From Technical Indicators to Trading Decisions: A Deep Learning Model Combining CNN and LSTM

SAHIB Mohamed Rida, ELKINA Hamza, ZAKI Taher
Innovation in Mathematics and Intelligent Systems Research Laboratory, Faculty of Applied Sciences,
Ibn Zohr University, Agadir, Morocco

***Abstract*—Stock market prediction is a highly attractive and popular field within finance, driven by the potential for significant profits that come with substantial risks due to data non-linearity and complex economic principles. Extracting features from trading data is crucial in this domain, and numerous strategies have been developed. Among these, deep learning has achieved impressive results in financial applications because of its robust data processing capabilities. In our study, we propose a hybrid deep learning model, the CNN-LSTM, which combines the 2D Convolutional Neural Network (CNN) for image processing with the Long Short-Term Memory (LSTM) network for managing image sequences and classification. We transformed the top 15 of 21 technical indicators from financial time series into 15x15 images for 21 different day periods. Each image is then categorized as Sell, Hold, or Buy based on the trading data. Our model demonstrates superior performance in stock predictions over other deep learning models.**

***Keywords*—Stock market prediction; CNN-LSTM hybrid model; financial time series; technical indicators; CNN; LSTM**

I. Introduction

The global financial markets are characterized by their dynamic nature, where the profit potential is equally matched by the susceptibility to risk. This duality is largely due to the complex interplay of economic indicators, investor sentiment, and global financial events, making stock market forecasting a highly sophisticated area of study. Forecasting these markets requires an understanding of both macroeconomic trends and the minute fluctuations within trading data [1]. As markets evolve, the tools and techniques employed to forecast these changes must also develop, incorporating new data and adapting to changing conditions.

Traditional financial models, such as the Efficient Market Hypothesis and Fundamental Analysis, have long been used to understand and predict market behaviors. However, these models often fall short in times of increased market volatility and when dealing with large unstructured datasets. In contrast, advanced computational techniques, especially those involving machine learning and deep learning, have shown remarkable success in decoding complex patterns that underlie financial markets [2]. These techniques can process vast amounts of data in real-time, learning from new information as it becomes available, which is a crucial advantage in today's fast-paced markets.

Deep learning, a subset of machine learning, has emerged as a transformative force in financial predictions. The deep neural networks, with their multiple layers of processing, can extract high-level features from raw data, which is pivotal in identifying profitable trading opportunities. Specific architectures like Convolutional Neural Networks (CNNs) and Long Short-Term Memory networks (LSTMs) have been at the forefront of this revolution. CNNs are particularly effective in dealing with spatial data, whereas LSTMs excel in capturing temporal dependencies, addressing two critical dimensions of financial data [3].

The approach of combining CNNs and LSTMs aims to harness the strengths of both architectures to improve the accuracy and reliability of financial predictions. This hybrid model leverages CNN's ability to effectively process and analyze images derived from structured data, such as graphs and charts of market trends, and complements it with LSTM's capability to understand time series data, ensuring that temporal sequences in stock prices are accurately predicted. The synergistic combination of these technologies is designed to handle the multifaceted nature of financial datasets more effectively than models employing a single methodology [4].

Despite significant advancements in machine learning techniques, stock market prediction remains a challenging task due to the inherent volatility and complexity of financial markets. Traditional models often fail to capture the nuanced and multifaceted nature of market data, leading to inaccurate predictions. This research aims to address the gap by developing a hybrid model that combines CNNs and LSTMs to enhance the accuracy and robustness of stock trend predictions. How effective is the CNN-LSTM hybrid model in predicting stock trends compared to traditional financial models?

The objectives of this research are to develop a hybrid CNN-LSTM model for stock trend prediction and to evaluate the performance of the hybrid model against standalone deep learning models.

The significance of this research lies in its potential to revolutionize stock market forecasting by leveraging advanced deep learning techniques. By combining CNNs and LSTMs, the proposed model aims to provide more accurate and reliable predictions, which could significantly benefit investors and financial analysts. This research contributes to the field of financial forecasting by demonstrating the effectiveness of hybrid deep learning models and providing insights into their practical applications in dynamic market environments.

The reminder of this paper is organized into several distinct sections to facilitate a thorough exploration of our research. Section II reviews related works, emphasizing the evolution of

predictive models from traditional to modern deep learning approaches. Section III delves into the technologies underpinning our study, particularly CNNs and LSTMs, elucidating their principles and advantages in financial applications. Section IV details our methodology, including data preprocessing, model development, and algorithmic considerations. The empirical evaluation of our model is presented in Section V, where we discuss its performance against traditional and contemporary benchmarks. We conclude in Section VI, summarizing our contributions and proposing future research directions for enhancing predictive models in finance.

II. Related Works

Stock market prediction has long been a central theme in financial research, with various models being developed to forecast market trends and price movements. Historically, predictive models in finance were largely dominated by linear regression and time-series analysis, focusing on historical data to predict future prices. Seminal works by Fama [5] introduced the Efficient Market Hypothesis, suggesting that stock prices reflect all available information and follow a random walk. However, the hypothesis has been challenged by subsequent studies that recognize patterns and trends in market data, suggesting predictability under certain conditions [6].

Stock market analysis relied heavily on statistical methods and basic machine learning models. Time series forecasting techniques such as ARIMA and exponential smoothing were commonly used due to their simplicity and effectiveness in handling linear trends and seasonality [7]. However, these methods often fall short of capturing the complex, non-linear patterns typically exhibited in financial markets.

With the advent of more advanced computational resources, machine learning techniques have gained prominence. Researchers have explored various algorithms from simple decision trees to complex ensemble methods to predict stock prices. A significant contribution in this area was made by Patel et al. [8], who compared different technical indicators with machine learning algorithms and found that models like Random Forest and SVM outperformed traditional statistical methods.

Deep learning has introduced a paradigm shift in predictive accuracy and data processing capabilities. Among the first to apply deep learning to financial forecasting, Dixon et al. [9] demonstrated that deep neural networks could significantly enhance prediction performance over traditional models. The ability of deep learning models to learn complex, non-linear relationships in data offers unprecedented advantages in the noisy, volatile environment of financial markets.

Convolutional Neural Networks (CNNs) have been primarily utilized in image processing but have found applications in financial markets where pattern recognition in chart analysis plays a crucial role [10]. On the other hand, Long Short-Term Memory networks (LSTMs) are a type of recurrent neural network (RNN) ideal for processing sequences of data, making them suitable for analyzing time series data prevalent in stock market predictions [11].

The innovation of combining CNN and LSTM models is relatively recent, with researchers beginning to explore the synergy between spatial feature extraction and sequential data processing. In one notable study, Zhang et al. [12] developed a hybrid model that utilizes CNNs to interpret visual patterns from stock market charts and LSTMs to analyze the temporal patterns in trading data. Their findings suggest that such hybrid models can outperform models based on a single architecture, particularly in handling the multifaceted nature of financial time series data.

Recognizing the limitations of singular approaches, recent research has shifted towards hybrid models that combine the strengths of CNNs and LSTMs. These models leverage CNNs for robust feature extraction from complex input formats, such as images or transformed time series, and LSTMs to interpret these features over time, enhancing the predictive accuracy for various financial applications [13].

One notable study introduced an attention-based hybrid CNN-LSTM model that incorporates the XGBoost algorithm for feature selection and dimensionality reduction, further refining the model's predictions for stock prices [13]. Similarly, Shang et al. [14] employed a CNN-LSTM hybrid model to enhance signal processing capabilities for damage detection in infrastructure, demonstrating the versatility of hybrid models in diverse applications beyond the financial market.

Khalid et al. presents in his study [15] a convolutional deep neural network model leveraging a 2D-CNN for image processing and classification. The image creation process involves transforming top technical indicators from a financial time series, each calculated over 21 different-day periods, to generate images of specific sizes. These images are then labeled as Sell, Hold, or Buy based on the original trading data. In comparison to the Long Short-Term Memory Model and the one-dimensional Convolutional Neural Network, the proposed model demonstrates superior performance. This research underscores the efficacy of employing a convolutional deep neural network with 2D-CNN for processing and classifying financial time series data. The utilization of top technical indicators in image creation contributes to enhanced predictive capabilities, making the proposed model a promising approach for stock price trend prediction.

III. Background

Deep learning has risen to prominence as a pivotal subset of machine learning, renowned for its efficacy across a broad spectrum of applications from image recognition to natural language processing. This method employs multiple layers of neural networks to interpret vast quantities of data, revealing intricate patterns those traditional techniques could not uncover. Among the most influential architectures within deep learning are Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

A. Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) is a specialized type of neural network model designed for processing data that has a grid-like topology, such as images. CNNs are particularly powerful for tasks involving image recognition, classification, and analysis, and have been widely adopted in various

applications ranging from medical imaging to autonomous vehicle technology.

A Convolutional Neural Network typically consists of an input layer, multiple hidden layers, and an output layer "Fig. 1". The hidden layers usually include a series of convolutional layers, pooling layers, and fully connected layers at the end:

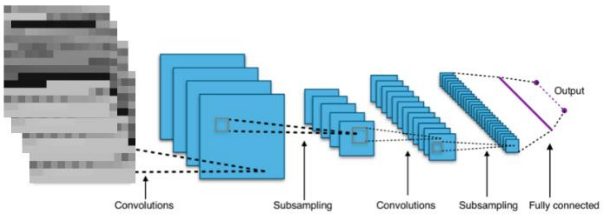

Fig. 1. CNN architecture.

1) Convolutional layers: The core building blocks of a CNN are its convolutional layers, which apply a number of filters to the input. These filters are small matrices used to perform convolution operations that process the data and create feature maps. This process effectively captures spatial hierarchies in data by recognizing patterns such as edges, shapes, and textures within the input images [16]. Mathematically, it is expressed as given in Eq. (1) for a single dimension:

$$(f * g)(t) = \int f(\tau)g(t-\tau)d\tau \tag{1}$$

In the context of CNNs, this is typically simplified to a discrete convolution as shown in Eq. (2), especially for image processing:

$$(f * g)[n] = \sum_{m=-M}^{M} f[m] \times g[n-m] \tag{2}$$

In 2D (for images), it becomes Eq. (3):

$$(I * K)[i,j] = \sum_m \sum_n I[m,n] \times K[i-m, j-n] \tag{3}$$

where:

- I is the input image or feature map.
- K is the kernel or filter.
- m, n index the elements of the kernel.
- i, j index the resulting matrix.

2) Activation function: After a convolution operation, an activation function such as the ReLU (Rectified Linear Unit) as given in Eq. (4) is typically applied to introduce non-linear properties to the system. This helps the network learn complex patterns during training.

$$ReLU(x) = \max(0, x) \tag{4}$$

3) Pooling layers: These layers reduce the spatial size of the convoluted features, helping to decrease the computational load, memory usage, and the number of parameters. Max pooling, which selects the maximum value from the feature region covered by the filter, is a common method used.

4) Fully connected layers: Towards the end of the network, fully connected layers use the features extracted by the convolutional and pooling layers to determine the final output, such as the classification of the image. Each neuron in a fully connected layer has connections to all activations in the previous layer.

5) Output layer: The final layer outputs the prediction of the network using a Softmax or Sigmoid activation function, depending on the task (e.g., multi-class classification or binary classification).

B. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a special kind of Recurrent Neural Network (RNN) that are capable of learning long-term dependencies in data sequences. Introduced by Hochreiter and Schmidhuber in 1997, LSTMs were designed to overcome the limitations of traditional RNNs, particularly problems related to learning long-term dependencies and the vanishing gradient problem during training [17]. LSTMs are particularly well-suited for classifying, processing, and predicting sequences where there are lags of unknown duration between important events. This capability makes them ideal for applications such as time series prediction, natural language processing, and speech recognition.

An LSTM unit "Fig.2" typically consists of a cell state and three gates that regulate the flow of information: the input gate (6), forget gate (5), and output gate (9). Here's how each component works mathematically:

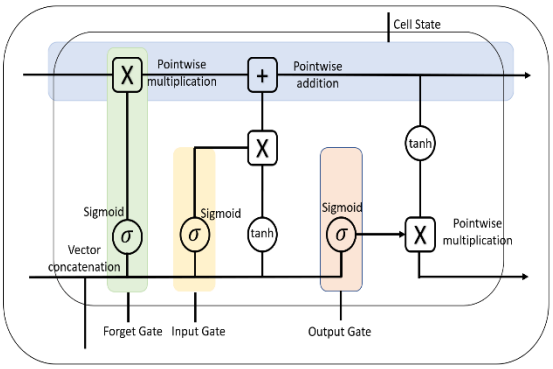

Fig. 2. The structure of LSTM unit.

1) Forget gate: This gate decides what information is discarded from the cell state. σ denotes the sigmoid function, W_f are the weights of the forget gate, h_{t-1} is the previous hidden state, x_t is the input at step t, and b_f is the bias.

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \tag{5}$$

2) Input gate: The input gate decides which values will update the cell state. $\tilde{C}$ (7) represents the candidate values for the state update.

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \tag{6}$$

$$\tilde{C}_t = \tanh(W_C . [h_{t-1}, x_t] + b_C) \quad (7)$$

3) Cell state update: The cell state C_t (8) is updated by forgetting the old state C_{t-1} as regulated by f_t and adding new candidate values scaled by i_t.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (8)$$

4) Output gate: The output gate controls the output of the cell state through the hidden state h_t (10). The actual output $h_t$is filtered by the output gate o_t and then passed through a tanh function to scale the values between -1 and 1.

$$o_t = \sigma(W_o . [h_{t-1}, x_t] + b_o) \quad (9)$$

$$h_t = o_t * \tanh(C_t) \quad (10)$$

C. CNN-LSTM Hybrid Model

The most important and useful deep neural models come from the combining of the different types of networks together into hybrid models. The CNN-LSTM method for the stock market forecasting, composed of a series connection of CNN and LSTM. CNN-LSTM can extract complex features and can store complex irregular trends of stocks market.

In a CNN-LSTM architecture [18], the TimeDistributed layer is used to wrap a convolutional neural network (CNN) so that it can process input data that varies over time, such as frames in a video or a series of images. This layer allows the same CNN model to be applied to each timestep independently and efficiently. Essentially, it acts as a bridge between the CNN and LSTM layers, managing the temporal aspects of the model while preserving spatial feature extraction capabilities of the CNN.

The TimeDistributed layer is a crucial component in neural network architectures where it is necessary to apply the same layer independently to every timestep of input data. This is particularly useful in models that need to maintain temporal order in their inputs, such as CNN-LSTM networks used for sequence prediction tasks that involve spatial data (like videos or time series of images).

1) CNN-LSTM Model with TimeDistributed: In a typical CNN-LSTM setup:

a) Feature Extraction (CNN part): The TimeDistributed wrapper applies the CNN across each timestep. For instance, in video processing [19], each frame (image) of the video passes through the same convolutional layers. This ensures that the spatial features from each frame are extracted in the same way.

b) Temporal Processing (LSTM part): The output from the TimeDistributed-CNN part, now a series of feature vectors (one for each timestep), is then passed to the LSTM layers. The LSTM processes these features over time, capturing dynamic temporal behaviors and interactions between the timesteps, which are crucial for tasks like video classification or predicting sequences of images.

IV. METHODOLOGY

We propose a hybrid analytical model that integrates Convolutional Neural Networks (CNN) and Long Short-Term Memory networks (LSTM) to effectively identify optimal buying and selling points in stock prices. This model employs fifteen selected technical indicators from a set of twenty, each evaluated over various time intervals, to generate representative images. The methodology of our proposed system encompasses five principal stages: data extraction, feature engineering, feature selection, data labeling, and the management of class imbalance, culminating in the creation of images. The primary objective of our research is to accurately determine the most advantageous positions for buy, sell, and hold decisions within the time series data of stock prices.

A. Data Extraction

In our research, the dataset employed comprises several key features that encapsulate the dynamics of the stock market. Specifically, it includes the following attributes: Date, Open Price, Low Price, High Price, Close Price, Adjusted Close Price, and the Trading Volume for each respective date. These features are extracted from the daily stock prices of Apple Inc., sourced from Alpha Vantage, which is known for its comprehensive provision of real-time and historical financial market data. The dataset spans from January 1, 2004, to December 31, 2021, for training purposes, and from January 1, 2022, to December 31, 2023, for testing, allowing a robust assessment of our model's predictive capabilities within the specified periods.

B. Feature Enginnering

Following the extraction of the dataset, our methodology involves calculating 21 technical indicators for each trading day, covering varying intervals ranging from 6 to 27 days. These indicators predominantly fall into two categories: momentum indicators and oscillators. Momentum indicators are used to assess the speed at which stock prices change, providing insights into the strength or weakness of a trend. Oscillators, on the other hand, help determine overbought or oversold conditions by measuring the price momentum and its deviations. This comprehensive analysis of technical indicators enhances our model's ability to accurately predict optimal trading points within the stock market.

1) Moving Average (MA): Shows the average stock price over a specific period of time, smoothing out price data. Eq. (11) shows its calculation.

$$MA = \frac{\sum_{i=1}^{n} P_i}{n} \quad (11)$$

Where P_i is the price at each point and n is the number of points.

2) Exponential Moving Average (EMA): Similar to MA but gives more weight to recent prices, reacting more significantly to recent price changes. Eq. (12) unveils its computational heart.

$$EMA_t = (V_t \times SF) + (EMA_{t-1} \times (1 - SF)) \quad (12)$$

SF is the Smoothing factor is typically $\frac{2}{n+1}$, where n is the number of days.

3) Moving Average Convergence Divergence (MACD) : Indicates the relationship between two moving averages of a stock's price. Eq. (13) and Eq. (14) show the calculations of MACD and Signal Lines:

$$MACD = EMA_{12} - EMA_{26} \tag{13}$$

And the signal line:

$$Signal = EMA_9(MACD) \tag{14}$$

4) Relative Strength Index (RSI): Measures the speed and change of price movements, typically over a 14-day period, to identify overbought or oversold conditions. Eq. (15) provides the calculation of RSI value:

$$RSI = 100 - \frac{100}{1 + RS} \tag{15}$$

where RS (Relative Strength) is:

$$RS = \frac{Average\ Gain}{Average\ Loss} \tag{16}$$

5) Bollinger bands: Consists of a middle band being an N-period simple moving average (SMA) flanked by upper and lower bands at two standard deviations away from the SMA to measure volatility. The inner workings of the Bollinger Bands are detailed in Eq. (17) to Eq. (19):

$$Middle\ Band = MA_{20} \tag{17}$$

$$Upper\ Band = MA_{20} + (2 \times Std_{20}) \tag{18}$$

$$Lower\ Band = MA_{20} - (2 \times Std_{20}) \tag{19}$$

6) Stochastic oscillator: Compares a stock's closing price to its price range over a certain period, indicating momentum and possible trend reversals. Eq. (20) illustrates the specific computation employed by the Stochastic Oscillator:

$$\%K = \frac{C - L_{14}}{H_{14} - L_{14}} \times 100 \tag{20}$$

where, C is the lasted closing price, L_{14} is the low of the 14 previous trading sessions, and H_{14} is the highest price traded during the same 14-day period.

7) On-Balance Volume (OBV): Uses volume flow to predict changes in stock price. Eq. (21) unveils its computational heart:

$$OBV_t = \begin{cases} OBV_{t-1} + Vol_t\ if\ Close_t > Close_{t-1} \\ OBV_{t-1} - Vol_t\ if\ Close_t < Close_{t-1} \\ OBV_{t-1}\ if\ Close_t = Close_{t-1} \end{cases} \tag{21}$$

8) Average Directional Index (ADX): Measures the strength of a trend, regardless of its direction. Eq. (22) illustrates the calculation of ADX:

$$ADX = \frac{SMAoAV(DI^+ - DI^-)}{DI^+ + DI^-} \tag{22}$$

where SMAoAV is the Smoothed Moving Average of the Absolute Value.

9) Accumulation/Distribution Line (A/D Line): Measures the cumulative flow of money into and out of a stock, which can indicate potential price movements. Eq. (23) unveils the A/D's inner workings:

$$A/D = Prev_{A/D} + Vol \times \frac{C - L - (H - C)}{H - L} \tag{23}$$

where Vol is the Volume, C is the close price, L is the Low price, and H is the high price.

10) Ichimoku cloud: Provides more data points, which give a more comprehensive look at resistance and support, as well as momentum and trend direction. One of its components. Eq. (24) shows how Ichimoku Cloud is calculated:

$$Leading\ Span\ A = \frac{ConversionLine + BaseLine}{2} \tag{24}$$

Conversion Line and Base Line involve calculating midpoints of high and low prices over different periods.

11) Standard deviation: Measures the dispersion of a dataset relative to its mean, commonly used to gauge the volatility. Eq. (25) details the SD's calculation:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(P_i - \mu)^2} \tag{25}$$

Where P_i is each individual price and μ is the mean price.

12) Volume Weighted Average Price (VWAP): Gives an average price a stock has traded at throughout the day, based on both volume and price. These VWAP, captured in Eq. (26):

$$VWAP = \frac{\sum(Price \times Volume)}{\sum Volume} \tag{26}$$

13) Momentum: Indicates the rate of change or speed of price movement of a stock. Eq. (27) illustrates its calculation:

$$Momentum = Close_{current} - Close_{n\ periods\ ago} \tag{27}$$

14) Commodity Channel Index (CCI): Determines overbought or oversold levels, helping to identify price reversals. Eq. (28) shows the calculation of CCI.

$$CCI = \frac{TypicalPrice - 2OPeriod\ MA\ of\ TP}{0.015 \times Mean\ Deviation} \tag{28}$$

Typical Price (TP) is the average of the high, low, and close prices.

15) Williams %R: Measures the level of the close relative to the highest high for the look-back period, similar to the Stochastic Oscillator. Eq. (29) details the calculation of Williams %R.

$$\%R = \frac{H_n - C}{H_n - L_n} \times -100 \tag{29}$$

16) Chaikin Money Flow (CMF): Combines price and volume to show where the money is flowing, into or out of a stock. Eq. (30) illustrates the specific computation of CMF

$$CMF = \frac{\sum_1^N \left[\frac{((C-L)-(H-C))}{(H-L)} \times Volume\right]}{\sum_1^N Volume} \quad (30)$$

17) Aroon indicator: Measures whether a stock is trending or not and the strength of the trend. For the mathematically inclined, the inner workings of the Aroon indicator are detailed in Eq. (31) to Eq. (32):

$$Aroon\ Up = \frac{(N - Days\ Since\ Nday\ High)}{N} \times 100 \quad (31)$$

$$AroonDown = \frac{(N - Days\ Since\ Nday\ low)}{N} \times 100 \quad (32)$$

18) Keltner channel: Similar to Bollinger Bands, uses envelopes set above and below an exponential moving average, but the bands are based on the Average True Range (ATR). Eq. (33) to Eq. (35) details the mathematical principles behind the Keltner Channel.

$$Middle\ Line = EMA_{20} \quad (33)$$

$$Upper\ Channel\ Line = EMA_{20} + (2 \times ATR) \quad (34)$$

$$Lower\ Channel\ Line = EMA_{20} - (2 \times ATR) \quad (35)$$

19) Elder's Force Index (EFI): Elder's Force Index combines price movement and volume to measure the strength of bulls and bears in the market. It can indicate potential reversals and price corrections. Eq. (36) details the EFI's calculation.

$$EFI = Volume \times (Current\ Close - Prev\ Close) \quad (36)$$

20) Rate of Change (ROC): The Rate of Change indicator measures the percentage change in price between the current price and the price a certain number of periods ago. It's used to identify the momentum behind price movements. Eq. (37) details the ROC's calculation.

$$ROC = \left(\frac{Current\ close - Close\ n\ periods\ ago}{Close\ n\ periods\ ago}\right) 100 \quad (37)$$

21) Average True Range (ATR): The Average True Range is a technical analysis indicator that measures market volatility by decomposing the entire range of an asset price for that period. ATR is not directional and only measures volatility, making it useful for assessing risk. Eq. (38) and (39) unveil the mathematical principles behind this indicator.

$$TrueRange = Max[|H - L|, |H - PreC|, |L - PreC|] \quad (38)$$

$$ATR = MA\ (True\ Range\ over\ n\ period) \quad (39)$$

C. Feature Selection

In the pursuit of enhancing model performance, a rigorous feature selection process was implemented subsequent to the computation of various indicators. The selection involved two established methodologies: the ANOVA F-value [20][21] method (f_classif) and the Chi-Squared test (chi2) [21]. These methods were employed to identify features with the highest statistical significance in relation to the predictive outcome. An intersection of the features identified by both methods was conducted to ensure the inclusion of the most robust features. Furthermore, the features common to both selection results were organized such that indices were sorted, facilitating the clustering of similar types of indicators. This arrangement aims to maintain spatial coherence when these indicators are represented as images, optimizing the model's ability to discern patterns relevant to the predictive tasks at hand.

D. Labeling the Target

To determine the target labels, a computational algorithm is utilized. This algorithm analyzes a sliding window of 11 days at a time, checking the day that falls in the middle of this window. It assigns a "SELL" label if this day has the highest price in the window, a "BUY" if it has the lowest, and a "HOLD" for all other cases. This method can be used to guide trading decisions, suggesting optimal days for buying or selling based on historical price movements within each window [10].

E. Handling Class Imbalance

Upon labeling our target variables, it was observed that the dataset exhibited significant class imbalance. The "Hold" category substantially outnumbered the "Buy" and "Sell" classes. Addressing class imbalance is a pivotal challenge in machine learning, especially in datasets where the frequency of instances across different classes is markedly disproportionate. Such imbalances can detrimentally affect the performance of predictive models by inducing a bias towards the majority class.

To counteract this issue, several methodologies have been developed and are widely recognized within the research community. These include:

- Oversampling the minority class: This involves artificially augmenting the minority class by replicating its instances until the class distribution is more balanced. A popular method is the Synthetic Minority Over-sampling Technique (SMOTE), which synthesizes new examples rather than duplicating existing ones [22].
- Undersampling the majority class: This method reduces the number of samples in the majority class to balance the class distribution. Care must be taken to ensure that this does not lead to the loss of important information.
- Bagging: Using bagging techniques like Random Forest can help by building multiple decision trees on various sub-samples of the dataset and then averaging the results to improve the model's robustness and balance [23].
- Cluster-based Over Sampling: Techniques that involve clustering the minority class and then performing oversampling within each cluster to maintain intra-class diversity [24].

For the purposes of this study, cluster-based oversampling was selected to address the imbalance within the dataset. This choice was predicated on its efficacy in maintaining the diversity and representativeness of the minority class, thereby enhancing the overall predictive accuracy and reliability of the model.

F. Image Generation

Upon completing the aforementioned procedural steps which encompass dataset acquisition, computation of technical indicators, feature selection, target labeling, and data normalization we proceed to organize the daily tabular data, which consists of 225 features, into an image-like format. This transformation facilitates the application of convolutional neural networks, which are adept at processing image data. "Fig. 3" illustrates sample images, each composed of a 15x15 pixel grid, generated during the image creation phase.

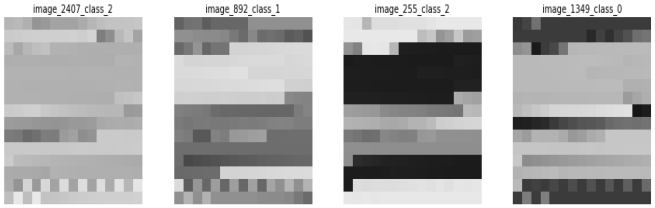

Fig. 3. Sample of images.

In the context of this research, the image dataset comprises a substantial collection of stock price data for Apple Inc. Specifically, the dataset includes approximately 3,481 images designated for training, 1000 images set for testing purposes. This structured division supports a robust framework for evaluating the efficacy of the predictive model under study.

G. CNN-LSTM Architecture

The architecture of the used neural network (see Fig. 4) outlines a hybrid Convolutional Neural Network-Long Short-Term Memory (CNN-LSTM) model, strategically designed to process sequential data that integrates spatial hierarchies. This hybrid model is particularly effective in scenarios where both spatial features and temporal sequences are crucial, such as in video processing, time-series analysis, and complex natural language tasks.

In this model, the data flows through multiple layers, each designed for specific tasks. Initially, spatial features are extracted through time-distributed CNN layers, where each CNN operates independently across different time steps but shares weights. These layers help to capture spatial dependencies within individual time frames of the input data. Subsequent dropout layers are incorporated following each CNN layer to mitigate overfitting by randomly deactivating neurons during training. The outputs are then flattened and sequenced through an LSTM layer, which is adept at understanding and retaining information across time steps, thus capturing the temporal relationships between the extracted features. Finally, the sequential data, now encoded with both spatial and temporal information, is processed through dense layers with another dropout in between to further control overfitting. The last dense layer outputs the final predictions of the model.

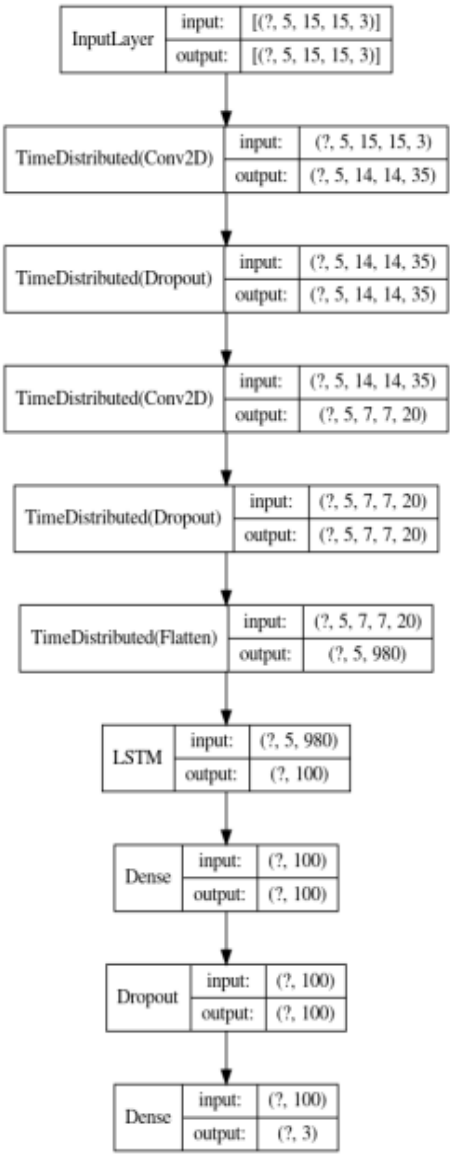

Fig. 4. The architecture of the CNN-LSTM model.

V. Performance and Evaluation

The efficacy of our proposed CNN-LSTM model is assessed primarily through computational evaluation metrics that ascertain how adeptly the classifier distinguishes among the 'Buy', 'Hold', and 'Sell' categories. This assessment involves comparing the labels predicted by the model against the actual stock prices, thereby evaluating the model's practical utility in real-world trading scenarios. The decision to buy, sell, or hold stocks is predicated on these predicted labels, which aim to reflect the optimal trading actions based on the observed data.

Our research employs a sophisticated evaluation methodology for our proposed CNN-LSTM model, utilizing Apple Stock data. The model undergoes rigorous training using the complete dataset, supplemented by cross-validation techniques to ensure generalizability and robustness. The F1 score, a harmonic mean of precision and recall, serves as the primary metric during the training phase, providing a balanced

measure of the model's accuracy in distinguishing between the classes of 'Buy', 'Hold', and 'Sell'.

For the evaluation of test data, we extend our metrics to include a confusion matrix, which offers a detailed visualization of the model's performance across the actual and predicted classifications. This matrix is crucial for understanding the specific types of errors made by the model, such as misclassifications between different trading signals.

Additionally, we utilize the weighted F1 score to account for class imbalance by assigning a weight to each class that reflects its relative importance or frequency. This metric is particularly useful when dealing with skewed class distributions, as it ensures that the performance of the model is not disproportionately influenced by the majority class.

Lastly, the Kappa score, or Cohen's Kappa, is employed to measure the degree of agreement between the actual and predicted classifications, adjusted for the agreement that could occur by chance. This statistical measure provides a more nuanced indication of the model's predictive accuracy and reliability in operational settings.

Together, these metrics furnish a comprehensive framework for evaluating the predictive capabilities of our CNN-LSTM model, ensuring it meets the rigorous standards required for effective stock market trading applications.

On Apple stock data the model gave the following result:

TABLE I. CONFUSION MATRIX OF TEST SET (APPLE)

Actual	**Predicted**		
	Hold	***Buy***	***Sell***
Hold	807	18	12
Buy	32	46	0
Sell	36	0	49

TABLE II. EVALUATION OF TEST SET (APPLE)

	Total Accuracy: 0.86		
	Hold	***Buy***	***Sell***
Recall	0.96	0.59	0.58
Precision	0.92	0.72	0.80
F1-Score	0.94	0.65	0.68
Weighed-F1	0.90		
Kappa score	0.62		

The provided tables elucidate the performance metrics of a classification model dedicated to forecasting stock trading decisions—namely Hold, Buy, and Sell—using Apple stock data. The first table, designated as Table I, presents a confusion matrix that details the accuracy and misclassifications across different trading actions, as predicted by the model. This matrix reveals: For the Hold class, the model achieved substantial accuracy with 807 true positives, while inaccuracies were relatively minor, involving 18 instances predicted as Buy and 12 as Sell. In the Buy category, the model successfully identified 46 instances but incorrectly categorized 32 as Hold, indicating no errors in predicting Buy as Sell. The Sell predictions included 49 correct classifications, but 36 were erroneously predicted as Hold, with no instances misclassified as Buy.

The second table, Table II, provides a comprehensive overview of various evaluation metrics: Total Accuracy stands at 86%, showcasing high overall precision in the model's predictions. The Precision and Recall metrics demonstrate: Exceptional precision (0.92) and recall (0.96) for Hold predictions, indicating the model's efficiency in this category. Moderate precision (0.72) and lower recall (0.59) for Buy predictions, suggesting difficulties in consistently identifying buy transactions. Reasonable precision (0.80) and moderate recall (0.58) for Sell predictions, highlighting some challenges in capturing all actual Sell transactions. F1-Scores further reflect the nuanced performance across categories, with a high of 0.94 for Hold, and lower scores of 0.65 for Buy and 0.68 for Sell, suggesting areas for improvement in balancing precision and recall, particularly for Buy and Sell predictions. The Weighted F1 Score at 0.90% and a Kappa Score of 0.62% suggest a good overall model performance but also room for enhancement, particularly in the precise classification of Buy and Sell actions.

Multi-Layer Perceptron (MLP), Long Short-Term Memory (LSTM), and Convolutional Neural Network (CNN) serve as established methodologies for forecasting stock market movements, and have been selected as baseline models for comparison against our proposed model. The outcomes of these comparisons are detailed in Table III, where the highest Average F1-Score results are highlighted in bold.

TABLE III. THE AVERAGE OF F1-SCORE OF TEST DATA (APPLE) ON DIFFERENT MODELS

Model	***Avg F1-Score***
MLP	0.44
CNN	0.57
LSTM	0.45
CNN-LSTM	**0.76**

VI. CONCLUSION

In this study, we developed and evaluated a hybrid deep learning model combining Convolutional Neural Networks (CNNs) and Long Short-Term Memory networks (LSTMs) for stock market prediction. Our findings demonstrate that this hybrid model outperforms traditional financial models and other deep learning approaches in terms of accuracy and reliability. By effectively processing and analyzing both spatial and temporal dimensions of financial data, the CNN-LSTM model captures complex market patterns and provides robust trading signals.

The superior performance of the hybrid model underscores the potential of integrating advanced machine learning techniques in financial market predictions. This research contributes to the growing body of evidence that deep learning models can significantly enhance the accuracy of financial forecasts, offering valuable insights for investors and traders.

Implications for future research include the exploration of additional hybrid architectures, the incorporation of diverse data sources such as macroeconomic indicators and news sentiment, and the development of real-time analysis capabilities. Additionally, expanding the model's application to other financial markets and improving the interpretability of deep learning models will further enhance their practical utility.

Overall, our study highlights the transformative potential of hybrid deep learning models in financial market analysis, paving the way for more sophisticated and reliable predictive tools in the finance industry.

REFERENCES

[1] Brown, S., Miao, H. (2022). Complex Systems and Stock Market Volatility: New Perspectives on Forecasting Accuracy. Journal of Financial Econometrics.

[2] Turner, J., Lee, C. (2019). From Market Fundamentals to Data Science: Transforming Financial Strategies with Machine Learning. Finance and Technology Review.

[3] Nguyen, D., Tran, Q. (2020). Deep Learning in Financial Markets: A Comprehensive Overview. Artificial Intelligence Review.

[4] Fischer, T., Krauss, C. (2021). Hybrid Deep Learning for Real-Time Financial Data Processing. Journal of Financial Data Science.

[5] A. Cooray, P. Gangopadhyay, and N. Das, "Causality between volatility and the weekly economic index during COVID-19: The predictive power of efficient markets and rational expectations," International Review of Financial Analysis, vol. 89, p. 102792, (2023), doi: https://doi.org/10.1016/j.irfa.2023.102792.

[6] D. Durusu-Ciftci, M. S. Ispir, and D. Kok, "Do stock markets follow a random walk? New evidence for an old question," International Review of Economics & Finance, vol. 64, pp. 165–175, (2019), doi: https://doi.org/10.1016/j.iref.2019.06.002.

[7] Stewart, J.A. (2015). Nonlinear Time Series Analysis.

[8] Patel, J., Shah, S., Thakkar, P., Kotecha, K. (2015). Predicting Stock Market Index Using Fusion of Machine Learning Techniques. Expert Systems with Applications.

[9] Dixon, M., Klabjan, D., Bang, J.H. (2016). Classification-based Financial Markets Prediction using Deep Neural Networks. Algorithmic Finance.

[10] Sezer, O.B., Ozbayoglu, A.M. (2018). Algorithmic Financial Trading with Deep Convolutional Neural Networks: Time Series to Image Conversion Approach. Applied Soft Computing.

[11] Chen, K., Zhou, Y., Dai, F. (2017). A LSTM-based method for stock returns prediction: A case study of China stock market. IEEE International Conference on Big Data.

[12] Zhang, Y., Pei, W., Yang, L. (2019). A CNN-LSTM Hybrid Model for Stock Market Prediction. Journal of Computational Finance.

[13] Zhu, R., Yang, Y., & Chen, J. (2023). XGBoost and CNN-LSTM hybrid model with Attention-based stock prediction.

[14] Shang, L., Zhang, Z., Tang, F., Cao, Q., Pan, H., & Lin, Z. (2023). CNN-LSTM Hybrid Model to Promote Signal Processing of Ultrasonic Guided Lamb Waves for Damage Detection in Metallic Pipelines.

[15] T. Khalid, M. Rida, and Z. Taher, "From Time Series to Images: Revolutionizing Stock Market Predictions with Convolutional Deep Neural Networks," 2024. [Online]. Available: www.ijacsa.thesai.org

[16] H. S. Park and B. K. Oh, (2024). CNN-based model updating for structures by direct use of dynamic structural response measurements, Engineering Structures, vol. 307, p. 117880, doi: https://doi.org/10.1016/j.engstruct.2024.117880.

[17] A. Rahmadeyan and Mustakim, "Long Short-Term Memory and Gated Recurrent Unit for Stock Price Prediction," Procedia Computer Science, vol. 234, pp. 204–212, 2024, doi: https://doi.org/10.1016/j.procs.2024.02.167.

[18] Shi, X., Chen, Z., Wang, H., Yeung, D. Y., Wong, W. K., & Woo, W. C. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. Advances in neural information processing systems, 28,

[19] Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., & Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2625-2634).

[20] A. P. Mercado Rueda, "Chapter 31 - Analysis of variance: ANOVA," in Translational Sports Medicine, A. E. M. Eltorai, J. A. Bakal, S. F. DeFroda, and B. D. Owens, Eds., in Handbook for Designing and Conducting Clinical and Translational Research. , Academic Press, (2023), pp. 157–160. doi: https://doi.org/10.1016/B978-0-323-91259-4.00099-0.

[21] A. F. Siegel and M. R. Wagner, "Chapter 17 - Chi-Squared Analysis: Testing for Patterns in Qualitative Data," in Practical Business Statistics (Eighth Edition), A. F. Siegel and M. R. Wagner, Eds., Academic Press, (2022), pp. 531–547. doi: https://doi.org/10.1016/B978-0-12-820025-4.00017-8.

[22] P. Tyagi, J. Singh, and A. Gosain, "Whale Optimization-based Synthetic Minority Oversampling Technique for Binary Imbalanced Datasets," Procedia Computer Science, vol. 235, pp. 250–263, (2024), doi: https://doi.org/10.1016/j.procs.2024.04.027.

[23] J. Sun, J. Li, and H. Fujita, "Multi-class imbalanced enterprise credit evaluation based on asymmetric bagging combined with light gradient boosting machine," Applied Soft Computing, vol. 130, p. 109637, (2022), doi: https://doi.org/10.1016/j.asoc.2022.109637.

[24] Q. Zhou and B. Sun, "Adaptive K-means clustering based under-sampling methods to solve the class imbalance problem," Data and Information Management, p. 100064, (2023), doi: https://doi.org/10.1016/j.dim.2023.100064.