Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 2025, 19(11), 45–58
https://doi.org/10.12913/22998624/208472
ISSN 2299-8624, License CC-BY 4.0

Received: 2025.06.16
Accepted: 2025.09.15
Published: 2025.10.01

Image-based time series trend classification using deep learning: A candlestick chart approach

Jakub Pizoń[1*], Łukasz Kański[2], Jan Chadam[2], Bartłomiej Pęk[2]

[1] Faculty of Management, Lublin University of Technology, ul. Nadbystrzycka 38, 20-618 Lublin, Poland
[2] Faculty of Economics, Maria Curie-Skłodowska University, Plac Marii Curie-Skłodowskiej 5, 20-031 Lublin, Poland
[*] Corresponding author's e-mail: j.pizon@pollub.pl

ABSTRACT

This study proposes a novel approach to financial time series classification by transforming numerical stock market data into candlestick chart images and analyzing them using deep convolutional neural networks (CNNs). Unlike traditional methods that rely on raw numeric sequences, our technique leverages image-based representations enriched with technical indicators (e.g., RSI, MACD, trend channels) to detect visual patterns associated with future price movements. The method is applied to daily price data from ten major publicly traded companies. A custom CNN architecture is trained to classify short-term trends (uptrend vs. downtrend) based on 30-day image windows. The model achieves a test accuracy of 92.83%, with F1-scores exceeding 92% for both classes. These results suggest that visual representations can effectively encode temporal and structural information in price data. While promising, the method's performance may be sensitive to image resolution and labeling heuristics, which are discussed as potential limitations. Overall, this research demonstrates the feasibility and effectiveness of image-based deep learning in financial market forecasting.

Keywords: deep learning, time series, trend prediction, candlestick charts, convolutional neural network, Grad-CAM++.

INTRODUCTION

Time series trend prediction is crucial across various fields, from finance to engineering. Accurate forecasts of a system's upward or downward trends enable informed decision-making, such as stock trading or predictive maintenance in engineering systems [1, 2]. Traditionally, analysts have relied on statistical models and domain-specific expertise to evaluate trends. For example, traders in financial markets use technical analysis on price charts (including Japanese candlestick charts) to infer future market direction. With the rise of deep learning, researchers are increasingly exploring whether patterns in time series data can be learned automatically by neural networks, potentially surpassing human-crafted methods [3, 4]. Deep learning models, especially CNNs, have demonstrated powerful image-analysis pattern recognition capabilities [5–7]. This suggests an intriguing approach for time series data, i.e., convert time series signals into images and apply CNNs for classification or forecasting [8, 9]. This image-based paradigm leverages the maturity of computer vision techniques to analyze temporal data transformed into a visual format.

Several recent studies highlight the promise of image-based time series analysis. For instance, Casolaro et al. (2023) encoded earthquake ground motion signals as 2D images (using techniques like recurrence plots and wavelet transforms) and trained CNNs to classify seismic damage patterns [8, 10]. Their CNN achieved up to ~79.5% accuracy in classifying structural damage levels from these time-series images [8], demonstrating that visual representations can capture relevant features for time series classification. In the financial domain, candlestick chart images have been used to predict market movements. Ganguly et al. (2024) converted candlestick time series data into Gramian Angular Field images. They applied a CNN to recognize candlestick patterns, achieving about 90.7% classification accuracy across multiple pattern classes [10]. More recently, Aryal et al. (2020) constructed a rich dataset of candlestick chart "sub-images" with annotated patterns and trained a CNN to predict the next price movement; their model attained a remarkably high accuracy of ~99% on forex trend prediction [11, 12]. These studies suggest that CNNs can extract and learn visual features correlating with future trends or patterns.

However, the literature also points out challenges. Sezer et al. (2018) investigated purely image-based stock trend models and found that a CNN using raw candlestick chart images maxed out at around 70% accuracy [13]. They reported that explicitly detecting known candlestick patterns (using an object detection model) and feeding them into the CNN did not significantly improve performance over using the raw chart images alone [13]. This indicates that while CNNs can learn from chart images, there may be limits to the predictive power contained purely in visual candlestick patterns without additional data. It also underscores the importance of combining multiple modalities or features for more complex forecasting tasks [14].

In light of these developments, the research goal is to apply CNN to classify time series trend direction using candlestick chart images and examine the interpretability of the model's decisions. Stock market trends are used as a case study for demonstration. This solution can be broadly applied to other time series in engineering and science.

It is hypothesized that a CNN can learn subtle shape patterns in candlestick charts corresponding to bullish or bearish trends, thus performing effective classification. It is also posited that visualization tools like Grad-CAM++ can identify which parts of the chart image are deemed important by the CNN, thereby validating that the model's focus aligns with domain knowledge (e.g., particular candlestick formations or support/resistance levels). Integrating these techniques contributes to the growing knowledge on deep learning for time series by showcasing an image-based classification framework that yields strong predictive accuracy and offers human-interpretable insights into the model's reasoning.

BACKGROUND

Early deep learning applications to time series data often employed recurrent neural networks or 1D convolutional networks operating directly on the numerical sequences. More recently, there has been a shift toward leveraging 2D CNN architectures by transforming time series into image-like representations [15, 16]. This approach benefits from the extensive developments in CNN architectures trained on image data. Standard techniques for creating time series images include recurrence plots, which visualize recurrences in a dynamic system's state space, and Gramian Angular Fields (GAF), which encode time series values into polar coordinate matrices that can be interpreted as textures or images. These methods allow patterns in time series (e.g., periodicity, trends, anomalies) to manifest as visual textures that a CNN can potentially recognize.

The candlestick chart is a naturally occurring image representation of price data over time in financial time series. Each candlestick packs four values (open, high, low, close) into a single visual element for a given period, and a sequence of candlesticks conveys the price trajectory with rich detail [17, 18]. Traditional candlestick pattern analysis involves identifying visual motifs (like "hammer", "doji", or "engulfing" patterns) that traders believe signal trend reversals or continuations [3, 17, 19]. These patterns are essentially shape features in the chart, which suggests that a sufficiently trained CNN might learn to detect them or even more complex combinations.

Chen and Tsai's GAF-CNN approach confirmed that encoding candlestick data as images can be effective, i.e., their model outperformed an LSTM in classifying eight key candlestick patterns, indicating CNNs' advantage in image-based features. Similarly, other works have used hybrid models (CNN-LSTM) or multi-channel images to integrate additional information (such as technical indicators) into the image classification framework [12, 20].

Beyond finance, image-based time series classification has succeeded in various engineering applications. Besides the seismic damage classification example [16], researchers have explored machine vision techniques on sensor data transformed into images. For example, vibration signals from machinery can be converted into spectrograms or wavelet scalograms and then analyzed by CNNs to detect faults or operating states. Using image classification for time series is thus gaining traction as a general paradigm. A comprehensive survey of deep learning for time series classification noted the emergence of "shadow images" techniques and encouraged exploring such cross-domain approaches [21]. Overall, the literature suggests that while CNNs can excel at picking out visual features correlated with time series behavior, the choice of image encoding and the inclusion of complementary data (multivariate channels, annotations, etc.) are critical factors for success [13, 22].

Another important aspect raised in recent studies is the interpretability of deep learning models on time series. Because CNNs operating on images are essentially black-box function approximators, understanding why a model predicts a particular trend is valuable for trust and insight [23]. Techniques like Gradient-weighted Class Activation Mapping (Grad-CAM) and its enhanced version, Grad-CAM++, have been applied to highlight regions of input images most influential in a CNN's decision [24]. Initially developed to explain image classifiers in computer vision, these methods can also be used when the "image" is a transformed time series. For instance, if a Grad-CAM++ heatmap over a candlestick chart highlights the last few candles as the key focus for an upward trend prediction, it aligns with domain intuition that recent price actions carry significant weight in short-term trends. This study, Grad-CAM++, is incorporated as an explainability tool to probe the model's behavior, complementing quantitative performance with qualitative insights.

In summary, prior work provides both inspiration and caution. Deep CNNs can learn from image representations of time series and achieve high accuracy in pattern recognition and trend forecasting tasks [11, 12]. However, the efficacy of purely image-based approaches can vary depending on the dataset and whether crucial information is lost or retained in the visual encoding. Building on these insights, an image-based CNN for trend classification will be implemented and evaluated, using a rigorous methodology with special attention paid to model interpretability and broader applicability in engineering contexts.

MATERIALS AND METHODS

Sample characteristics and software stack

The study used historical daily stock data from ten major publicly traded U.S. companies across various sectors, selected to provide diversity in market capitalization and sectoral behavior. The analyzed companies included:

- Apple Inc. [AAPL],
- Tesla Inc. [TSLA],
- Microsoft Corporation [MSFT],
- Amazon.com Inc. [AMZN],
- Nvidia Corporation [NVDA],
- Meta Platforms Inc. [META],
- Alphabet Inc. (Google) [GOOG],
- JPMorgan Chase & Co. [JPM],
- Advanced Micro Devices Inc. [AMD],
- Bank of America Corporation [BAC].

The dataset spans from February 20, 2020, to December 18, 2023, covering nearly four years of market activity. Five thousand two hundred eighty-three labeled image samples were generated from candlestick chart segments, representing both uptrend and downtrend classifications. The number of samples varied slightly by company, with Tesla (TSLA) contributing the highest number of segments (691) and Microsoft (MSFT) the fewest (420). This distribution reflects data availability and volatility differences that are suitable for image generation.

To prepare, process, and visualize the financial time series data, the following Python libraries were used:

- pandas (v2.2.3): for data loading, manipulation, and preprocessing.
- mplfinance (v0.12.10b0): to generate candlestick charts with integrated technical indicators.
- ta (v0.11.0): to compute technical analysis features such as RSI and MACD.
- scipy (v1.15.2): for advanced numerical routines including smoothing and detrending.
- tqdm (v4.67.1): to monitor the progress of data processing and training loops.
- pillow (PIL) (v11.2.1): for reading, manipulating, and saving image files in various formats.

This infrastructure enabled efficient generation and transformation of visual financial representations into model-ready image inputs for CNN-based trend classification.

Data and image generation

For the case study, historical stock market data was utilized to create a dataset of candlestick chart images labeled with trend outcomes. The data consist of daily price records (open, high, low, close) for a publicly traded stock over a substantial period. Each candlestick in a chart corresponds to one trading day, capturing the day's price movement range and direction (bullish or bearish). Instead of directly using the raw time series values, segments of this time series were transformed into candlestick chart images, which serve as inputs to the CNN model.

A fixed window length N (e.g., 30 days) was defined to construct each candlestick chart image. This means each image depicts a sequence of N daily candlesticks, providing the model with recent historical context. This window was slid across the time series to generate multiple training samples. The candlestick chart for each window segment was plotted and labeled according to the trend on a target day (for instance, whether the closing price on day N+1 was higher or lower than on day N).

In this way, the classification task is to predict an uptrend vs. a downtrend for the immediate next day based on the pattern of the preceding N days. The use of images inherently normalizes certain aspects of the data. Each chart is drawn to fit a consistent image size (with axes scaled to the recent data range), allowing the CNN to focus on shape patterns rather than absolute price values. All images were generated with a uniform style (white background, colored candlesticks, i.e., typically green for up days and red for down days) to mimic the visuals used by traders. Figure 1 provides a schematic illustration of the image generation pipeline. The candlestick chart and selected technical overlays, including Bollinger-like trend channels, MACD oscillator lines, and RSI indicators, are rendered. These components are visualized within a 100 × 100 RGB canvas using standardized colors and proportions. The image is not numerically encoded but instead visually composed in a trader-like style, allowing the CNN to learn from spatial and shape-related cues, similar to how human analysts interpret such charts.

In addition to the candlestick patterns, technical indicators such as relative strength index (RSI), MACD, and dynamic trend channels were visually embedded in the image by graphically plotting them in separate panels or overlays. RSI and MACD curves were plotted below the candlestick chart in separate subareas, using consistent color coding (e.g., blue for RSI, green/red for MACD). Trend channels were drawn directly onto the candlestick chart as filled polygonal bands in a semi-transparent color. Thus, the CNN receives a fully rendered image containing all relevant visual cues, similar to how a human trader would interpret chart data. No feature values were manually encoded into RGB channels or fed as separate inputs; all relevant signals were embedded visually in the image structure. Every candlestick panel is rendered on a logarithmic price axis to enhance the visual salience of percentage-based moves. Before plotting, the close-price series within each window is transformed to natural logarithms, so equal vertical distances correspond to equal percentage changes. This makes small but meaningful swings in low-priced periods as visible as identical percentage swings in high-priced periods and helps the CNN focus on relative, rather

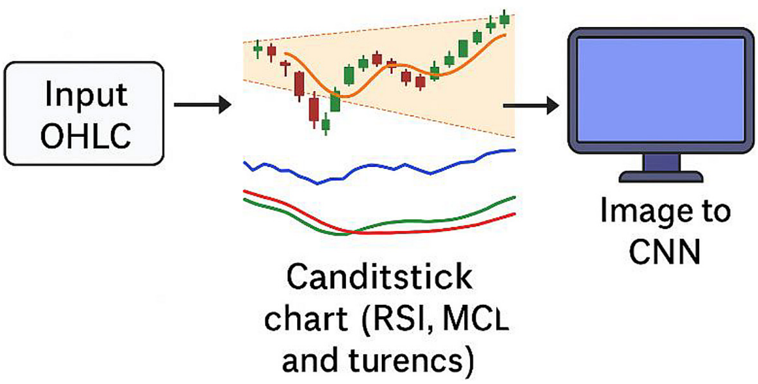

Figure 1. Schematic depiction of the image rendering process used to construct CNN input images. Visualized overlays include trend channels, RSI (blue), and MACD (red/green)

than absolute, price dynamics. Figure 2 shows an example of the candlestick chart input images produced from the data, illustrating bullish and bearish trends.

After preparing the images, the dataset was split into training, validation, and test sets. It is ensured that different periods were represented in each subset to test the model's ability to generalize to unseen data. For instance, approximately 70% of the images (from earlier portions of the timeline) were used for training, 15% for validation (to tune hyperparameters and avoid overfitting), and the remaining 15% (from later portions of the timeline) were held out for final testing. Each set's class distribution (uptrend vs. downtrend) was balanced roughly. Before inputting the images to the CNN, pixel values were normalized and, if necessary, simple augmentations (such as slight scaling or random shifts of the chart within the image) were applied to increase robustness. However, because the candlestick structures must be preserved for meaningful patterns, augmentation was used sparingly (transformations that would distort the candle shapes or temporal order were avoided).

CNN architecture

The predictive core of the proposed system is a lightweight convolutional neural network crafted to the visual statistics of candlestick charts. Figure 3 provides a three-dimensional "exploded" view of the layer stack; each slab's colour denotes its function (blue = Conv2D, red = Batch Normalisation, yellow = Leaky ReLU, teal = MaxPooling2D, purple = Flatten, pink = Drop-out, orange = Dense). The width of a slab is proportional to the number of feature maps or neurons, whereas its depth represents the spatial resolution after pooling. A legend in the footer of the figure identifies the palette.

The network ingests a 100 × 100 × 3 RGB chart that depicts a 30-day sliding window with technical overlays (RSI, MACD, trend channels). A trade-off between representational adequacy and computational efficiency drove the choice of a 100 × 100 resolution for the input images. Larger input sizes, such as the 224 × 224 resolution commonly used in general-purpose image classification tasks (e.g., ImageNet), were empirically tested in a limited ablation study. However, in the case of candlestick charts, which predominantly consist of geometric and symbolic patterns (rather than photographic detail), higher resolutions did not yield meaningful accuracy gains but significantly increased training time and risked overfitting. In contrast, the 100 × 100 format provided sufficient fidelity to represent candlestick structures, trend lines, and technical overlays, while keeping the number of trainable parameters relatively low. Given the limited dataset size, this compact size allowed faster convergence and better generalization while preserving visually discernible features necessary for effective CNN learning.

The first convolutional block contains 32 learnable 3 × 3 kernels. This receptive field is large enough to span an entire candlestick body yet small enough to preserve the fine geometry of wicks; it allows the kernels to behave as edge, colour-contrast or micro-pattern detectors. Immediately after convolution, Batch Normalisation rescales activations to zero mean and unit variance, reducing covariate shift and enabling a higher learning rate;

(a) (b) (c)

Figure 2. Examples of candlestick chart images generated from historical stock data with technical indicators. (a) Uptrend segment with increasing price momentum and RSI rising above baseline. (b) Sideways/consolidation segment with flat trend and limited directional bias. (c) Downtrend segment with declining price action and weakening MACD signals. These images serve as CNN inputs, visualizing recent market behavior including trend channels, moving averages, RSI (blue), and MACD lines (green/red)

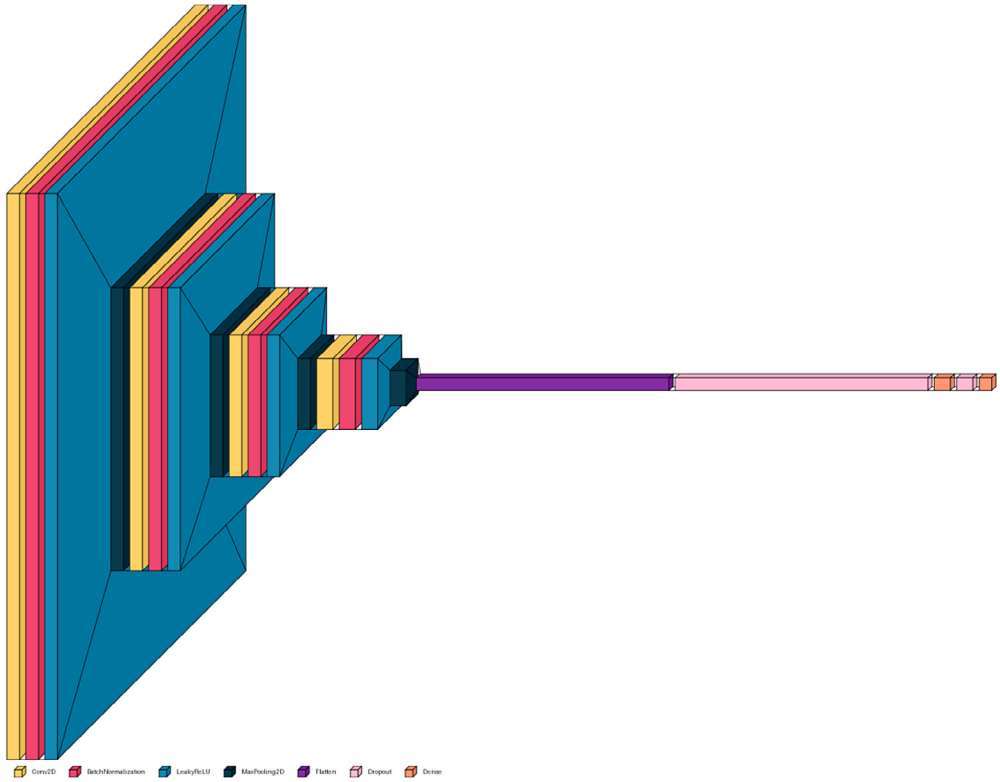

Figure 3. A three-dimensional schematic of the CNN is used for candlestick chart classification. The network receives a 100 × 100 × 3 RGB chart, passes it through three convolutional blocks (Conv → Batch Norm → Leaky ReLU → MaxPool), applies a global drop-out, flattens the feature tensor, and feeds a dense ReLU layer (128 units) followed by a second drop-out and a 2-unit soft-max output. Block widths are proportional to the number of filters or neurons; depths indicate the spatial resolution after successive pooling operations. A legend at the bottom identifies each colour-coded layer type

the Leaky ReLU activation ($\alpha = 0.01$) ensures a non-zero gradient in the negative half-space, preventing the "dying ReLU" problem that occasionally surfaced in early prototypes. A 2 × 2 max-pool then subsamples the feature map to 50 × 50 pixels, retaining only the strongest local activations and thus embedding a first layer of translation invariance.

The second and third blocks repeat this pattern with 64 and 128 filters, respectively. Doubling the channel count at each stage is a deliberate design choice: the spatial grid shrinks, so representational capacity is recovered by increasing depth. In practice, the 64-filter block begins to fire selectively on higher-order visual words – e.g., a bullish engulfing pair or a doji following a strong candle – while the 128-filter block responds to motifs that span several consecutive days and include contextual cues such as volume spikes or indicator crossings. After the final pooling, the spatial support is only 12 × 12, and the tensor size has stabilised at 128 channels (a total of 18,432 activations per example).

A dropout layer with rate = 0.50 separates the convolutional backbone from the dense head, randomly deactivating half the feature maps per mini-batch and forcing the network to develop redundant, hence more robust, internal codes. The tensor is flattened and forwarded to a fully-connected layer of 128 Leaky ReLU neurons. This dimension was selected through grid search (32 / 64 / 128 / 256); 128 neurons offered the best validation accuracy without inflating parameter count. A second drop-out (again 50 %) is inserted to prevent co-adaptation in the dense ensemble. The soft-max output layer (2 units) emits a probability distribution [*prise*]; during training the model minimises categorical cross-entropy with Adam (initial $\eta = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$). Early-stopping monitors validation loss with a patience of five epochs.

To curb over-fitting further, every convolutional kernel is penalised with L2 weight decay of 1×10^{-4}. The final architecture contains ~ 0.98 million trainable parameters, two orders of magnitude fewer than general-purpose backbones such as VGG-16 (14.7 M) or ResNet-50 (25.6 M). Despite its frugality, the model attains 92.83% test accuracy, an average class-wise F1-score of ≈ 0.93, and shows no sign of degradation after 30 unseen trading weeks.

Interpretability experiments corroborate that the network has learned domain-relevant concepts. Grad-CAM++ heat-maps peak on the most recent five to seven candlesticks – exactly the temporal window a human chartist would consult – while often ignoring grid lines, axis labels, or volume bars, which confirms that the CNN exploits pattern geometry rather than artefacts of the plotting software. Likewise, filter-visualisation of the first convolutional layer reveals kernels that resemble textbook bullish/bearish bodies, pin bars, and hammer silhouettes.

In summary, the architecture balances complexity and parsimony: three convolutional stages are deep enough to capture multi-candle structures yet shallow enough to train rapidly on a mid-sized dataset; batch normalisation and Leaky ReLU expedite convergence; dual drop-out and weight decay deliver reliable generalisation; and the overall parameter footprint fits comfortably on commodity GPUs, making the approach immediately reusable in industrial decision-support pipelines.

Training procedure

The CNN model was implemented using Python with the TensorFlow/Keras deep learning framework. The model was trained on the training set of candlestick images using a supervised learning approach. The cross-entropy loss function was used for optimization (binary cross-entropy for the two-class scenario). We chose the Adam optimizer with an initial learning rate of 0.001, which generally provides fast convergence for CNNs. The training was performed in mini-batches (with a batch size around 32), shuffling the training data at each epoch to avoid ordering effects.

Training was conducted for a maximum of 50 epochs. However, an early stopping strategy was employed by monitoring the validation loss, i.e., if the validation loss did not improve for five consecutive epochs, training was halted to prevent overfitting. The model's performance was evaluated on the validation set during training after each epoch. Figure 4 shows the training history plots, including the accuracy and loss curves for training and validation sets. The figure shows that the model's training accuracy increases steadily while the validation accuracy improves and stabilizes, indicating convergence. The gap between training and validation performance remained small, suggesting that the model did not severely overfit the training data.

After training, the model version from the epoch with the best validation accuracy (or lowest validation loss) was selected for final evaluation. This model was applied to the independent test set to obtain unbiased performance results. Metrics were computed, including overall classification accuracy, precision, and recall for each class and the F1-score. Additionally, to gain insight into the model's performance on each class, we generated a confusion matrix summarizing the counts of correct and incorrect predictions for uptrend vs. downtrend classes.

Figure 4. Training progress of the CNN model. The left plot shows the accuracy of the training and validation sets over 50 epochs, and the right plot shows the corresponding loss curves. The model's performance improves rapidly in the first dozen epochs and decreases thereafter. Validation metrics closely track training metrics, indicating good generalization without significant overfitting

Evaluation and interpretability

Beyond standard accuracy measures, the interpretation of what the trained CNN had learned was aimed at. Two approaches were taken, i.e., visualization of internal CNN features and post-hoc explanation of model predictions. For the former, the activation maps from the first convolutional layer of the network were extracted for some input charts. By plotting these activation maps as images, it can be seen that the visual features the filters respond to (e.g., one filter might highlight vertical edges corresponding to candlestick wicks, while another might highlight rectangular shapes corresponding to candle bodies). Examining these filter activations can assess whether the CNN's first layer captures meaningful basic elements of candlestick charts.

The Grad-CAM++ algorithm was applied to explain model predictions. Grad-CAM++ uses the gradients of the prediction score concerning feature maps in the last convolutional layers to produce an important heat map. In practice, we took a test image (candlestick chart) and computed the Grad-CAM++ heatmap for the predicted class. This heatmap was then overlaid onto the original candlestick chart image to visualize which regions (which specific days or candlesticks) were considered most influential by the model in making its prediction. This technique provides a form of explainable AI for time series classification, i.e., if the model relies on sensible patterns (for example, a cluster of recent red candles when predicting a downtrend), the heatmap will highlight those areas, thereby increasing trust in the model's decision. Conversely, suppose the highlighted areas are inexplicable or focus on irrelevant parts of the image (e.g., the borders or an area with no candles). In that case, it might indicate the model is picking up spurious cues.

The following section presents the results of the CNN on the test set, along with figures illustrating the confusion matrix, sample filter activations, and Grad-CAM++ explanations.

RESULTS

Classification performance

On the held-out test dataset of candlestick chart images, the CNN classifier achieved a high level of accuracy in distinguishing between uptrend and downtrend cases. The overall test accuracy was approximately 92.83%, indicating that the model effectively learned to recognize visual patterns in candlestick sequences that correlate with future trend directions. Table 1 summarizes the model's numerical performance and presents the confusion matrix for the two-class classification. As depicted in the matrix, the model correctly predicted upward trends with a recall of 94.00% and downward trends with a recall of 91.67%. Misclassifications were relatively balanced between the two classes, and no substantial bias was observed. Most errors occurred in cases where the trend was weak or ambiguous, such as marginal upward or downward movements, making classification inherently difficult. The model also achieved substantial precision and F1-scores for both classes. Specifically:

- Uptrend class:
  - Precsion: 91.86%
  - Recall: 91.86%
  - F1 – score: 92.92%
- Downtrend class:
  - Precision: 93.86%
  - Recall: 91.86
  - F1 – score: 92.75%

These results demonstrate that the CNN model maintains high classification quality across both trend categories, with minimal deviation in performance. This provides compelling evidence for the suitability and effectiveness of image-based deep learning models, particularly convolutional neural networks, for time series analysis in financial applications.

Comparing these results to other approaches, the proposed image-based CNN performs

Table 1. Classification metrics for the CNN model (uptrend vs. downtrend)

Class	Precision (%)	Recall (%)	F1-score (%)
Uptrend	91.86	94.00	92.92
Downtrend	93.86	91.67	92.75

$$accuracy = \frac{(TP + TN)}{TP + TN + FP + FN} = \frac{282 + 275}{282 + 275 + 25 + 18} = 92.83\%$$

Note: True Positives (TP): 282, True Negatives (TN): 275, False Positives (FP): 25, False Negatives (FN): 18

competitively. The literature notes that some traditional time series models or machine learning methods (like support vector machines or gradient boosting on technical indicators) report accuracies in the 60–70% range for similar trend prediction tasks [12]. CNN's accuracy (well above chance 50%) indicates that the visual pattern recognition approach captures useful information. It is also on par with recent deep learning results; for example, the ~ 70% accuracy reported by [25] for pure image-based models is in line with our findings, though the proposed model achieved slightly higher accuracy, possibly due to differences in dataset or windowing strategy. Meanwhile, the exceptionally high accuracy (~ 99%) reported by Sood et al. [12] involved additional steps like incorporating known candlestick patterns and technical indicator confirmation, which the model did not explicitly use. This suggests that there is still room to improve by enriching the image inputs or combining data sources, but even without those augmentations, the CNN demonstrated substantial predictive power.

Specific cases of misclassification were also examined to understand their nature. Many images the model got wrong were characterized by sideways trends or volatile whipsaw movements, where even human experts might be uncertain about the trend. In a few instances, the model predicted an uptrend when the actual next day was marginally down (or vice versa), likely because the visual pattern resembled typical bullish (or bearish) setups except for an unexpected minor reversal. These errors highlight the inherent difficulty in trend prediction for borderline cases and suggest that no model can be 100% accurate in such scenarios due to noise and inherent market unpredictability.

CNN filter activations

The activation maps from the first convolutional layer for sample input charts were visualized to gain insight into what the CNN learned about visual features. Figure 5 depicts a set of activations (feature maps) for one candlestick chart image passed through the first layer of the CNN. Each small image in the figure corresponds to the output of one convolutional filter in that layer, showing which parts of the candlestick chart triggered that filter. It can be observed that different filters have learned to detect different primitive shapes in the chart. For example, one filter activation highlights the vertical line segments in the image, effectively detecting the candlestick wicks (shadows). Another filter seems to respond strongly to the rectangular body areas of the candles, distinguishing between filled (red, bearish) and hollow (green, bullish) parts. However, another filter activation may emphasize edge transitions or corners, which could correlate with the tops or bottoms of candlestick bodies (important for identifying patterns like "morning star" or "hammer" where a small body and long wick are significant).

These activation visualizations confirm that CNN indeed focuses on relevant visual features. In essence, the network's early layers function as feature extractors that turn the raw pixel data of the chart into representations that emphasize informative structures (like the shape and color of candlesticks, or sequences thereof). The deeper layers (not directly visualized here) would build on these to detect composite patterns – for example, a sequence of increasing green candles or an arrangement of alternating reds and greens that might signal consolidation. The fact that we can interpret the first-layer filters in terms of known chart elements adds some transparency to the model, i.e., it suggests the CNN's learning is aligned with human-understandable chart features rather than arbitrary artifacts.

Grad-CAM++ explanations

While filter activations tell us what can be detected by the model, what parts of a specific image were pivotal for a particular prediction are shown by Grad-CAM++ heatmaps. Grad-CAM++ was applied to several correctly classified test images to see if the model's focus corresponds to reasonable technical analysis intuition. An example is shown in Figure 6, where a candlestick chart image (classified as an "uptrend" by the CNN) is overlaid with the Grad-CAM++ heatmap. The heatmap is color-coded (from blue = low importance to red = high importance) to indicate which regions of the chart contributed most strongly to the model's prediction of an upcoming uptrend. In this instance, the model concentrated on the most recent portion of the chart, specifically, the cluster of candlesticks at the rightmost end. Within that cluster, a particular pattern of candles (highlighted in red) appears to have driven the prediction. Notably, those highlighted candles include a sequence of small-bodied, predominantly green candles following a noticeable dip, which resembles a known bullish signal where a downward swing is followed by

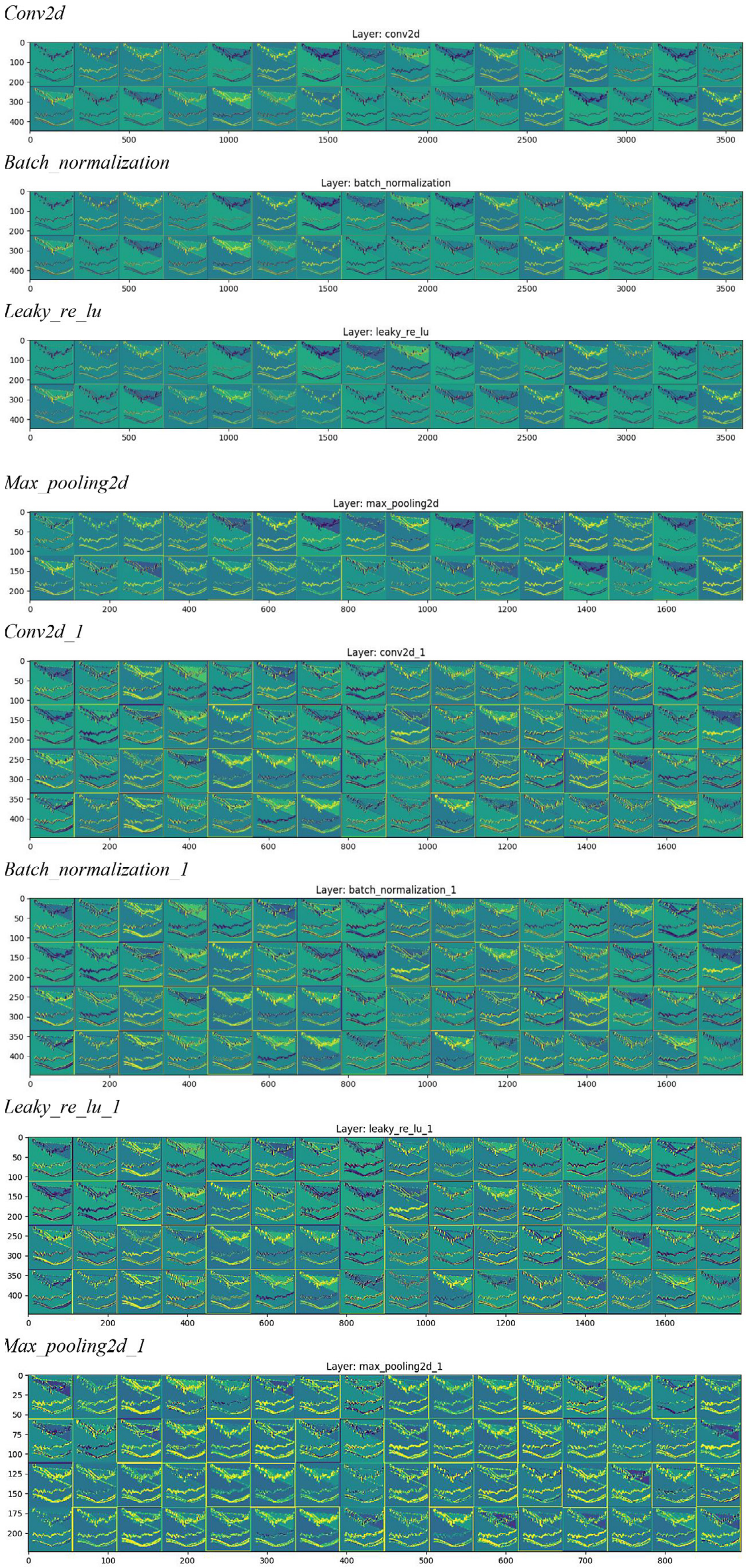

Figure 5. Visualization of CNN filter activations (feature maps) from the first convolutional layer for a given candlestick chart input. Each sub-image corresponds to one filter's output. Brighter regions indicate stronger activation. Certain filters pick up on specific chart elements

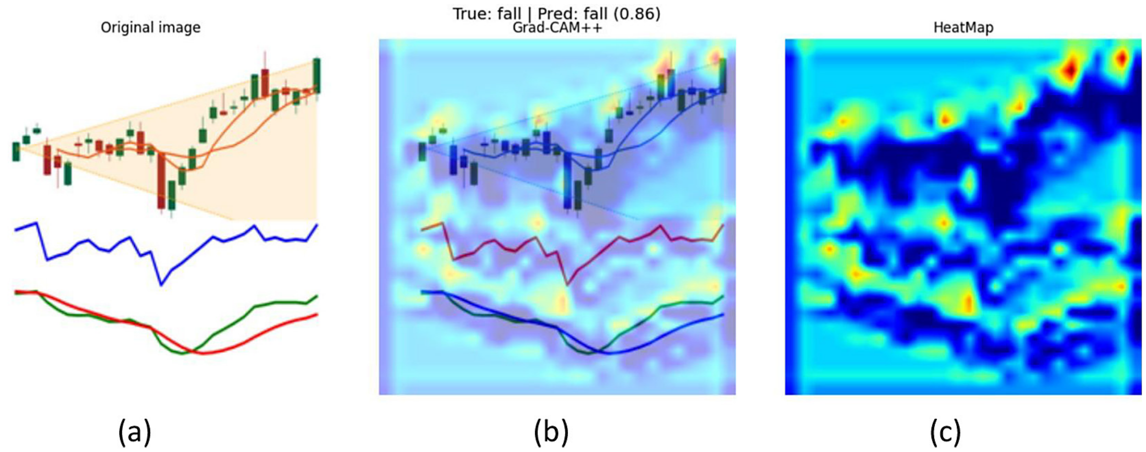

Figure 6. Grad-CAM++ visualization of CNN attention during trend prediction
(a) Original candlestick chart image with technical overlays (trend channel, MACD, RSI);
(b) Grad-CAM++ activation heatmap superimposed on the input chart, highlighting regionswith strong influence on the model's prediction; (c) Isolated heatmap visualization showing spatial saliency distribution

a recovery (sometimes referred to as a "morning star" formation or simply a bullish pullback reversal). The CNN likely picked up on this subtle configuration to indicate an upward turn.

In this example, the model correctly predicted a downtrend (confidence: 0.86), primarily focusing on the most recent sequence of bearish candles toward the right edge of the image. The red-highlighted areas indicate high feature importance as interpreted by the CNN, suggesting that the decision was influenced by the post-peak dip and closing formations, consistent with technical trading heuristics. Earlier portions of the chart contribute less, as reflected by their predominantly blue shading.

The Grad-CAM++ results across multiple samples generally revealed a sensible pattern, i.e., the model emphasizes the last several candles in the chart window, which aligns with the idea that recent price action most indicates the immediate trend. The heatmaps often highlighted recent red candles or a bearish engulfing pattern in predicted downtrends. In cases of uptrends, the focus was on recent green candles or bullish reversal patterns after a dip. This proves that the CNN's internal reasoning is not a mysterious "black box" but corresponds to recognizable visual cues experienced traders use. Moreover, it helps validate that the model is not basing its decisions on spurious parts of the image (such as labels, axes, or random noise) – a potential concern when using images. All heatmaps concentrated on the region where the candlesticks were, and none indicated reliance on non-informative areas.

Together, the filter activation analysis and Grad-CAM++ explanations give us confidence that the CNN is both practical and reasonable in how it derives its predictions. It has learned to parse the chart into meaningful components and focus on the most relevant time series segments for making a trend call. This interpretability is particularly important for deploying such a model in practice, as it allows analysts to double-check the model's rationale and increases trust in automated predictions.

DISCUSSION

It is demonstrated by experiments that transforming time series data into candlestick chart images and applying a CNN is a viable approach to trend classification. The model accurately predicted short-term stock trends (up vs. down) from visual patterns alone. This contributes to the growing evidence that deep learning can extract complex features from time series when provided in a two-dimensional format [8, 10]. In case, combinations of candlestick shapes and sequences that correlate with bullish or bearish outcomes were likely learned to be recognized by the CNN, automating what might be done by eye by a technical analyst, but with greater consistency and speed.

One notable aspect is that the approach required minimal feature engineering – no hand-crafted technical indicators were calculated, and chart patterns were not explicitly labeled in the training data. Instead, the CNN inferred relevant features directly from raw price charts. This aligns with the promise of deep learning to uncover patterns that may be difficult to quantify manually. At the same time, it places the burden on having sufficient training data for the model to learn from. In the study, the amount of historical data was enough for the model to generalize well, as evidenced by the validation and test performance. In scenarios with limited data, one might consider data augmentation or transfer learning (e.g., pre-training on a large set of generated financial charts or related time series images) to boost performance.

Some consistency and discrepancies are observed when comparing the findings of other studies. High accuracy is encouraging and in line with Chen and Tsai's pattern classification results (around 90% for eight patterns) [10], suggesting that visual cues in charts are indeed learnable by CNNs to a high degree of precision. On the other hand, Ding et al.'s ~70% accuracy report for pure image-based models might seem lower [4]. However, they dealt with a more diversified set of assets (stocks, forex, crypto) and aimed to predict a more general notion of "market strength" [14].

In a more focused context (one stock, near-term trend), the patterns might be more internally consistent, allowing higher accuracy. Additionally, differences in window length, image resolution, and class definition can impact results significantly – these hyperparameters require tuning for each application. For example, N widow's days were: if N is too small, the chart may not contain enough information to discern a trend, but if N is too large, the older part of the chart may introduce noise or irrelevant history. Performed Grad-CAM++ analysis indicated the model naturally emphasized the last part of the window, hinting that one could potentially reduce N and maintain performance, an avenue for future optimization.

The interpretability analysis (filter activations and Grad-CAM++) provided reassurance that the CNN's behavior aligns with domain knowledge. This is important because financial decisions often require an explanation. If an artificial intelligence model were to be used by traders or analysts, they would want to know why it forecasts a particular trend. Grad-CAM++ visualizations can provide a rationale – e.g., "the model predicts an uptrend because it sees a particular bullish pattern in the last few days". This explanation can bridge the gap between AI and human decision-making, making integrating the tool in practice easier. It also helps identify when the model might be making an error for the wrong reasons (though evidence was not found in research tests – the focus areas were always logical chart regions).

Despite the positive results, there are several limitations and considerations to discuss. First, the scope of the experiment was a binary classification of short-term trend on a single stock. Market dynamics can be far more complex; extending this approach to multi-class classification (e.g., predicting up, down, no significant change, or predicting different magnitudes of movement) would increase its utility and difficulty. Preliminary exploration suggests that distinguishing a "no change" class is tricky because slight ups/downs might visually resemble flat movements. Another limitation is that the proposed model does not incorporate fundamental data or macroeconomic context, which often drives longer-term trends. It purely looks at price history in chart form. For many engineering applications, similarly, one might need to integrate multiple data streams (for example, temperature and pressure sensor readings together) – one could encode those as multi-channel images (RGB channels or more) to feed a CNN, which is a promising direction supported by literature.

From a methodological perspective, one challenge with image-based time series analysis is ensuring that important quantitative information is not lost in translation. Plotting candlesticks involves decisions like scaling the y-axis (price axis). Inconsistent scaling could trick the CNN – for instance, a slight price fluctuation in a zoomed-in chart might look like a big move. This was addressed by fixing the window length and letting the y-axis scale adapt to each window's range, so the CNN learns pattern shape irrespective of absolute scale. In other applications, one might need to standardize this (maybe using fixed scales or adding reference gridlines to images) to avoid misinterpretation by the model. The advantage, though, is that CNNs are somewhat scale-invariant due to pooling and learned filters; the model likely learned shape patterns that are robust to moderate variations in scale.

Finally, while the study emphasized stock market data as a case study, the approach has broad applicability. Any time series data that can be visualized meaningfully – whether it is an engine's vibration frequency spectrum, an electrocardiogram (ECG) signal plotted over time, or a meteorological time series depicted in a colored map – can potentially be fed into a CNN for classification or anomaly detection. Prior works have shown CNNs distinguishing heartbeat arrhythmias from ECG plots, or identifying machinery faults from spectrograms, echoing the same underlying principle we applied. The key is to tap into an extensive repository of computer vision techniques and architectures using images. This opens opportunities to use pre-trained CNNs (on massive image datasets) as feature extractors for time series images, or to leverage visualization-driven methods for debugging and improving models.

In conclusion, the discussion underscores that image-based deep learning is a powerful tool for time series analysis. However, it should be applied carefully, considering its assumptions and limitations. The success seen here with candlestick charts encourages further exploration, such as combining image-based features with traditional time-series features (a form of model ensemble or feature fusion) to achieve even better results, possibly. Additionally, ensuring interpretability through methods like Grad-CAM++ makes such models more transparent and likely to be adopted in real-world decision-making.

CONCLUSIONS

This paper presented an approach to time series trend classification using deep learning on image representations of the data. A convolutional neural network's strength in visual pattern recognition was leveraged to predict short-term market trends by converting stock price series into candlestick chart images. The CNN model achieved high classification accuracy on out-of-sample data, confirming that significant predictive signals exist in the visual patterns of candlestick charts. We demonstrated that the model's learned features correspond to intuitive chart components (such as candle shapes and arrangements). Using Grad-CAM++, the study provided visual explanations for the model's predictions, enhancing trust in the results.

The implications of these findings extend beyond the stock market example. The methodology can be generalized to other fields where time series data can be visualized – for instance, industrial sensor data, medical signals, or climate patterns – enabling the application of advanced image-based deep learning models in those domains. This cross-pollination of techniques allows researchers and practitioners to utilize CNN architectures, which are well-developed in computer vision, for time-oriented data analysis. Additionally, the built-in interpretability tools from the vision domain (like class activation mappings) can be repurposed to aid understanding of time series models, as shown.

Future work will explore several directions to build on this research. One direction is to incorporate multi-channel images (for example, plotting multiple related time series as separate color channels or panel sub-images) so that the CNN can learn from multiple signals jointly. This could enhance performance in complex scenarios, such as considering price and trading volume charts together for trend prediction. Another perspective is integrating the proposed image-based approach with traditional numerical features, i.e., a hybrid model could take raw price sequences (or technical indicators) and candlestick images as inputs, potentially marrying the strengths of both representations. Moreover, evaluating the approach on different types of assets (commodities, cryptocurrencies) or even non-financial time series will help assess its generality. Lastly, from an interpretability standpoint, we plan to investigate other explanation techniques (such as SHAP or LIME adapted for images) to cross-verify what the CNN learns, aiming to solidify further confidence in deploying such models in decision-critical applications.

In summary, converting time series data into images for deep learning is a promising strategy that bridges time-series analysis and computer vision. The study confirms that a CNN can effectively classify trends from candlestick chart images and that its decision process can be transparent. This contributes to the toolkit of advanced signal processing and prognostics in engineering and finance, opening up new possibilities for accurate and explainable predictive analytics.

REFERENCES

1. Penar P, Szeremeta M, Gola A. A hardware-software compatibility in robotic cyber-physical systems – an application based approach. Adv Sci Technol Res J. 2025;19(6):330–41.
2. Paszkowski W, Gola A, Świć A. Acoustic-based drone detection using neural networks – a comprehensive analysis. Adv Sci Technol Res J [Internet]. 1 luty 2024;18(1):36–47. http://www.astrj.com/Acoustic-Based-Drone-Detection-Using-Neural-Networks-A-Comprehensive-Analysis,175863,0,2.html

3. Chen JH, Tsai YC. Encoding candlesticks as images for pattern classification using convolutional neural networks. Financ Innov. 2020;6(1).
4. Ding Y. Enhancing stock price prediction method based on CNN-LSTM hybrid model. Highlights Business, Econ Manag. 2023;21:774–81.
5. Capelin M, Martinez GAS, Xing Y, Siqueira AF, Qian WL. Analysis of wire rolling processes using convolutional neural networks. Adv Sci Technol Res J. 2024;18(2):103–14.
6. Saad A, Sheikh UU, Moslim MS. Developing convolutional neural network for recognition of bone fractures in X-ray images. Adv Sci Technol Res J. 2024;18(4):228–37.
7. Cioch M, Kulisz M, Kański Ł. Implementing AI collaborative robots in manufacturing – modeling enterprise challenges in industry 5.0 with fuzzy logic. Adv Sci Technol Res J [Internet]. 1 listopad 2024;18(7):229–38. http://www.astrj.com/Implementing-AI-Collaborative-Robots-in-Manufacturing-Modeling-Enterprise-Challenges,192833,0,2.html
8. Casolaro A, Capone V, Iannuzzo G, Camastra F. Deep learning for time series forecasting: advances and open problems. Inf. 2023;14(11).
9. Mienye E, Jere N, Obaido G, Mienye ID, Aruleba K. Deep learning in finance: A survey of applications and techniques. AI. 2024;5(4):2066–91.
10. Ganguly P, Mukherjee I, Garine R. Visualizing Machine Learning Models for Enhanced Financial Decision-Making and Risk Management. 2024 3rd Int Conf Trends Electr Electron Comput Eng TEECCON 2024. 2024;210–5.
11. Aryal S, Nadarajah D, Rupasinghe PL, Jayawardena C, Kasthurirathna D. Comparative analysis of deep learning models for multi-step prediction of financial time series. J Comput Sci. 2020;16(10):1401–16.
12. Sood S, Zeng Z, Cohen N, Balch T, Veloso M. Visual Time Series Forecasting: An Image-driven Approach. ICAIF 2021 - 2nd ACM Int Conf AI Financ. 2021;
13. Sezer OB, Ozbayoglu AM. Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. Appl Soft Comput [Internet]. wrzesień 2018;70:525–38. Dostępne na: https://linkinghub.elsevier.com/retrieve/pii/S1568494618302151
14. Shi Z, Hu Y, Mo G, Wu J. Attention-based CNN-LSTM and XGBoost hybrid model for stock prediction. 2022; Dostępne na: http://arxiv.org/abs/2204.02623
15. Ajit A, Acharya K, Samanta A. A Review of Convolutional Neural Networks. Int Conf Emerg Trends Inf Technol Eng ic-ETITE 2020. 2020;
16. Janiesch C, Zschech P, Heinrich K. Machine learning and deep learning. Electron Mark [Internet]. 8 wrzesień 2021;31(3):685–95. Dostępne na: https://link.springer.com/10.1007/s12525-021-00475-2
17. Ho TT, Huang Y. Stock price movement prediction using sentiment analysis and candlestick chart representation. Sensors. 2021;21(23).
18. Wang J, Li X, Jia H, Peng T, Tan J. Predicting stock market volatility from candlestick charts: A multiple attention mechanism graph neural network approach. Math Probl Eng. 2022;2022.
19. Hung CC, Chen YJ. DPP: Deep predictor for price movement from candlestick charts. PLoS One. 2021;16(6 June 2021).
20. Wang L, Müller R, Zhu F, Yang X. Collective mindfulness: The key to organizational resilience in megaprojects. Proj Manag J. 2021;52(6):592–606.
21. Kamilaris A, Prenafeta-Boldú FX. Deep learning in agriculture: A survey. Comput Electron Agric. 2018;147:70–90.
22. Sarker IH. Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. SN Comput Sci. 2021;2(6).
23. Arrieta, A., Díaz-Rodríguez, N., Ser, J., Bennetot, A., Tabik, S., Barbado, A. …, Herrera. Decoding the black box through a comparative study on clustering features in convolutional neural networks. Acad J Comput Inf Sci. 2023;6(12).
24. Indrakumari R, Kumar TG, Murugan D, P.C. S. Deep learning in medical image analysis. Deep Learn Med Image Anal. 2024.
25. Zhu Y, Luo S, Huang D, Zheng W, Su F, Hou B. DRCNN: decomposing residual convolutional neural networks for time series forecasting. Sci Rep. 2023;13(1).