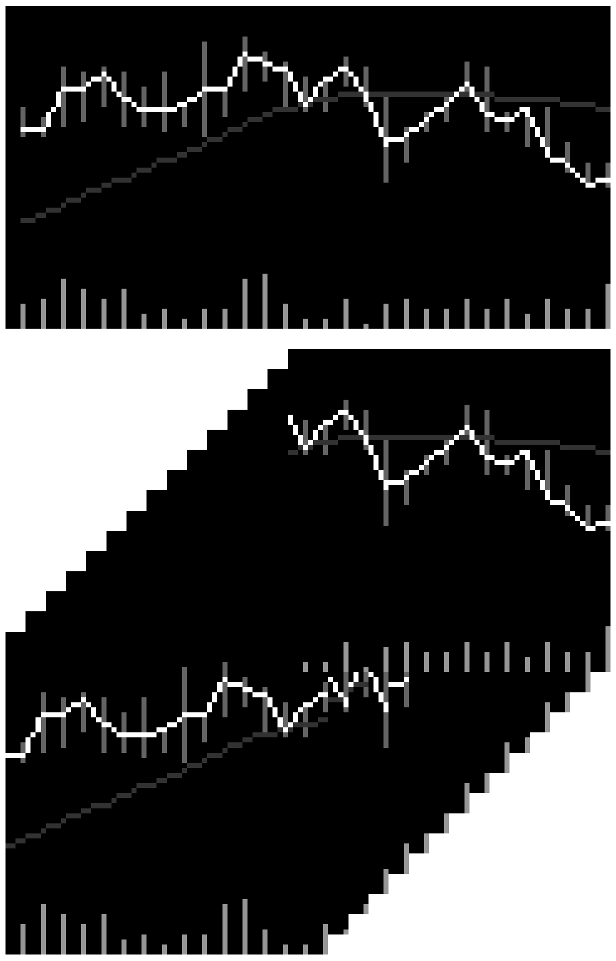

Figure 3: Frames of a Video from a Base Chart

This figure demonstrates the transformation of a base chart into frames. The base chart (at the top of the figure) has dimension 64x120 pixels. It is separated into 15 frames (at the bottom of the figure) with dimension 64x64 pixels. The first frame and the last are demonstrated in the front of others.

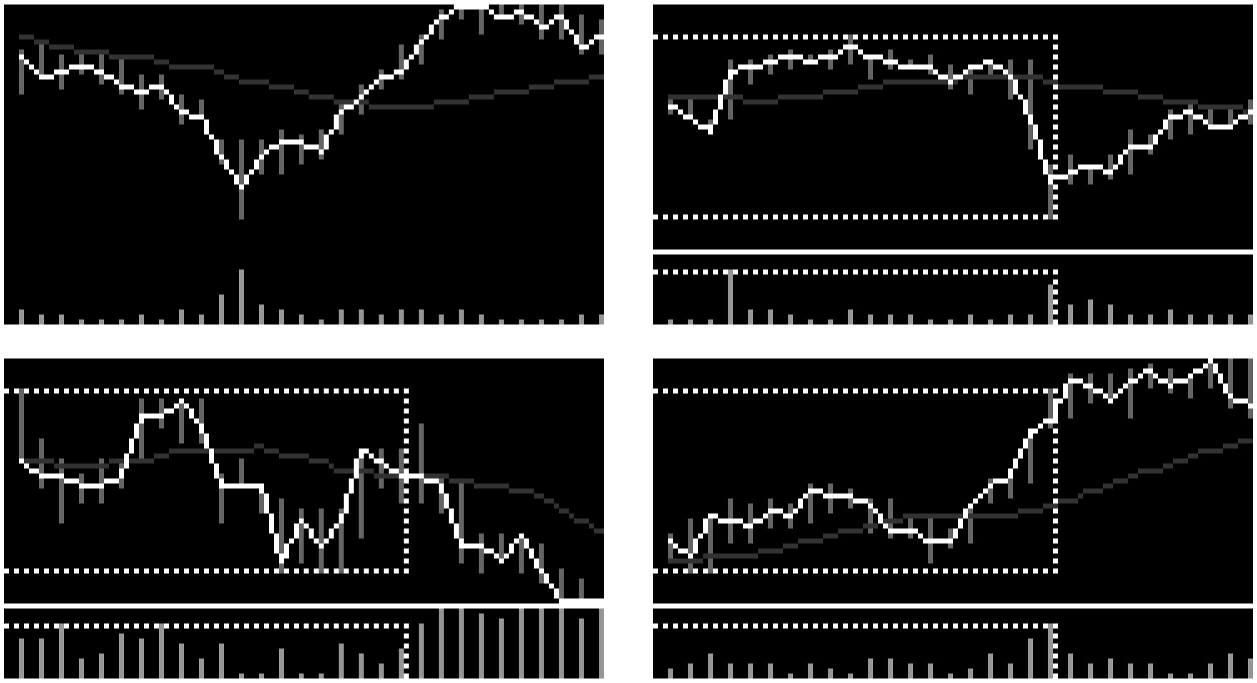

Figure 4: The data scaling procedure

This figure demonstrates how we scale the data on the base graph. We present four base graphs that are used to form video frames. The top-right chart is a pure input to create frames. On the other three graphs, we add, for information purposes, dotted lines that demonstrate the borders forming extreme values for prices (upper part of the chart) or volume (bottom part).

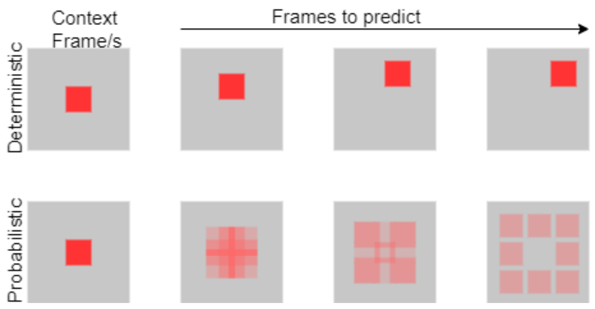

Figure 5: Generative Outcome of Deterministic vs. Probabilistic Model

The top part demonstrates frames of deterministic model where a box moves in a random direction. The bottom part, shows a probabilistic outcome. By introducting the uncertainty the generated objects are blurry and averaged. Figure inspired by Babaeizadeh, Finn, Erhan, Campbell, and Levine (2017).

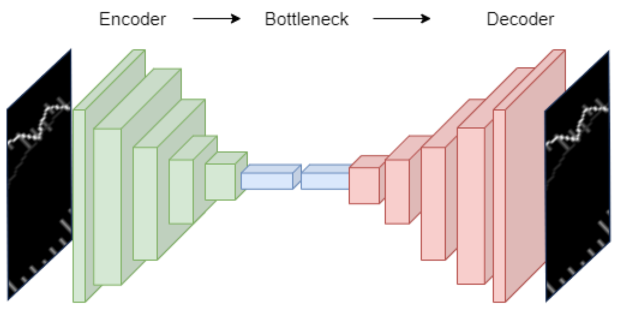

Figure 6: Autoencoder

In autoencoder the input data is compressed with encoder to bottleneck and then it is decompressed with decoder. In the training process the algorithm's task it to minimize the comparison error between the original input and the decoded output.

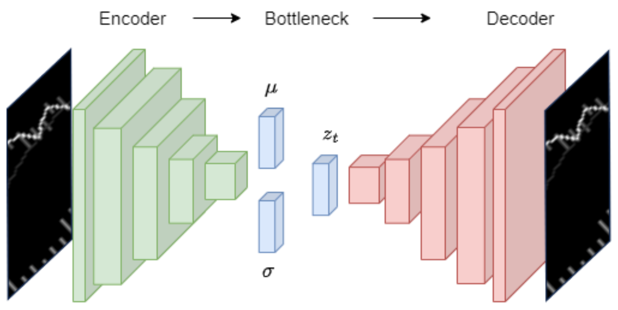

Figure 7: Variational Autoencoder (VAE)

Variational autoencoder is a type of autoencoder where encoder takes input and output two vectors, one with the means and the other with variance. Next, mean and variance vector are used to create a sampled vector. The decoder reconstruct input from the sampled vector.

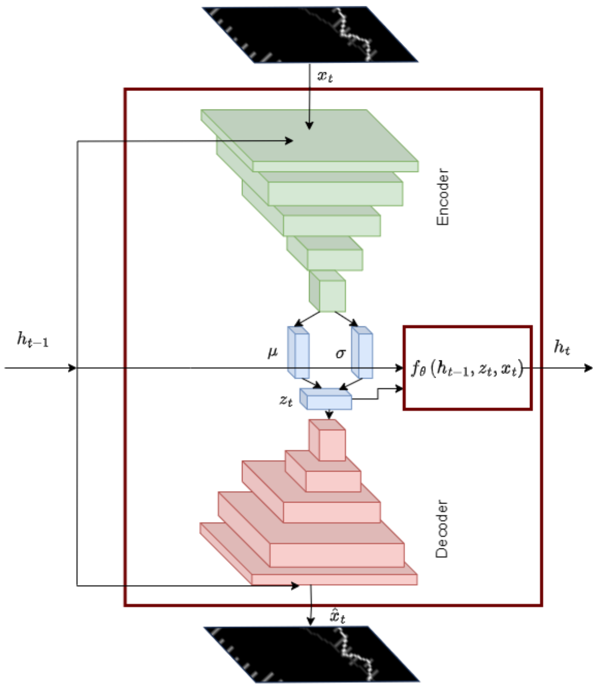

Figure 8: Variational Recurrent Neural Network (VRNN)

The figure illustrates a single Variational Recurrent Neural Network (VRNN) cell. In VRNN each cell contains a Variational Autoencoder (VAE) with Convolutional Neural Network (CNN) layers, which condenses high-dimensional data into a low-dimensional latent vector. The prior hidden state (h_{t-1}), current latent state (z_t), and new input frame (x_t) are processed through a recursive function to yield the current hidden state (h_t).

Table 1: Descriptive Statistics

The table presents an overview of the research sample (SAMPLE) vs. S&P500 Index (S&P500) in yearly periods. For stocks in the research sample, it demonstrates equal-weigh average weekly returns (RetEw), value-weight average weekly returns (RetVw), total market capitalization in millions of dollars (Mcap), and number of stocks (No). The data for S&P Index covers market capitalization in millions of dollars (Mcap).

	Sample				**S&P500**
year	RetEw	RetVw	Mcap	No	Mcap
2001	0.18	0.13	8,737	444	10,731
2002	-0.32	-0.28	7,459	438	9,143
2003	0.70	0.59	7,322	436	8,900
2004	0.35	0.29	8,551	439	10,522
2005	0.18	0.19	9,084	441	11,060
2006	0.31	0.38	9,747	449	11,849
2007	0.03	0.20	10,871	451	13,141
2008	-0.81	-0.50	8,924	457	10,666
2009	0.92	0.75	7,096	449	8,295
2010	0.43	0.38	8,578	442	10,288
2011	0.07	0.18	9,514	443	11,554
2012	0.28	0.32	10,248	438	12,445
2013	0.67	0.64	11,999	433	14,664
2014	0.31	0.37	14,033	430	17,197
2015	-0.04	0.08	14,772	439	18,217
2016	0.31	0.32	14,699	437	18,197
2017	0.38	0.47	16,938	443	21,027
2018	-0.19	-0.05	18,699	438	23,293
2019	0.54	0.64	19,511	431	24,315
2020	0.43	0.61	21,820	440	26,622
2021	0.51	0.58	28,733	440	36,176

Table 2: Accuracy of Predicted Price Direction

This table reports the out-of-sample accuracy of price change direction depending on the length of the predicted period in days. We define the positive direction as not smaller than zero and the negative direction as less than zero. The correct predictions state that the model predicts positive (negative) price change when the price change is positive (negative), and incorrect demonstrate that the model predicts positive (negative) and it is negative (positive). The unclear observations denote graphs where the pixels displaying closing prices are unavailable. We calculate accuracies by dividing the number of correct (incorrect) observations by the total number of clear observations. The share of unclear observations is estimated as the number of unclear observations to the total number of samples.

	1	2	3	4	5	6	7	8	9	10
Correct	0.58	0.57	0.56	0.56	0.56	0.56	0.56	0.56	0.56	0.57
Incorrect	0.42	0.43	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.43
Unclear	0.04	0.05	0.05	0.06	0.07	0.07	0.07	0.07	0.07	0.29

Table 3: Portfolio Performance

The table compares performance of value-weight (Panel A) and equal-weight (Panel B) quintile portfolios. Animated Stock Market portfolio (ASM) is sorted on out-of-sample predicted pixel change for the next week. Momentum (MOM), short term reversal (ST_REV), and LT_REV portfolios are sorted on prior monthly returns 12-2, 1-0, and 60-13, respectively. Panels report annual return (Ret), annualized standard devation (SD), maximum drawdown (MD), Sharpe ratio (SR), and Calmar Ratio (CR).

	ASM					**MOM**					**ST_REV**					**LT_REV**				
	Ret	SD	MD	SR	CR	Ret	SD	MD	SR	CR	Ret	SD	MD	SR	CR	Ret	SD	MD	SR	CR
PANEL A: Value-Weight																				
Low	0.03	0.17	0.74	0.15	0.04	0.10	0.31	0.79	0.31	0.12	0.07	0.18	0.62	0.40	0.12	0.10	0.22	0.53	0.44	0.19
2	0.11	0.16	0.51	0.68	0.22	0.11	0.19	0.57	0.59	0.20	0.09	0.15	0.47	0.61	0.20	0.10	0.17	0.53	0.62	0.19
3	0.16	0.16	0.39	1.00	0.42	0.11	0.15	0.47	0.74	0.24	0.10	0.15	0.45	0.70	0.23	0.09	0.15	0.48	0.64	0.20
4	0.19	0.17	0.40	1.11	0.47	0.11	0.14	0.41	0.75	0.26	0.12	0.18	0.50	0.66	0.24	0.11	0.14	0.44	0.83	0.26
High	0.31	0.16	0.25	1.87	1.23	0.11	0.17	0.51	0.66	0.22	0.12	0.26	0.75	0.45	0.15	0.11	0.17	0.54	0.63	0.20
H-L	0.28	0.11	0.15	2.47	1.84	0.01	0.27	0.70	0.05	0.02	0.04	0.18	0.49	0.24	0.09	0.01	0.15	0.56	0.05	0.01
PANEL B: Equal-Weight																				
Low	-0.00	0.19	0.80	-0.02	-0.01	0.15	0.30	0.69	0.50	0.22	0.10	0.21	0.62	0.47	0.16	0.17	0.26	0.64	0.66	0.27
2	0.08	0.19	0.63	0.43	0.13	0.15	0.20	0.60	0.72	0.24	0.13	0.18	0.56	0.72	0.23	0.14	0.19	0.57	0.72	0.24
3	0.13	0.19	0.60	0.66	0.21	0.14	0.17	0.52	0.83	0.28	0.14	0.18	0.57	0.75	0.24	0.14	0.17	0.51	0.80	0.27
4	0.17	0.19	0.50	0.87	0.34	0.15	0.16	0.47	0.90	0.31	0.15	0.20	0.63	0.71	0.23	0.13	0.17	0.54	0.76	0.24
High	0.30	0.19	0.40	1.59	0.75	0.15	0.19	0.56	0.79	0.27	0.17	0.28	0.63	0.61	0.27	0.13	0.20	0.57	0.65	0.23
H-L	0.30	0.10	0.13	2.94	2.31	0.00	0.23	0.71	0.01	0.00	0.08	0.15	0.20	0.51	0.38	-0.04	0.14	0.73	-0.30	-0.06

Table 4: VRNN Portfolios vs. Transactions Costs

The table presents impact of transactions costs on the strategy performance and the average monthly turnover. Panel A shows the performance of value-weight portfolios and Panel B of the equal-weight. Panels report annual return (Ret), Sharpe ratio (SR), Calmar Ratio (CR), and average monthly turnover (Turnover).

	PANEL A: Value-Weight										
	0.00%	0.01%	0.02%	0.03%	0.04%	0.05%	0.06%	0.07%	0.08%	0.09%	0.10%
Ret	0.28	0.27	0.25	0.24	0.23	0.21	0.20	0.18	0.17	0.16	0.14
SR	2.49	2.36	2.24	2.12	1.99	1.87	1.75	1.63	1.50	1.38	1.26
CR	1.85	1.74	1.64	1.53	1.43	1.33	1.23	1.14	1.04	0.95	0.86
Turnover	1156%										
	PANEL B: Equal-Weight										
Ret	0.30	0.29	0.27	0.26	0.24	0.23	0.22	0.20	0.19	0.18	0.16
SR	2.95	2.82	2.68	2.55	2.42	2.28	2.15	2.01	1.88	1.75	1.61
CR	2.32	2.19	2.06	1.94	1.82	1.70	1.58	1.47	1.35	1.25	1.14
Turnover	1121%										

Table 5: Alphas from the VRNN based strategy

This table presents alpha from regressing the equal-weithed and value-weighted returns of the long-short strategies sorted by the out-of-sample predicted pixel change for the next week. The independent variables are the factors from CAPM, FF3, FF5, FF6, DHS, and all combined. ***, **, and * present one, five and ten percent significance, respectively.

	Equal-Weight		**Value-Weight**	
Model	Coefficient	t-stat	Coefficient	t-stat
CAPM	0.0058***	13.28	0.0054***	11.75
FF3	0.0058***	13.26	0.0058***	13.26
FF5	0.0058***	13.41	0.0056***	12.04
FF6	0.0058***	13.45	0.0056***	12.10
Q4	0.0059***	13.35	0.0055***	11.87
DHS	0.0059***	13.58	0.0056***	12.20
All+STR +LTR	0.0059***	13.49	0.0053***	11.03

Table 6: VRNN on stock characteristics

This table present the Fama-MacBeth cross-sectional regression. The dependent variable is predicted pixels change in one week horizon based on our VRNN model. The independent variables are lagged and collected from Gu, Kelly, and Xiu, 2020. Mov_avg_Ndays is moving average of daily returns over N days. Mov_avg_Ndays are lagged by one week, the monthly variables such as momentum (mom) variables are lagged by one month and others variables are lagged annually consistent with the original paper. The standard errors are Newey-West adjusted and reported in the parentheses. ***, **, and * present one, five and ten percent significance, respectively. Errors reported in parentheses

	(1)	**(2)**	**(3)**
Dep. Variable	Predicted Pixels	Predicted Pixels	Predicted Pixels
Intercept	13.804***	10.665***	10.682***
	(0.150)	(0.640)	(0.609)
mov_avg_20days	0.871***		0.886***
	(0.064)		(0.079)
mov_avg_5days	-0.921***		-0.886***
	(0.067)		(0.080)
mov_avg_0days	0.070***		0.017
	(0.017)		(0.014)
BETA		-0.305**	-0.245*
		(0.134)	(0.127)
baspread		99.181***	97.664***
		(7.955)	(7.747)
dolvol		0.024	0.034
		(0.070)	(0.067)
mom12m		0.109	0.082
		(0.206)	(0.186)
mom1m		-2.337***	-2.167***
		(0.487)	(0.459)
mom36m		-0.320***	-0.314***
		(0.082)	(0.077)
mom6m		0.497**	0.479**
		(0.246)	(0.233)
mve		0.101	0.084
		(0.072)	(0.068)
pricedelay		0.509***	0.446***
		(0.155)	(0.148)
retvol		-44.393***	-42.648***
		(6.137)	(5.937)
zerotrade		1.757e+07***	1.779e+07***
		(5.437e+06)	(5.269e+06)
No. Observations	396,544	396,544	
R-squared	-0.1077	-2.343e+07	-2.402e+07

Table 7: Future returns on VRNN and stock characteristics

This table present the Fama-MacBeth cross-sectional regression. The dependent variable is weekly return. The independent variables are predicted pixels based on our VRNN model, Mov_avg_Ndays and lagged characteristics collected from Gu, Kelly, and Xiu, 2020. Mov_avg_Ndays is moving average of daily returns over N days. The Mov_avg_Ndays are lagged by one week, monthly variables such as momentum (mom) variables are lagged by one month and others variables are lagged annually consistent with the original paper. The standard errors are Newey-West adjusted and reported in the parentheses. ***, **, and * present one, five and ten percent significance, respectively.

	(1)	**(2)**	**(3)**	**(4)**
Dep. Variable	WeekRet	WeekRet	WeekRet	WeekRet
Intercept	-0.1309	0.2687***	0.7701***	0.4744*
	(0.0903)	(0.0810)	(0.2327)	(0.2467)
pred_week_lag	0.0274***			0.0272***
	(0.0022)			(0.0021)
mov_av_est_20		-0.0180	0.0043	-0.0184
		(0.0205)	(0.0128)	(0.0128)
mov_av_est_5		-0.0033	-0.0143	0.0082
		(0.0135)	(0.0092)	(0.0090)
mov_av_est_60		0.0189*	0.0102*	0.0099*
		(0.0098)	(0.0060)	(0.0057)
BETA			-0.0463	-0.0393
			(0.0594)	(0.0591)
baspread			0.0322	-2.4084
			(2.9797)	(2.9694)
dolvol			0.0176	0.0129
			(0.0235)	(0.0230)
mom12m			0.0434	0.0291
			(0.0702)	(0.0755)
mom1m			-0.2518	-0.2014
			(0.1726)	(0.1764)
mom36m			0.0503	0.0566*
			(0.0323)	(0.0337)
mom6m			-0.0627	-0.0840
			(0.0885)	(0.0955)
mve			-0.0556**	-0.0537**
			(0.0271)	(0.0272)
pricedelay			-0.0832	-0.0900
			(0.0541)	(0.0555)
retvol			-0.1970	1.0002
			(2.2137)	(2.2077)
zerotrade			1.242e+06	2.793e+05
			(2.067e+06)	(2.094e+06)
No. Observations	396544	396544	396544	396544
R-squared	0.0037	-0.0022	-3.218e+05	-1.628e+04