Trinigence

Next-Gen AI Trading

Table of Contents

Trinigence

Trinigence

Introduction

We introduce Artificial Trading Intelligence (ATI), a fully autonomous trading agent designed to adapt and respond to any financial market.

It is based on an ensemble of AI models continuously monitored and self-calibrated in real time with the aim to provide an edge in financial trading.

Using Large Language Model technology gives users the ability to communicate about results, insights and trading strategies directly.

This whitepaper presents an in-depth analysis of the mechanisms and advantages of the ATI. The document is structured as follows: the next section reviews related literature, highlighting existing solutions and their limitations. We then detail our methodology, explaining the architecture of the system, processes, AI techniques and models used. Following this, we present our results, demonstrating the system's performance across different market conditions. Finally, we discuss the implications of our findings and conclude with potential future directions for research and development.

Overview

Quantitative algorithmic trading has shown incredible results in the past. A natural evolution of algorithms comes in the form of AI. The problem with AI algorithms is that they introduce a new layer of complexity as well as being very prone to overfit past data due to the extremely noisy nature of financial price data.

We present a solution based on deep neural networks and ensemble modeling. By using many models instead of one we lower the variance of predictions made by the models, and make a trading decision based on a weighted majority vote.

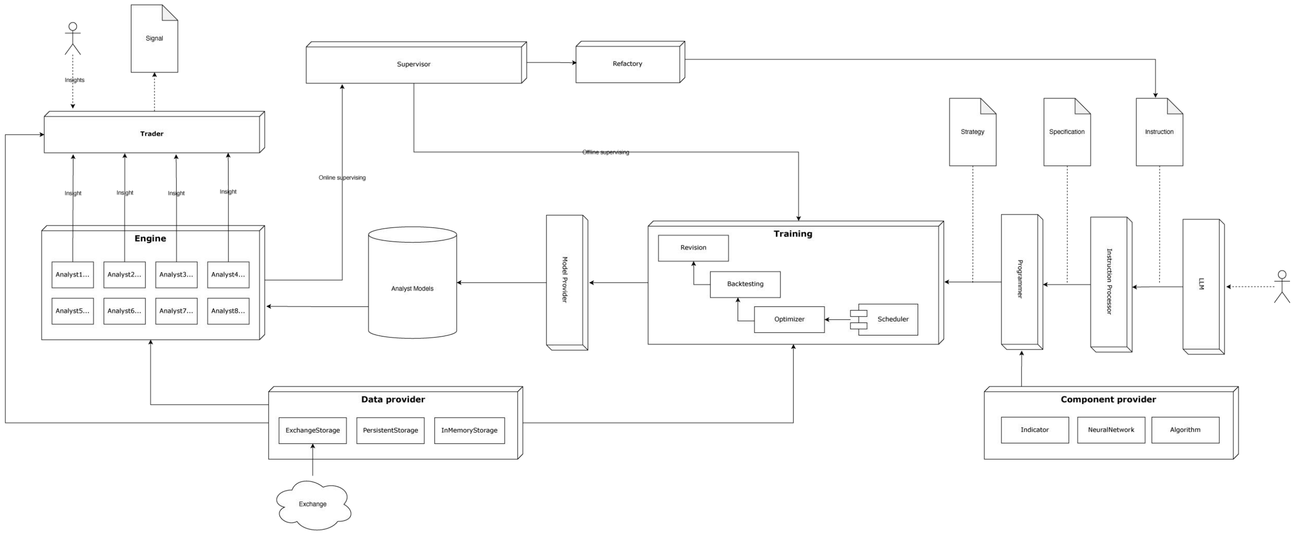
We go a step further by automating the complete process of building, training, deploying and monitoring the trading models, and creating a closed loop where bad performing models will be automatically refactored and retrained. With this we provide a constantly evolving system that quickly adapts to market changes, which also allows for efficient scaling on multiple asset classes as well as different trading strategies.

Our long term goals are to integrate the LLM technology to provide direct and simple communication with the system, allowing the democratization of financial investing and trading for a large number of users. Our system would be able to verify any investment hypothesis, as well as suggest effective investment insights to the user.

Trinigence

Architecture

The architecture below offers a complete view of the automated trading system. We present the structure, as well as data flow through the system, and all the actors in decision-making and development.



Within the system we recognize 3 functional sections that are connected:

- **Execution environment** - Ensures automated trading
  Consists of Analyst Models Store, Data Provider, Analyst Engine, Trader and Supervisor
- **Development environment** - Provides a space for development of both analyst and trader models.
  Consists of Data Provider, Training, Re-factory and Model Provider
- **Statement environment** - Models based on human language statements, serving as a simple interface to communicate directly with the complete system.
  Consists of LLM, Instruction Processor, Programmer and Component Provider

Execution Environment

Trader
The main component responsible for making trading decisions. AI model receives insights from Analysts, acts upon these signals, making trading decisions based on the insights provided.

Engine
Main analysis engine. Contains multiple Analyst models.
Analysts are AI models that process market data and generate insight.

Supervisor
Monitors the online operation of the system. Supervises both trader and analysts in real-time based on defined performance metrics.

Model Repository
A repository where the models used by the analysts are stored. These models are provided to the Engine where Analyst is created for generating insights.

Model Provider
Supplies models to the Analyst Models repository. Ensures the models are up-to-date and effective.

Data Provider
Supplies data to the system. Provides an entry point for all external data from exchanges, trading platforms, officials, news, internal data (trading history).

Development Environment

Training
Automated training of models for best performance including validation.
Backtesting: Tests models against historical data to validate their effectiveness
Optimizer: Refines models for better performance

Re-factory
Works with the Supervisor to make adjustments and improvements on models based on trading history, current market conditions and other inputs.

Component Provider
Supplies various components like indicators, neural networks, and algorithms. All components are described and categorized by different trading methodologies and targets.

Statement Environment

Instruction Processor
Converts instructions into actionable and defined specifications for the system.

Programmer
Develops and implements strategies based on specifications for the system using various development patterns (composite)

LLM (Large Language Model)
Provides natural language processing capabilities to interpret and generate instructions.

Workflow

Data Flow
The Data Provider collects data from the Exchange and stores it in persistent and in-memory storage. This data is then fed to any layer of the system, usually the Engine or Training layers.

Trading Flow
The process begins with the Trader receiving insights from Analysts. Each analyst generates insights based on the data. The analysts could be running different strategies or models to ensure diverse insights. These insights represent various market conditions, indicators, news, or other relevant information that might influence trading decisions.

The Trader uses these insights to make informed trading decisions. Decision is used in a form of Signal to notify interested parties that can further process (for example create order on trading platform)

Supervision and Refinement
Both analyst and trader models provide each own expectation metrics that are used for comparison to current progress. Supervisor is in charge of comparing these metrics and based on market conditions and previous system trading history makes a decision which models should continue progress or stop and mark them for changes.

Each model can operate in two environments: live or staging. Environment depends on the model's performance.

Those stopped models will be processed by Refactory layer for further analyzing, adjustments and schedule them for training.

Instruction Processing
Users define a statement which will be processed by LLM. LLM will recognize and provide instruction with trading details. Instruction is processed by Instruction Processor for further analysis and gives specification as a result. Specification is an extensive description of trading details, strategy details and necessary data providers. Once Specification is ready, The Programmer module will use it as a blueprint to develop a raw model. As a next step, that raw model is delivered to the Development environment.

Trinigence

Model Building Process

In this section we present our approach to building, training and evaluating a new model. The following chart presents an example flow of a creation of a model based on trading type and asset traded.

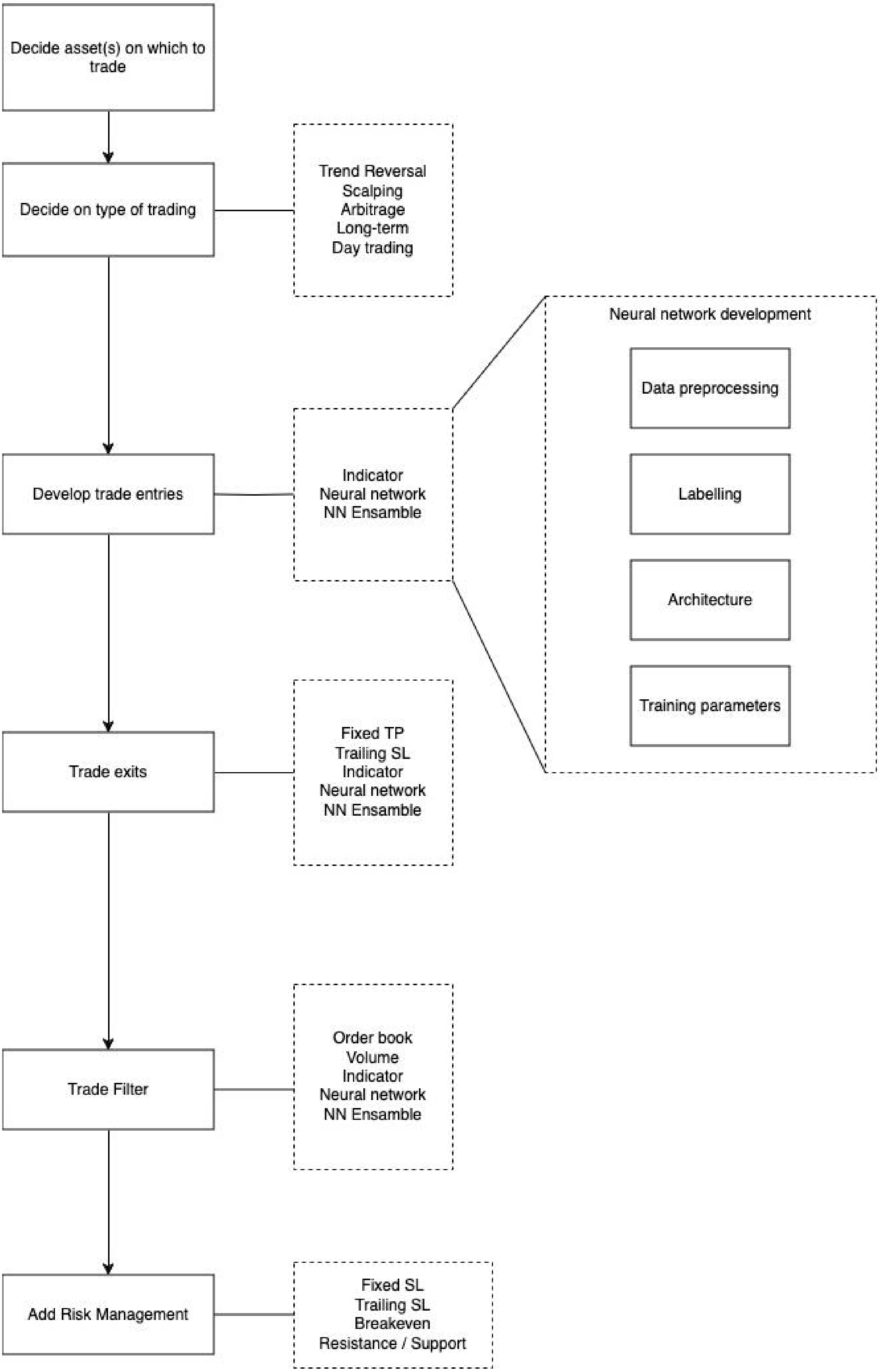
Each trading model has 4 main components:

Trade entry mechanism
The first part of the model scans the market for good opportunities. A good entry can be determined by either a stochastic indicator, a large number of indicators input into a neural network, or a number of indicators input into an ensemble of networks. The choice is made using a hyper-parameter grid search.

Trade exit mechanisms
In a similar fashion, after the model enters a position, the market is monitored for optimal trade exits. The exit might be a fixed profit or loss limit in case of a scalping algorithm, or might scan for changes in trend indicators that signal a trend reversal when using a swing trading algorithm. Similarly to how entry mechanism is determined, a gridsearch is performed over multiple options to determine the optimal method for exits.

Trade filter mechanism
An additional layer is added when deciding trade entries, where some trades will be skipped based on market volatility, volume, anomalous movement or sudden changes in trend. This mechanism ensures as much risk-free entries as possible. Again, this is constructed with indicators, networks or ensemble chosen by gridsearch.

Risk management mechanisms
The risk management layer of a trading model acts as an emergency switch when in a position, if sudden market movements occur or if any manipulation is detected in order to protect the assets that are traded.


Decide asset(s) on which to trade
Decide on type of trading
Trend Reversal
Scalping
Arbitrage
Long-term
Day trading
Neural network development
Data preprocessing
Labelling
Architecture
Training parameters
Develop trade entries
Indicator
Neural network
NN Ensamble
Trade exits
Fixed TP
Trailing SL
Indicator
Neural network
NN Ensamble
Trade Filter
Order book
Volume
Indicator
Neural network
NN Ensamble
Add Risk Management
Fixed SL
Trailing SL
Breakeven
Resistance / Support

Data Collection and Preprocessing

To create an effective trading model, we start by selecting the assets we want to trade based on their volume and volatility. We are focusing on assets with sufficient liquidity and price movement to provide trading opportunities. Our data collection process includes following steps:

Asset Historical and Current Prices
ATI gathers extensive historical price data and continuously updates it with current prices to capture the asset's market behavior over time.

Correlated Asset Prices
Prices of assets that have a historical correlation with the chosen asset are also collected to identify potential market influences and interdependencies.

Best Exchange Information
Information on the best exchanges for trading the asset based on volume.

Financial Reports and News Feeds
Relevant financial reports and real-time news feeds related to the asset are integrated into our data pipeline to capture market sentiment and fundamental factors.

Preprocessing and Normalization
Price data is preprocessed and normalized in order that the model be price-agnostic and focus on changes in price rather than price itself.

Feature engineering
Additional information from price is extracted using technical indicators and stochastic analysis on the price data.

Separation of Concerns

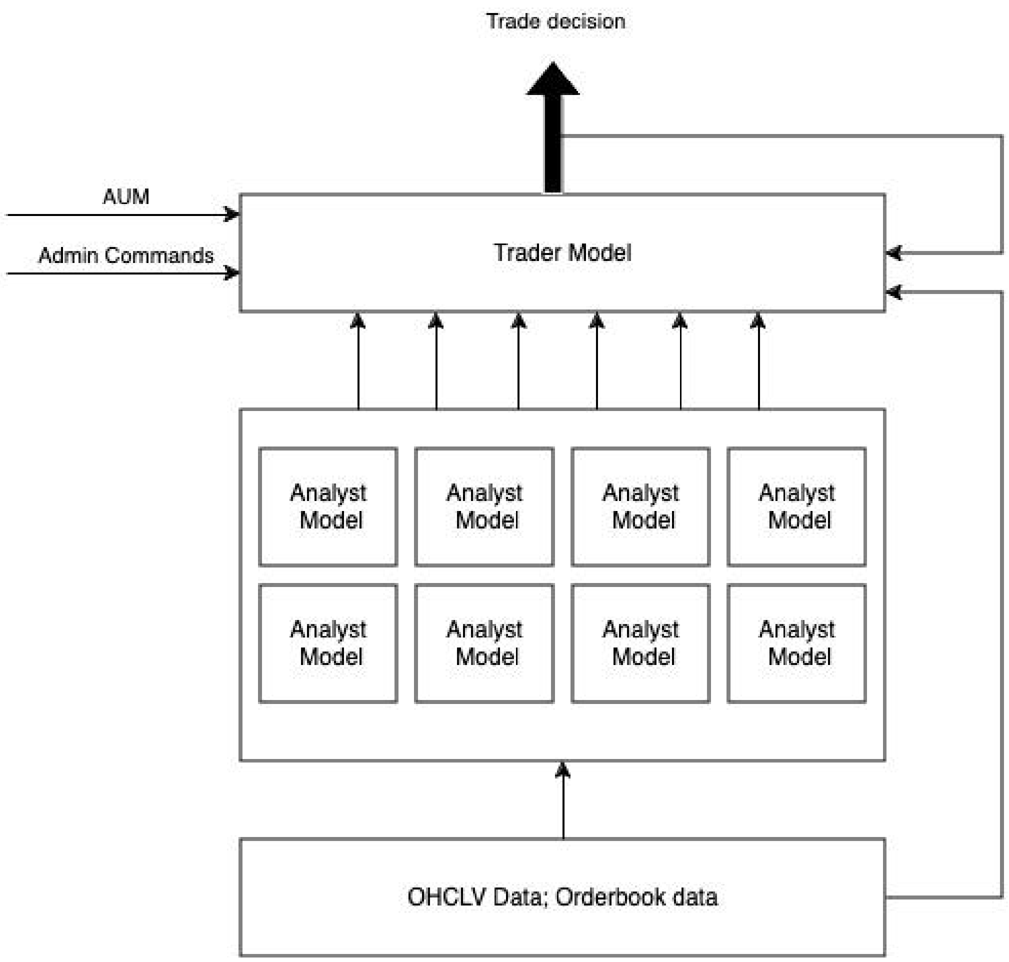
Each model built follows the flow described so far. However, we add another layer of separation and introduce two types of models to further reduce a risk of overfitting, as well as to make behavior of ATI more transparent.

So, two types of models are introduced:

Analyst Models
These models focus only on the market and price, and suggest trades based on a number of methods like trend analysis, anomaly detection, crash/rally detection etc.

Trader Models
The outputs from the Analyst models are then processed by a decision model, known as the Trader. The Trader aggregates all the signals from the Analysts, and includes additional information of the ATI, such as the amount of Assets under management, current market volume, past performance and admin commands , and makes the final trading decisions.

Trader has a feedback loop that updates it's parameters continuously based on each trade it's made.



Evaluation Metrics

For performance metrics we rely on time-tested performance metrics used in trading. No single metric is all encompassing, so we use a number of them combined.

Main Investment Performance Metrics

Calmar Ratio
The Calmar Ratio measures the annual return of an investment compared to its maximum drawdown (the largest peak-to-trough decline). It is calculated as:

Calmar Ratio = Annual Return / Maximum Drawdown

A higher Calmar Ratio indicates a better risk-adjusted return, making it a valuable metric for evaluating the stability and performance of a trading strategy.

K-Ratio
The K-Ratio assesses the consistency of returns over time by analyzing the slope of the cumulative returns line. It is calculated using the following formula:

KRatio = Slope of the Cumulative Returns over Time / Standard Error of the Slope

A higher K-Ratio indicates more consistent returns, making it a key metric for evaluating the reliability of a trading strategy.

Expectancy
Expectancy measures the average amount a trader can expect to win or lose per trade. It is calculated as:

$E = p(w) * avg(w) - p(l) * avg(l)$

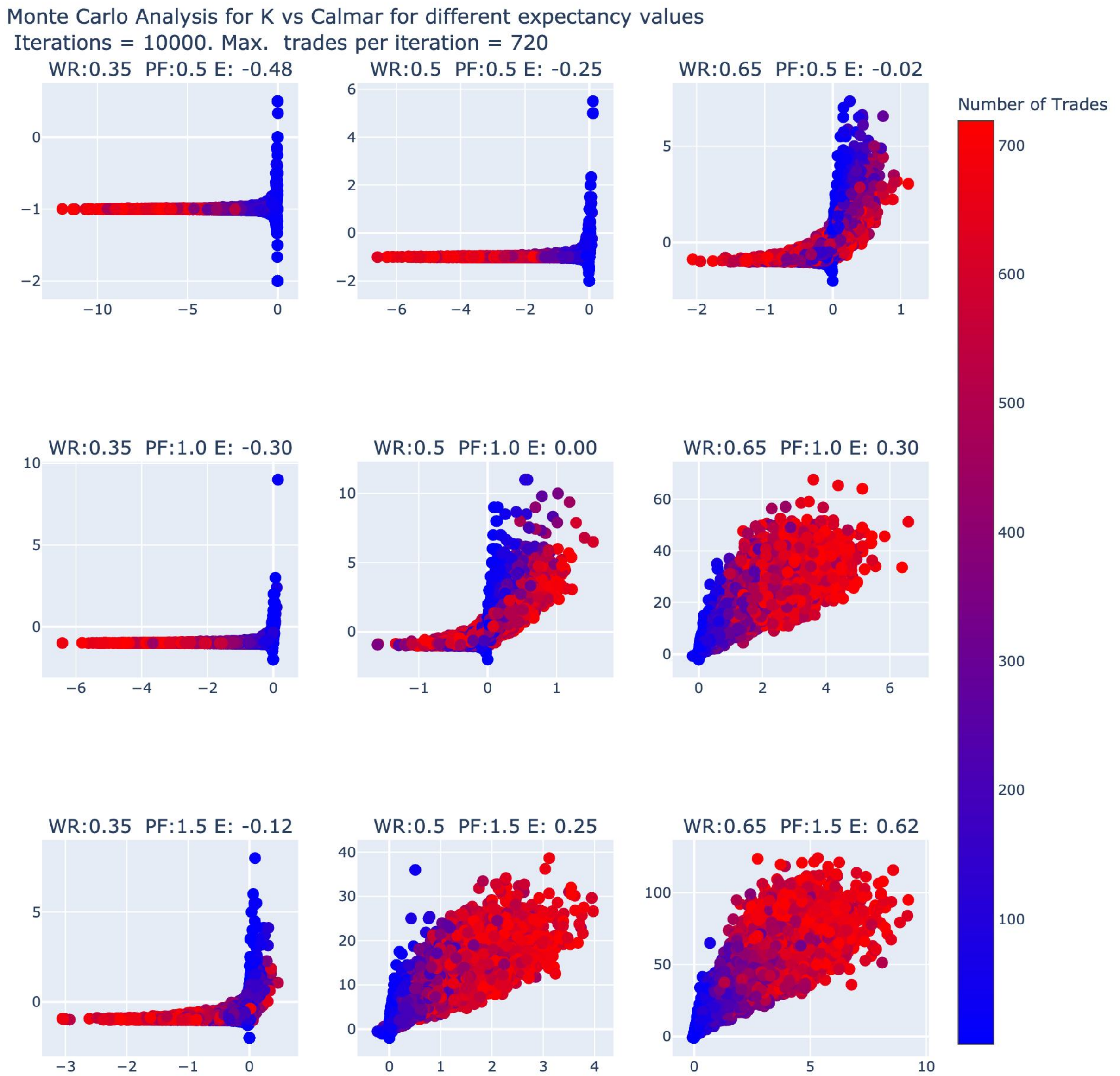
Where p(w) and p(l) represent the probability of a winning and a losing trade respectively, and avg(w) and avg(l) the average value of a winning and the average value of a losing trade.
A positive expectancy indicates a profitable strategy over the long term.

Relation between primary metrics

To understand how the primary metrics correlate to each other, we have performed a Monte-Carlo analysis of various scenarios of a trading model.
Notice that when the expectancy of trades increases, other metrics will follow and there comes a tradeoff in whether the model should provide higher profits (illustrated by a high Calmar ratio) or more consistent and stable profits (indicated by a high K-Ratio).

Our primary goal then shows itself in increasing the Expectancy of our model first, which intuitively means finding better Trade Entry opportunities, and optimal Trade Exit opportunities for each trade.

Note also that even when the expectancy is low (around 0), good profits can still be achieved by lowering the amount of trades with smart filtering techniques.



Secondary Metrics

Sharpe Ratio
The Sharpe Ratio measures the risk-adjusted return of an investment by comparing the excess return to its standard deviation. It helps in assessing the reward per unit of risk.

Sortino Ratio
Similar to the Sharpe Ratio, the Sortino Ratio focuses on downside risk by using the standard deviation of negative returns. It provides a clearer picture of the risk-adjusted return by penalizing only downside volatility.

Risk/Reward Factor
This ratio compares the potential profit of a trade to the potential loss, helping in assessing the attractiveness of trade opportunities.

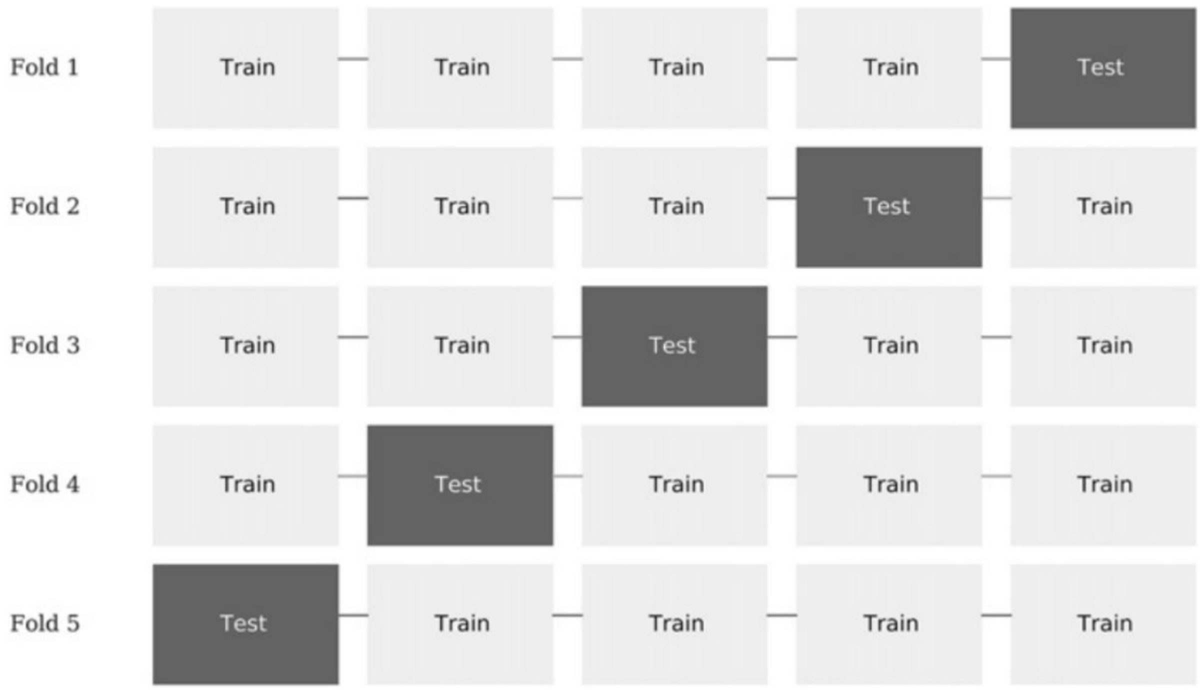
Win Rate
Win Rate represents the percentage of winning trades out of the total trades executed.

Maximum Drawdown
Maximum Drawdown measures the largest decline from a peak to a trough in the investment value, and it gives a simple measure of risk.

Average Monthly Profit
This metric tracks the average profit generated per month, helping in assessing the consistency and profitability of the strategy.

Training

After construction each model is trained based on past data to maximize the evaluation metrics described above. Due to models being prone to overfit past data, we use a KFold approach to training the models on past data. This means dat when trained iteratively, each time the training will take a different part of the dataset as training and testing.



Supervision

Finally, ATI integrates a separate module called the Supervisor, which is responsible for monitoring the performance of each model based on the aforementioned metrics. The Supervisor evaluates models and makes decisions regarding their status using predefined thresholds. The possible statuses include:

Staging
Models in this phase are being tested and validated before full deployment.

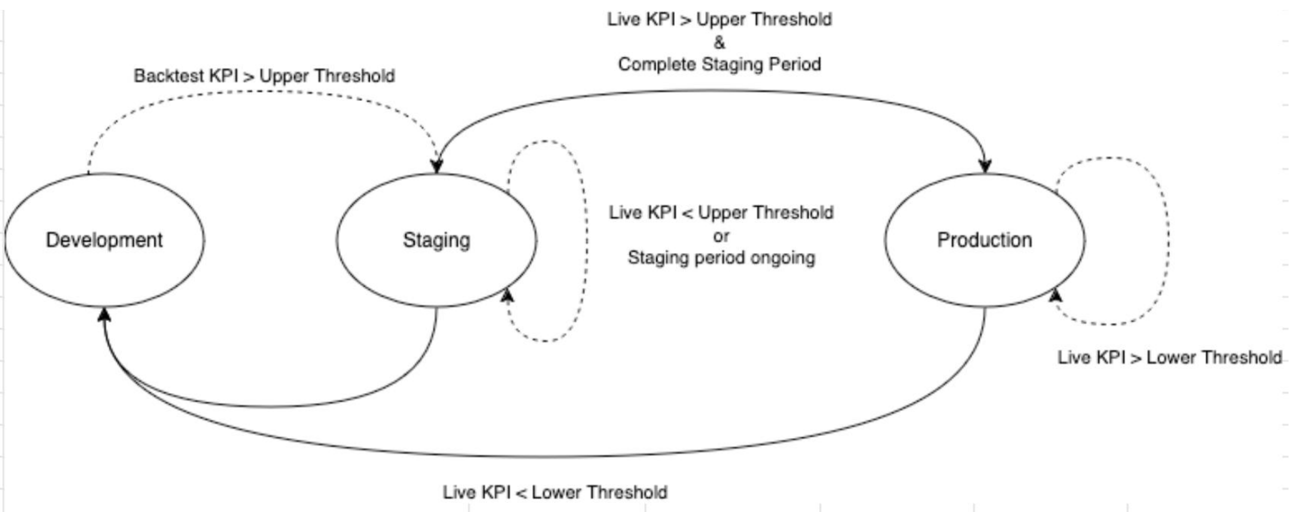
Production
Models that meet performance criteria and demonstrate consistent success are moved to this phase, where they actively participate in trading.

Suspension
Models that underperform or exhibit inconsistent results are temporarily removed from trading activities for further evaluation and potential adjustments.

Removal
Models that fail to meet performance standards over an extended period are permanently removed from the system.

Thresholds for each level (Staging, Production, Suspension, Removal) are determined based on historical performance, industry standards, and specific objectives of the trading system.

Production: *Calmar > 5; KRatio > 1; E > 0.5*
Staging: *3 < Calmar < 5; 0.5 < KRatio < 1; 0.1 < E < 0.5*
Suspension: *1 < Calmar < 3 < 5; 0.1 < KRatio < 0.5; 0 < E < 0.1*
Removal: *Calmar < 1 < 5; KRatio < 0.1; E < 0*



Trinigence

AI trading Proof-of-Concept

In order to show the proof of concept, we conducted an experiment comparing four models over a one-year period, trading Bitcoin (BTC) from June 2023 to June 2024. Each model decides on position entries, which can be both long and short, and exits a position when it reaches either 1% profit or 1% loss. The aim of this experiment is to show the benefits of AI methodologies that are employed in the ATI - neural networks, ensemble methods and continuous training

Model Descriptions

Simple Algorithm (Default MACD+Supertrend)

- **Strategy:** Enters a position when both MACD and Supertrend indicators signify a change of trend.
- **Parameters:** MACD and Supertrend indicators are set to their default parameters.

Trained Algorithm (Optimized MACD+Supertrend)

- **Strategy:** Same as the Simple Algorithm.
- **Parameters:** Optimized using machine learning based on historical data from June 2020 to June 2023. The optimal parameters aim to achieve the best possible KPIs.
- **Testing Period:** Results are tested on data from June 2023 to June 2024.

MLP Neural Network

- **Architecture:** Multi-Layer Perceptron (MLP) that observes 30 technical indicators.
- **Training Period:** Trained on historical data from June 2020 to June 2023.
- **Strategy:** Predicts market movements based on output of neural network.

Ensemble of MLP Neural Networks

- **Architecture:** 100 MLP estimators that observe 30 technical indicators.
- **Training Period:** Trained on historical data from June 2020 to June 2023.
- **Strategy:** Predicts market movements based on output of ensemble.

Continuously Trained MLP Neural Network

- **Architecture:** Same as the MLP Neural Network.
- **Training Period:** Continuously retrained every three months with new data.
- **Strategy:** Predicts market movements based on output of ensemble.

Experiment Results

Simple Algorithm (Default MACD+Supertrend)

The strategy resulted in profits around zero, indicating that the default parameters were not effective in capturing profitable trends in the market.

Trained Algorithm (Optimized MACD+Supertrend)

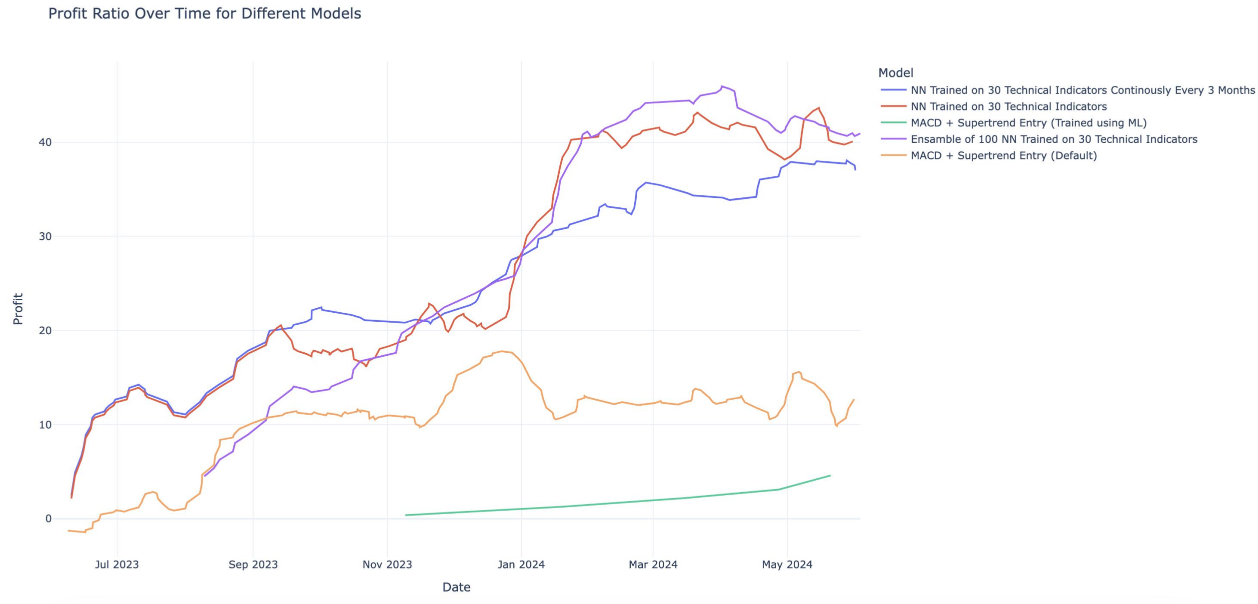
The strategy showed consistently growing profits, but the overall profit was very low, and the number of trades executed was minimal. This suggests that while optimization improved the strategy's effectiveness, it still lacked robustness.

MLP Neural Network

The neural network displayed very good results initially but showed inconsistent profits over time. There were months without any profit, indicating that the model struggled to maintain performance in varying market conditions.

Ensemble of MLP Neural Networks:
Ensemble of networks displayed very good results and generally more consistent. The predictions from the network and trades have less variance.

Continuously Trained MLP Neural Network
This model demonstrated consistently growing profits over the one-year period. The continuous retraining allowed the model to adapt to new market conditions, leading to steady and reliable performance.



Results of experimentations. MACD+Supertrend (Purple) shows lack of robustness and large swings and losses. Trained MACD+Supertrend (Green) shows consistency but low number of trades and Profits. Neural Network based on 30 Technical Indicators (Red) show better but inconsistent profits with months at a time without profit. Same network but retrained (blue) shows consistently rising profits over time.

	Total Trades	Total Profit	Max Drawdown	Calmar Ratio	Expectancy	K-ratio	Average Monthly profit	Sharpe Ratio	Sortino Ratio	Profit Factor
Neural Network 30 Features, Continuously trained	113	38.70 %	5.47 %	**7.05**	0.20	**50.35**	3.17 %	1.22	17.30	1.51
Ensamble of 100 NNs Trained on 30 Technical Indicators	86	**50.27 %**	7.31 %	6.57	0.27	19.69	**3.95 %**	1.29	11.20	1.81
Neural network, 30 Features, Trained Once	162	47.93 %	7.94 %	6.02	0.16	35.09	3.93 %	**1.35**	11.70	1.38
MACD + Supertrend , Trained parameters	9	7.72 %	**1.43 %**	5.36	**0.59**	9.08	0.63 %	0.33	26.87	3.68
MACD + Supertrend, Default Parameters	164	14.83 %	9.45 %	1.56	0.09	12.73	1.22 %	0.73	16.07	1.19

Discussion

Our experiment demonstrates the importance of adaptability and continuous learning in trading strategies. Using default MACD and Supertrend indicators resulted in minimal profits, showing that static, unoptimized strategies are often not effective in changing markets. Training the MACD and Supertrend parameters improved performance, but profits remained low, indicating that optimization is useful but not enough on its own. The MLP Neural Network showed good initial results but became inconsistent over time, suggesting that static models may fail to adapt to new market conditions. Ensemble models amend this slightly by lowering overall variance in prediction. Finally, the continuously trained MLP Neural Network showed consistent profit growth, highlighting that regularly updating the model with new data helps maintain its effectiveness.

Overall, our results highlight the need for trading systems to adapt continuously to new information.

Trinigence

Live production Results

ATI has been trading live, while we are still implementing incremental changes to the system. Here are presented the results of the ATI trading in this period.

CATI v0.1

The initial version of CATI was designed to only enter Long positions. The model was designed to predict a bullish trend before which it would enter a position, and when the market shifted and the bull trend ended the model would exit the position.
Architecture was based on a Neural Network trained on 30 features, over 4 years of historical data. The model was trained to trade both BTC and ETH futures.

CATI v0.2

The first update to CATI was to enable it to enter Short positions, so the model was retrained to predict bearish trends.
The same architecture and training data were used.

CATI v0.3

The third update to CATI was an architecture overhaul that allowed us to train CATI on much more features than before. Also, it could trade on much shorter timeframes which allowed for finding more good trading opportunities.
The model used an advanced MLP architecture, and was trained on 100 features.

CATI v0.4

The fourth update to CATI was adding adaptive filtering mechanisms. Since the last update had allowed us to substantially raise the number of trades we can enter, the next step was to filter as many risky trades as possible, to allow the most consistent performance, while maintaining a profit target.

CATI v0.5

The next update to CATI was adding a separate AI model to determine optimal trade Exit opportunities . With this update we aimed to make CATI as robust as possible, by allowing it to stay longer in “good” positions, that were generating high profits, and exit early on bad positions, or whenever the market seemed to suddenly shift from the expected trend.
The model was based on an XGBoost architecture, trained on 50 features.

	Total Trades	Total Profit	Max Drawdown	Calmar Ratio	Expectancy	K-ratio	Average Monthly profit	Date Start	Date End
CATI v0.1	36	12.94 %	9.88 %	1.47	0.28	14.66	4.03%	2024-01-24	2024-03-17
CATI v0.2	56	**13.60 %**	8.6 %	6.90	0.14	7.38	**4.46 %**	2024-04-01	2024-06-24
CATI v0.3	26	10.32%	**5.78%**	10.71	**0.32**	2.02	**5.2 %**	**2024-05-28**	**2024-07-16**
CATI v0.4	22	9.38%	4.90%	22.9	0.35	11.87	9.38%	2024-06-14	2024-07-17

Table: Results of Live Trading

Trinigence

Regulatory Compliance and Ethical Considerations

An important point we need to reflect upon is maintaining rigorous adherence to regulatory standards and ethical considerations. We hold a deep commitment to uphold the integrity of financial markets and protect the interests of all stakeholders involved.

Regulatory Compliance

Our AI-powered trading platform operates in strict compliance with regulatory frameworks governing financial trading and technology. We are proactive in our approach, continually updating our systems and processes in response to new regulations and industry standards. Key aspects of our compliance strategy include:

Data Protection and Privacy: We adhere to global data protection regulations, including GDPR in Europe and relevant local laws, ensuring the security and confidentiality of user data.
Market Conduct: Our platform is designed to prevent market manipulation and abusive practices, incorporating mechanisms to detect and deter such behavior. We comply with the Market Abuse Regulation (MAR) and other applicable laws to ensure fair trading practices.
Transparency: We provide clear, comprehensive information about how our AI engine works, the data it uses, and the decision-making processes involved. This transparency extends to our pricing, fees, and any risks associated with using our platform.
Licensing and Registration: We have obtained all necessary licenses and registrations required to operate legally in the jurisdictions where our services are offered, working closely with regulatory authorities to ensure full compliance.

Ethical Considerations

Beyond regulatory compliance, we are committed to upholding high ethical standards in the development and deployment of our AI trading technology. Our ethical framework includes:

Fairness and Equality: Ensuring that our platform is accessible and provides equal opportunities for all users, without discrimination.

Transparency and Accountability: While AI decision-making can be complex, we strive for transparency in how our AI operates, making our systems as understandable as possible to users.
Privacy and Data Ethics: We implement robust data governance practices to protect user privacy and ensure that data is used responsibly, ethically, and with consent.
Continuous Monitoring and Improvement: We regularly review and update our AI engine to avoid unintended biases and ensure they make decisions that are ethical and in the best interest of our users.

Ongoing Engagement with Regulatory Bodies

Recognizing the dynamic nature of fintech regulation, we maintain an ongoing dialogue with regulators, industry groups, and other stakeholders. This engagement helps us anticipate and adapt to changes in the regulatory landscape, ensuring that our platform remains compliant and continues to set standards for ethical excellence in trading technology.

Trinigence

Future Work and Roadmap

As the technology of Artificial intelligence has been developed at a mind-boggling pace in recent years, many are trying to leverage these innovations to find success in the rapidly changing global market. We aim to master the technology behind AI and stand at the frontier of this new era, and architect a fully autonomous trading intelligence and deliver to our clients a level of efficiency, accuracy, and performance that was once thought impossible.

Our strategic roadmap for achieving full autonomous trading is structured around a phased approach, with new components to integrate into the overarching artificial intelligence. We intend to build our technology iteratively - through each step we will build confidence and experience, and every technological progress must be based on a stable and tested foundation.

Some of our goals we will list here:

Statement Trading
With the rise of LLM technology, interfaces to machines have never been more accessible. We aim to add a simple and intuitive language model, which will allow our users to communicate with the ATI, ask for trade opportunities, give it suggestions, or simply give it a set of instructions for a trading strategy which the ATI will test, optimize and return suggestions and results.
This approach will allow any user who isn't used to programming or trading tools to take part in the trading and investment landscape.

Expanding to new markets
One of our main goals is to adapt and expand the technology to efficiently invest and diversify it's portfolio over any financial market, be it cryptocurrencies, forex, stocks, commodities, indices etc. This is a continuous task and we aim to add at least one new market every year.

Adapt to other trading methods
In addition to current Swing trading, we aim to expand our technology and tackle the complex environments of High Frequency Trading, with first sights on arbitrage and scalping.

We aim not only to participate in the market, but to innovate and redefine the landscape of financial trading and investments. This strategy ensures that each phase builds upon the last, and in gradual, controlled, and documented steps we push the boundaries of what was thought possible in the field of financial technology and beyond.

Trinigence

Appendix A. Theoretical Background

Technical Analysis

Technical analysis is a method used in financial trading to evaluate and forecast market trends by analyzing historical price data and trading volumes. This approach relies on chart patterns and technical indicators to predict future market movements, assuming that historical performance can provide insights into future behavior.

Neural Networks

Neural networks are a class of artificial intelligence models inspired by the human brain's structure and function. They consist of interconnected layers of nodes (neurons) that process input data to recognize patterns and make decisions. Neural networks excel in handling complex, high-dimensional data, making them suitable for financial trading. In trading, neural networks can identify intricate market patterns and correlations that are difficult for traditional models to capture.

Deep Learning

Deep learning is a subset of machine learning that involves training neural networks with many layers (deep neural networks) to perform sophisticated tasks. By leveraging large datasets and high computational power, deep learning models can learn hierarchical representations of data, enabling them to make accurate predictions and adapt to new information over time. Deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have proven effective in processing time-series data and predicting market trends.

Ensemble Network Systems

Ensemble network systems combine multiple neural network models to improve prediction accuracy and robustness. By aggregating the outputs of different models, ensemble methods reduce the risk of overfitting and enhance the generalization capability of the system. This approach is particularly effective in financial trading, where diverse market conditions and data variability pose significant challenges.

Bagging (Bootstrap Aggregating) is an ensemble technique that improves model accuracy by training multiple versions of a model on different subsets of the data and then averaging their predictions. This method reduces variance and helps prevent overfitting. An example of a bagging model is the Random Forest, which constructs multiple decision trees and merges their outputs to make a final prediction.

Boosting is another ensemble technique that converts weak models into strong ones by focusing on the errors of previous models. It sequentially trains models, each one correcting the errors of its predecessor. XGBoost is a popular boosting algorithm that enhances prediction accuracy by emphasizing difficult-to-predict data points. In trading, these ensemble methods can enhance the model's ability to adapt to complex market patterns and improve overall performance by creating robust trading signals through combined outputs.

Bagging vs Boosting Ensemble Methods In Financial Applications
Boosting's main advantage is that it reduces both variance and bias in forecasts. However, correcting bias comes at the cost of greater risk of overfitting. It could be argued that in financial applications bagging is generally preferable to boosting. Bagging addresses overfitting, while boosting addresses underfitting. Overfitting is often a greater concern than underfitting, as it is not difficult to overfit an ML algorithm to financial data, because of the low signal-to-noise ratio. Further- more, bagging can be parallelized, while generally boosting requires sequential running.

Trinigence

Appendix B. A review of techniques used in financial market forecasting

Asset prices forecasting for the stock market is a very difficult and complicated task [1] since several micro and macroeconomic attributes and characteristics influence the price formation, such as political events, news, company balance sheets, among others [2]. These factors contribute to the nonlinear and non-stationary characteristics presented by the market, favoring the proposed task complexity [3], [4].

Therefore, the studies of these influences are made through market analysis, and their main objective is to predict future directions to assist decision making based on market behav- ior [5]. The literature presents two main approaches: Fundamental Analysis (FA) and Technical Analysis (TA). Both have the same primary objective, and the difference is the information set used for forecasting and decision making. The first focuses on studying company data and seeks to determine whether it has growth potential in the medium to long term [6].

In contrast, TA does not consider the company data, since investors who use this approach believe that information capable of moving the market is absorbed and reflected in the share price [7]. In other words, company balance sheets accounting scandals, financial crises, or any relevant information capable of generating volatility in an asset is reflected
in their price. Therefore, it is possible to avoid the FA data, which are often subjective, to identify patterns present in the asset graph through this strategy type.

Technical analysts make extensive use of Technical Indicators (TI) and candlestick pattern analysis to assist in price movement forecast. Several scientific articles used price information (Open, High, Low, Close prices – OHLC), trading volume, and indicators set as model input based on these techniques. However, when modeling these analyses, two different approaches are used; the works that used TIs generally adopted regression techniques [8]–[10] and those that analyzed candlestick patterns adopted image processing techniques [11]–[13].

A computational intelligence techniques survey for forecasting prices in the stock market was proposed byKumar et al. [14] and the authors identified that TIs play an essential role; however, identifying an adequate TIs set is still an open problem.

Regarding works based on statistical methods, several authors stated that they did not perform efficiently and generated inferior results to models based on artificial intelligence (AI) [15]–[18], as statistical techniques treat financial time series as linear systems. Additionally, the survey of Cavalcante et al. [5] stated that some financial time series characteristics are responsible for the difficult task of forecasting compared to other time series. Thus, traditional statistical methods are not effectively applied to the economic context.

White [19] was the pioneer in implementing an artificial neural network (ANN) for financial market forecasting. The author used the daily prices of IBM company as a database. As it was just an initial study, it did not achieve the expected results. It highlighted the difficulties encountered, such as the overfitting problem and low complexity of the neural network, since only a few entries and one hidden layer were used. It was also mentioned possible future works, such as adding a higher number of features in the ANN, working with different forecasting horizons, and evaluating model profitability.

Besides, Cavalcante et al. [5] selected publications on computational intelligence from 2009 to 2015 and noted that ANNs were widely used and highlighted Deep Learning (DL) as future work. Then, the survey of Kumar et al. [14] presented works that addressed computational intelligence and explored publications from 2016 to 2019, that is, a continuation of the previous work. They highlighted several hybrid implementations and some based on ANN, fuzzy, and DL. Additionally, Gandhmal and Kumar [20] and Nti et al. [21] noted that ANNs were widely used and performed better than fuzzy, support vector machine (SVM) and decision trees since ANNs had more significant potential for generalization. Besides that, Fawaz et al. [22] concluded that DL techniques were able to achieve performance similar to the state-of-the-art for time series classification.

TA is often used for investments with a shorter horizon, trend forecasts, and reversal points identification [5]. Therefore, the timeframe used for model training must be taken into account. The vast majority of previous works used daily candles for a one-day forecast horizon or more. In the review by Nti et al. [21], the 81 publications using TA only 5 worked with intraday candles, showing a differential potential for future works.

The justification for the lack of research that explores smaller timeframes can be either positive (a study yet to be explored) or negative (not showing exciting results). However, it is possible to justify, in principle, the advantage of using a smaller graphic period through the work of Kumar et al. [14], which presented the instances number of each reviewed articles and the one with the highest number was 4818, between the years 1986 and 2005, that is, 267 instances per year on average. As for training, DL models require large data volumes, and this amount of daily candles is relatively small. However, when training with intraday data, the 267 annual instances increase to 28836, considering 9 hours of trading and a 5-minute timeframe. Sezer et al. [23] conducted a DL techniques survey for forecasting financial time series and concluded that recurrent neural networks (RNN) are the most explored by researchers. However, in their review, the authors did not limit the entry attributes set and used FA data, news, price history, market behavior, and TIs. The work focus was to present and analyse the techniques used, including the performance criteria and platforms adopted.

Nowadays, with the development of natural language processing (NLP) and the large volume of news available, sentiment analysis has been applied with relative success in the financial market [24]. Several works use news information together with historical prices for forecasts and have shown results superior to models that use only OHLCV [25]–[27]. Cavalcante et al. [5] identified the works generally did not use trading strategies. Also, they did not evaluate the profitability, reinforcing the conclusion of White [19] and the affirmation of Vanstone and Finnie [6], which say there is much research that does not validate the profitability, resulting in several inconsistent models in the long term. Thus, these issues have generated the greatest contribution of Cavalcante et al. [5] work, which added two final phases for the financial forecasting standard methodology: trading strategy and profitability evaluation.

To reinforce the need for this new methodology it is possible to cite the Nazário et al. [28] work, which analyzed 85 articles and only 31 used some trading strategy. Also, Wang et al. [10] identified that the metrics used for Machine Learning (ML) models have a low correlation with financial metrics, reinforcing the great importance of a completely autonomous system for correct financial validation.

Comparative studies were done by [39], demonstrating that TIs improved the ML model prediction. In the proposed work, several TIs were generated through 1-hour intraday data and a 24-hour forecast window. During the pre-processing step, the data were normalized and an autocorrelation function was used to select only the relevant input data, resulting in 9 TIs. Then, 14 ML models, including CNNs, were implemented and the results obtained by the authors demonstrated that the TIs inclusion increased the next day price forecast accuracy. A different model was proposed by [52], which uses a CNN network with graph theory implemented. Two models were proposed for tests and comparisons, the first based on correlation and the second on causality. Besides, an ML predictor with linear regression and another using ARIMA also served to compare the results of root-mean-square error (RMSE), mean absolute percentage error (MAPE) e mean absolute error (MAE). The results showed that the proposed model presented smaller errors than traditional techniques, but did not perform tests with a simple CNN network or an LSTM network.

On the other hand, Sezer and Ozbayoglu [55] implemented a CNN network and used several models to compare the work developed performance, including an LSTM network. The significant difference lies in creating a 15 × 15 matrix formed by 15 TIs and 15 different periods, resulting in CNN input like an image. Finally, they obtained accuracy, precision, and profitability superior to the comparative models. However, Sim et al. [56] proposed a similar CNN but using the information of 1-minute. In the experimental tests, CNN showed better results than ANN and SVM, but when varying the amount of TIs, there were no improvements in results. Unlike most published works, Wang et al. [10] proposed a new predictor model based on a one-dimensional CNN (1D CNN) capable of extracting data characteristics; that is, it is not necessary to create TIs. Also, to classify the market as upward, downward, or consolidating, they used a function based on closing price and volatility.

Regarding the works that implemented the LSTM technique, which are more than half of the analysed publications, some approached the pre-processing, results comparisons, and accuracy metrics in similar ways. The authors by [40], [43], [44], [47], [53] used asset prices and TIs as network attributes, and the data were normalized to feed the model input based on the LSTM network. Among them, only Qiu et al. [53] proposed a new model: a combination of LSTM and GRU (Gated Recurrent Unit), to explore LSTM ability to process sequential data and the simplicity of GRU, reducing training time and computational cost.

In contrast, the works developed by [38], [41], [54] opted to use standardization in the pre-processing stage. Also, Chen et al. [38] proposed a predictive model based on LSTM with Attention Mechanism (AM) and Market Vector (MV), with the MV being responsible for capturing correlations between assets. The results showed that this predictor obtained the smallest errors, being more effective than the commonly implemented LSTM with TIs. In turn, JuHyok et al. [41] used 16 candlesticks patterns modeled in the TI format to feed an LSTM network, performing better than a CNN.

AutoSLSTM was developed by [35], which had as its first layer an LSTM network with autoencoder followed by two other simple LSTM layers. The autoencoder technique proved to be very useful in reducing input data noise, resulting in minor errors concerning a simple LSTM network and a traditional MLP. During the tests, the forecast horizon varied between 1, 5, and 10 days. There were concluded that the higher the parameter value, the more error is accumulated. Moreover, Labiad et al. [44] also obtained precision values that decreased with the increase of the forecast horizon, 10, 30, and 60-minutes.

A model with two layers of LSTM stacked and generated 400 characteristics based on market information was proposed by [37]. However, for training, only 250 were randomly selected. Thus, a different attribute set was selected for each new training, resulting in different training data. In contrast, Agrawal et al. [34] developed an optimized LSTM, and through the correlation-tensor technique, adaptive TIs were generated, resulting in better accuracy and lower MSE. Reference [45] used OHLCV information to generate 4 TIs and feed the input of an ARIMA model, then the output was used for the model based on LSTM. In contrast, Wen et al. [63] proposed the PCA-LSTM, whose the PCA (Principal Component Analysis) technique was responsible for extracting TI characteristics and reducing dimensionality, resulting in better predictions concerning the compared models.

In turn, Tan et al. [59] reduced dimensionality through an elastic net model and used LSTM as a predictor, but integrated with the Sharpe-Optimized method to achieve a balanced investment strategy with the risk-return. They obtained a financial accumulation of 75% higher than the traditional linear model and performance above the ML models. Despite Nelson et al. [8] used the same predictor of other studies, the authors generated a large number of indicators and normalized them using the log-return transform. The results showed an accuracy slightly above 50%, but it reduced the maximum drawdown in most studied assets. Works that made use of textual data, such as news, also obtained good results as shown in [48], [49], [62], [64], which used predictors based on LSTM, autoencoders, deep belief network (DBN), and AM, respectively. In all models, textual data is pre-processed using sentiment analysis techniques and later concatenated with the price and TI data.

Several authors have developed research using hybrid models for forecasting, and all models had LSTM or RNN layers linked to CNN layers. In the works of [46], [50], [51], [60], [61], the authors used CNN to extract textual data patterns, such as news channels and social networks, thus generating more information than just the asset price. All of them showed better results than a network with only LSTM implemented.

It is worth highlighting the work of Oncharoen and Vateekul [51] since it was proposed to change the loss function by adding Sharpe ratio information to the cross-entropy equation. Thus, the risk-return was calculated and weighted during the training. A metric based on the Sharpe ratio and F1 score, called Sharpe-F1 score, was proposed to select the best models based on risks presented.

Finally, Alonso-Monsalve et al. [36] used CNN layers to extract patterns in an 18 TIs set and OHLCV formed by six cryptocurrency, and an LSTM layer to generate a trend. Kelotra and Pandey [42] proposed an optimization algorithm, Rider-based monarch butterfly optimization, used to train the predictor based on a Convolutional LSTM Network (ConvLSTM), an algorithm used for sequential images. Furthermore, Zhou et al. [65] used a network with LSTM layers linked to CNN layers to market direction forecasting and implemented the GAN (Generative Adversarial Network) technique for the training process.

References

[1] L. A. Teixeira and A. L. I. De Oliveira, “A method for automatic stock trading combining technical analysis and nearest neighbor classification,” Expert Syst. Appl., vol. 37, no. 10, pp. 6885–6890, Oct. 2010.
[2] J. L. Ticknor, “A Bayesian regularized artificial neural network for stock market forecasting,” Expert Syst. Appl., vol. 40, no. 14, pp. 5501–5506, Oct. 2013.
[3] P.-C. Chang, D.-D. Wang, and C.-L. Zhou, “A novel model by evolving partially connected neural network for stock price trend forecasting,” Expert Syst. Appl., vol. 39, no. 1, pp. 611–620, Jan. 2012.
[4] M.-W. Hsu, S. Lessmann, M.-C. Sung, T. Ma, and J. E. V. Johnson, “Bridging the divide in financial market forecasting: Machine learners vs. financial economists,” Expert Syst. Appl., vol. 61, pp. 215–234, Nov. 2016.
[5] R. C. Cavalcante, R. C. Brasileiro, V. L. F. Souza, J. P. Nobrega, and A. L. I. Oliveira, “Computational intelligence and financial markets: A survey and future directions,” Expert Syst. Appl., vol. 55, pp. 194–211, Aug. 2016.
[6] B. Vanstone and G. Finnie, “An empirical methodology for developing stockmarket trading systems using artificial neural networks,” Expert Syst. Appl., vol. 36, no. 3, pp. 6668–6680, Apr. 2009.
[7] J. J. Murphy, Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications. New York, NY, USA: New York Institute of Finance, 1999.
[8] D. M. Q. Nelson, A. C. M. Pereira, and R. A. de Oliveira, “Stock market’s price movement prediction with LSTM neural networks,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Anchorage, AK, USA, May 2017, pp. 1419–1426.
[9] W. Bao, J. Yue, and Y. Rao, “A deep learning framework for financial time series using stacked autoencoders and long-short term memory,” PLoS ONE, vol. 12, no. 7, Jul. 2017, Art. no. e0180944.
[10] J. Wang, T. Sun, B. Liu, Y. Cao, and D. Wang, “Financial markets prediction with deep learning,” in Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA), Dec. 2018, pp. 97–104.
[11] M. Wang and Y. Wang, “Evaluating the effectiveness of candlestick analysis in forecasting U.S. stock market,” in Proc. 3rd Int. Conf. Compute Data Anal., Mar. 2019, pp. 98–101.
[12] S. Rajaa and J. K. Sahoo, “Convolutional feature extraction and neural arithmetic logic units for stock prediction,” in Proc. Int. Conf. Adv. Comput. Data Sci. Singapore: Springer, 2019, pp. 349–359.
[13] H. Qiu and F. Liu, “Candlestick analysis in forecasting U.S. stock market: Are they informative and effective,” in Proc. IEEE 4th Int. Conf. Big Data Anal. (ICBDA), Mar. 2019, pp. 325–328.
[14] G. Kumar, S. Jain, and U. P. Singh, “Stock market forecasting using com- putational intelligence: A survey,” Arch. Comput. Methods Eng., pp. 1–33, Feb. 2020.
[15] G. S. Atsalakis and K. P. Valavanis, “Surveying stock market forecasting techniques—Part II: Soft computing methods,” Expert Syst. Appl., vol. 36, no. 3, pp. 5932–5941, Apr. 2009.
[16] M. A. Boyacioglu and D. Avci, “An adaptive network-based fuzzy inference system (ANFIS) for the prediction of stock market return: The case of the Istanbul stock exchange,” Expert Syst. Appl., vol. 37, no. 12, pp. 7908–7912, Dec. 2010.
[17] D. A. Kumar and S. Murugan, “Performance analysis of indian stock market index using neural network time series model,” in Proc. Int. Conf. Pattern Recognit., Informat. Mobile Eng., Feb. 2013, pp. 72–78.
[18] R. Bisoi and P. K. Dash, “A hybrid evolutionary dynamic neural network for stock market trend analysis and prediction using unscented Kalman filter,” Appl. Soft Comput., vol. 19, pp. 41–56, Jun. 2014.
[19] H. White, “Economic prediction using neural networks: The case of IBM daily stock returns,” in Proc. IEEE Int. Conf. Neural Netw., San Diego, CA, USA, Jul. 1988, pp. 451–458.
[20] D. P. Gandhmal and K. Kumar, “Systematic analysis and review of stock market prediction techniques,” Comput. Sci. Rev., vol. 34, Nov. 2019, Art. no. 100190.

[21] I. K. Nti, A. F. Adekoya, and B. A. Weyori, “A systematic review of fundamental and technical analysis of stock market predictions,” Artif. Intell. Rev., vol. 53, pp. 3007–3057, Aug. 2019.
[22] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, “Deep learning for time series classification: A review,” Data Mining Knowl. Discovery, vol. 33, no. 4, pp. 917–963, Jul. 2019.
[23] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, “Financial time series forecasting with deep learning: A systematic literature review: 2005–2019,” Appl. Soft Comput., vol. 90, May 2020, Art. no. 106181.
[24] E. Cambria and B. White, “Jumping NLP curves: A review of natural language processing research [review article],” IEEE Comput. Intell. Mag., vol. 9, no. 2, pp. 48–57, May 2014.
[25] S. Merello, A. P. Ratto, Y. Ma, O. Luca, and E. Cambria, “Investigating timing and impact of news on the stock market,” in Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW), Nov. 2018, pp. 1348–1354.
[26] L. Malandri, F. Z. Xing, C. Orsenigo, C. Vercellis, and E. Cambria, “Public mood–driven asset allocation: The importance of financial sentiment in portfolio management,” Cognit. Comput., vol. 10, no. 6, pp. 1167–1176, Dec. 2018.
[27] A. Picasso, S. Merello, Y. Ma, L. Oneto, and E. Cambria, “Technical analysis and sentiment embeddings for market trend prediction,” Expert Syst. Appl., vol. 135, pp. 60–70, Nov. 2019.
[28] R. T. F. Nazário, J. L. E. Silva, V. A. Sobreiro, and H. Kimura, “A literature review of technical analysis on stock markets,” Quart. Rev. Econ. Finance, vol. 66, pp. 115–126, Nov. 2017.
[29] B. Kitchenham and S. Charters, “Guidelines for performing systematic literature reviews in software engineering,” Engineering, vol. 45, no. 4ve, p. 1051, 2007.
[30] X. Wang, Z. Fang, and X. Sun, “Usage patterns of scholarly articles on Web of science: A study on Web of science usage count,” Scientometrics, vol. 109, no. 2, pp. 917–926, Nov. 2016.
[31] M. K. Paras, R. Pal, and D. Ekwall, “Systematic literature review to develop a conceptual framework for a reuse-based clothing value chain,” Int. Rev. Retail, Distrib. Consum. Res., vol. 28, no. 3, pp. 231–258, May 2018.
[32] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering— A systematic literature review,” Inf. Softw. Technol., vol. 51, no. 1, pp. 7–15, Jan. 2009.
[33] F. J. Alves, E. A. De Carvalho, J. Aguilar, L. L. De Brito, and G. S. Bastos, “Applied behavior analysis for the treatment of autism: A systematic review of assistive technologies,” IEEE Access, vol. 8, pp. 118664–118672, 2020.
[34] M. Agrawal, A. U. Khan, and P. K. Shukla, “Stock price prediction using technical indicators: A predictive model using optimal deep learning,” Learning, vol. 6, no. 2, p. 7, 2019.
[35] K. A. Al-Thelaya, E.-S.-M. El-Alfy, and S. Mohammed, “Forecasting of bahrain stock market with deep learning: Methodology and case study,” in Proc. 8th Int. Conf. Modeling Simulation Appl. Optim. (ICMSAO), Apr. 2019, pp. 1–5.
[36] S. Alonso-Monsalve, A. L. Suárez-Cetrulo, A. Cervantes, and D. Quintana, “Convolution on neural networks for high-frequency trend prediction of cryptocurrency exchange rates using technical indicators,” Expert Syst. Appl., vol. 149, Jul. 2020, Art. no. 113250.
[37] S. Borovkova and I. Tsiamas, “An ensemble of LSTM neural networks for high-frequency stock market classification,” J. Forecasting, vol. 38, no. 6, pp. 600–619, 2019.
[38] Y. Chen, J. Wu, and H. Bu, “Stock market embedding and prediction: A deep learning method,” in Proc. 15th Int. Conf. Service Syst. Service Manage. (ICSSSM), Jul. 2018, pp. 1–6.
VOLUME 8, 2020 185241 A. W. Li, G. S. Bastos: Stock Market Forecasting Using Deep Learning and Technical Analysis

[39] S. Demir, K. Mincev, K. Kok, and N. G. Paterakis, “Introducing technical indicators to electricity price forecasting: A feature engineering study for linear, ensemble, and deep machine learning models,” Appl. Sci., vol. 10, no. 1, p. 255, Dec. 2019.
[40] A. Fazeli and S. Houghten, “Deep learning for the prediction of stock market trends,” in Proc. IEEE Int. Conf. Big Data (Big Data), Dec. 2019, pp. 5513–5521.
[41] U. JuHyok, P. Lu, C. Kim, U. Ryu, and K. Pak, “A new LSTM based reversal point prediction method using upward/downward reversal point feature sets,” Chaos, Solitons Fractals, vol. 132, Mar. 2020, Art. no. 109559.
[42] A. Kelotra and P. Pandey, “Stock market prediction using optimized deep ConvLSTM model,” Big Data, vol. 8, no. 1, pp. 5–24, Feb. 2020.
[43] K. Khare, O. Darekar, P. Gupta, and V. Z. Attar, “Short term stock price prediction using deep learning,” in Proc. 2nd IEEE Int. Conf. Recent Trends Electron., Inf. Commun. Technol. (RTEICT), May 2017, pp. 482–486.
[44] B. Labiad, A. Berrado, and L. Benabbou, “Short term prediction frame- work for moroccan stock market using artificial neural networks,” in Proc. 12th Int. Conf. Intell. Syst., Theories Appl. (SITA), 2018, pp. 1–6.
[45] C. Y. Lai, R.-C. Chen, and R. E. Caraka, “Prediction stock price based on different index factors using LSTM,” in Proc. Int. Conf. Mach. Learn. Cybern. (ICMLC), Jul. 2019, pp. 1–6.
[46] C.-Y. Lee and V.-W. Soo, “Predict stock price with financial news based on recurrent convolutional neural networks,” in Proc. Conf. Technol. Appl. Artif. Intell. (TAAI), Dec. 2017, pp. 160–165.
[47] W. Li and J. Liao, “A comparative study on trend forecasting approach for stock price time series,” in Proc. 11th IEEE Int. Conf. Anti-Counterfeiting,Secur., Identificat. (ASID), Oct. 2017, pp. 74–78.
[48] X. Li, P. Wu, and W. Wang, “Incorporating stock prices and news sentiments for stock market prediction: A case of Hong Kong,” Inf. Process. Manage., vol. 57, no. 5, Sep. 2020, Art. no. 102212.
[49] T. Matsubara, R. Akita, and K. Uehara, “Stock price prediction by deep neural generative model of news articles,” IEICE Trans. Inf. Syst., vol. 101, no. 4, pp. 901–908, 2018.
[50] P. Oncharoen and P. Vateekul, “Deep learning for stock market prediction using event embedding and technical indicators,” in Proc. 5th Int. Conf. Adv. Inform., Concept Theory Appl. (ICAICTA), Aug. 2018, pp. 19–24.
[51] P. Oncharoen and P. Vateekul, “Deep learning using risk-reward function for stock market prediction,” in Proc. 2nd Int. Conf. Comput. Sci. Artif.Intell. (CSAI), 2018, pp. 556–561.
[52] P. Patil, C.-S.-M. Wu, K. Potika, and M. Orang, “Stock market prediction using ensemble of graph theory, machine learning and deep learning models,” in Proc. 3rd Int. Conf. Softw. Eng. Inf. Manage., Jan. 2020, pp. 85–92.
[53] Y. Qiu, H.-Y. Yang, S. Lu, and W. Chen, “A novel hybrid model based on recurrent neural networks for stock market timing,” Soft Comput., vol. 24, pp. 15273–15290, 2020.
[54] Y. Qu and X. Zhao, “Application of LSTM neural network in forecasting foreign exchange price,” J. Phys., Conf. Ser., vol. 1237, Jun. 2019,Art. no. 042036.
[55] O. B. Sezer and A. M. Ozbayoglu, “Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach,” Appl. Soft Comput., vol. 70, pp. 525–538, Sep. 2018.
[56] H. S. Sim, H. I. Kim, and J. J. Ahn, “Is deep learning for image recognition applicable to stock market prediction?” Complexity, vol. 2019, pp. 1–10, Feb. 2019.
[57] Y. Song, J. W. Lee, and J. Lee, “A study on novel filtering and relationship between input-features and target-vectors in a deep learning model for stock price prediction,” Int. J. Speech Technol., vol. 49, no. 3, pp. 897–911, Mar. 2019.

[58] Y. Song and J. Lee, “Importance of event binary features in stock price prediction,” Appl. Sci., vol. 10, no. 5, p. 1597, Feb. 2020.
[59] X. Tan, S. Li, C. Wang, and S. Wang, “Enhancing high frequency technical indicators forecasting using shrinking deep neural networks,” in Proc. 6th Int. Conf. Inf. Manage. (ICIM), Mar. 2020, pp. 249–256.
[60] M. R. Vargas, B. S. L. P. de Lima, and A. G. Evsukoff, “Deep learning for stock market prediction from financial news articles,” in Proc. IEEE Int. Conf. Comput. Intell. Virtual Environ. Meas. Syst. Appl. (CIVEMSA), Jun. 2017, pp. 60–65.
[61] M. R. Vargas, C. E. M. dos Anjos, G. L. G. Bichara, and A. G. Evsukoff, “Deep leaming for stock market prediction using technical indicators and financial news articles,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2018, pp. 1–8.
[62] Q. Wang, W. Xu, and H. Zheng, “Combining the wisdom of crowds and technical analysis for financial market prediction using deep random subspace ensembles,” Neurocomputing, vol. 299, pp. 51–61, Jul. 2018.
[63] Y. Wen, P. Lin, and X. Nie, “Research of stock price prediction based on PCA-LSTM model,” IOP Conf. Ser., Mater. Sci. Eng., vol. 790, Apr. 2020, Art. no. 012109.
[64] Y. Xu and V. Keselj, “Stock prediction using deep learning and sentiment analysis,” in Proc. IEEE Int. Conf. Big Data (Big Data), Dec. 2019, pp. 5573–5580.
[65] X. Zhou, Z. Pan, G. Hu, S. Tang, and C. Zhao, “Stock market prediction on high-frequency data using generative adversarial nets,” Math. Problems Eng., vol. 2018, pp. 1–11, Apr. 2018.
[66] Li, A. W., & Bastos, G. S. (2020). Stock Market Forecasting Using Deep Learning and Technical Analysis: A Systematic Review. IEEE Access, 8, 185232–185242. doi:10.1109/access.2020.3030226
[67] Lopez de Prado, Marcos. Advances in Financial Machine Learning. Wiley, 2018.