Received 24 December 2023, accepted 22 January 2024, date of publication 25 January 2024, date of current version 20 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3358452

RESEARCH ARTICLE

Deep Transformer-Based Asset Price and Direction Prediction

ABDUL HALUK BATUR GEZICI, (Member, IEEE), AND EMRE SEFER, (Member, IEEE)
Computer Science Department, Özyeğin University, 34794 İstanbul, Turkey

Corresponding author: Emre Sefer (emre.sefer@ozyegin.edu.tr)

ABSTRACT The field of algorithmic trading, driven by deep learning methodologies, has garnered substantial attention in recent times. Within this domain, transformers, convolutional neural networks, and patch embedding-based techniques have emerged as popular choices within the computer vision community. Here, inspired by the latest cutting-edge computer vision methodologies and the existing work showing the capability of image-like conversion for time-series datasets, we apply more advanced transformer-based and patch-based approaches for predicting asset prices and directional price movements. The employed transformer models include Vision Transformer (ViT), Data Efficient Image Transformers (DeiT), and Swin. We use ConvMixer for a patch embedding-based convolutional neural network architecture without a transformer. Our tested transformer-based and patch-based methodologies aim to predict asset prices and directional movements using historical price data by leveraging the inherent image-like properties within the historical time-series dataset. Before the implementation of attention-based architectures, the historical time series price dataset is transformed into two-dimensional images. This transformation is facilitated through the incorporation of various common technical financial indicators, each contributing to the data for a fixed number of consecutive days. Consequently, a diverse set of two-dimensional images is constructed, reflecting various dimensions of the dataset. Subsequently, the original images depicting market valleys and peaks are annotated with labels such as Hold, Buy, or Sell. According to the experiments, trained attention-based models consistently outperform the baseline convolutional architectures, particularly when applied to a subset of frequently traded Exchange-Traded Funds (ETFs). This better performance of attention-based architectures, especially ViT, is evident in terms of both accuracy and other financial evaluation metrics, particularly during extended testing and holding periods. These findings underscore the potential of transformer-based approaches to enhance predictive capabilities in asset price and directional forecasting. Our code and processed datasets are available at https://github.com/seferlab/price_transformer.

INDEX TERMS Asset price prediction, deep learning, attention, vision transformers, convolutional neural network.

I. INTRODUCTION

The prediction of asset prices, including stocks, through artificial intelligence systems, has been a subject of study for nearly three decades. In contemporary financial markets, the scope of tradeable instruments extends beyond stocks to include options [1], Exchange-Traded Funds (ETFs) [2], cryptocurrencies [3], commodities [4], NFTs [5] etc. Correspondingly, the role of artificial intelligence-based trading systems has grown in significance and functionality across diverse global markets [4].

Deep learning methods have recently exhibited superior performance in various classification and prediction tasks compared to more traditional machine learning models like Support Vector Machines (SVMs). Notably, deep learning excels in image processing tasks such as segmentation and classification, marking a significant departure from conventional methodologies [6]. A parallel trend has emerged in financial prediction and classification tasks, encompassing asset price and directional prediction. While traditional

The associate editor coordinating the review of this manuscript and approving it for publication was Wei-Yen Hsu.

models like Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN) have been employed, the application of these techniques in financial prediction lags behind their prevalence in other artificial intelligence areas such as computer vision [7], [8], [9] and Natural Language Processing [10], [11], [12]. Among these techniques, CNNs have demonstrated notable performance in financial asset price prediction, even though their primary application has been in computer vision tasks like image classification [6].

Transformers [13], [14], [15], introduced primarily for sequence analysis in natural language processing, have recently surpassed CNNs in multiple vision tasks. Vision Transformer (ViT) [16], for example, has shown promising results and surpassed state-of-the-art convolutional neural networks while requiring significantly less training cost. This success has led to the development of various advanced vision transformers for computer vision tasks. Transformers leverage attention and self-attention mechanisms, facilitating the establishment of connections within sequences or images. Self-attention, a mechanism that relates different parts of an input while computing a representation of the same input, has proven particularly effective in addressing longer-range dependencies.

In this study, we evaluate three commonly used vision transformer-based approaches and a recently proposed patch embedding-based approach within our algorithmic trading framework, aiming to predict both asset prices and directional movements. Among these three transformer techniques, ViT, a vision transformer-based algorithm relies on attention and self-attention mechanisms. ViT can be seen as the first adaptation of the transformer which is proposed for natural language processing tasks to the vision tasks. ViT combines those attention mechanisms with patch-based embeddings. Another transformer, DeiT (Data-Efficient Image Transformers) [17] is similar to ViT except for the distillation mechanism which incorporates the soft and hard predictions from both the student and teacher sides respectively. More recently proposed Swin [18] is the last transformer we analyze. Swin transformer is a hierarchical Vision Transformer that uses shifted windows to efficiently calculate the attention. Lastly, we also apply a patch-embedding convolutional neural network-based approach ConvMixer [19] which combines multiple convolutional architectures with patch embeddings. ConvMixer architecture is similar to ViT, except it completely removes the transformer blocks. So, its performance will be important in analyzing the importance of transformer blocks.

In our financial prediction problem, patch-embedding convolutional neural networks and transformers represent two distinct methodologies employed in transforming one-dimensional financial time series datasets into two-dimensional representations akin to images. We transform our time-series datasets into an image by using 65 different technical indicators, each with various parameter combinations over a defined period. Some common examples of these technical indicators are MACD [20], RSI [21], Bollinger Bands [22], Fibonacci Retractment, etc. Once turned into a matrix, the resulting 2D representation organizes rows to cluster indicators with similar patterns, ensuring smooth transitions along the y-axis, and capturing consecutive patterns for deep learning techniques in asset price and direction predictions.

In our experiments, we generate images of varying dimensions through technical indicators, feeding them into Vision Transformer and patch embedding-based convolutional neural networks, respectively. The application of algorithmic trading approaches involving the transformation of time series datasets into 2D representations is relatively limited [23]. CNN-TA, proposed in [23], utilizes convolutional neural networks for understanding, yet as our results indicate, such networks do not surpass more recent architectures like transformers or patch embedding-based CNNs. To the best of our knowledge, the utilization of algorithmic trading by transforming time series datasets into 2D representations and subsequently processing them through transformers or patch embedding-based CNNs, analogous to 2D image classification tasks, is unprecedented, even in other financial prediction domains.

Our detailed experiments demonstrate that transformer-based approaches generally outperform the well-known baseline methods and similar methods utilizing simpler CNNs without the transformer architecture across extended periods. Across both longer and shorter testing periods, transformer-based architectures are still more accurate than all baselines and the competing CNN-based method CNN-TA. Moreover, transformer-based models outperform a simple buy-and-hold strategy [24], common technical indicator-based models [25], Multilayer Perception (MLP) [26], and a common deep learning time series forecasting model LSTM [27]. While the performance of transformer-based approaches is promising, further enhancement is achievable through more detailed hyperparameter optimization and fine-tuning.

A. RELATED WORK

Finance can be seen as one of the most focused machine learning application areas during the last 30 years. Until now, many research papers have been published. Traditional machine learning methodologies have been extensively employed for stock market forecasting, with studies focusing on applying time-series prediction techniques directly to financial datasets. Some have utilized fundamental or technical analysis techniques for accurate forecasting, as demonstrated in surveys such as [4]. Previous works, like [28], [29], [30], [31], [32], have applied Artificial Neural Networks (ANNs) to predict stock index values or forecast stock prices, integrating technical analysis indicators. [33] has compared the existing common techniques for text mining which are also adapted to stock market prediction problems such as Rule-based systems, Genetic algorithms (GAs), and neural networks. Among other related research, [34] used kernel techniques to forecast stock market changes. In general, machine learning techniques performance is limited since they cannot infer deeper attributes in the financial dataset which is a key factor in closing the gap between machine and human trading performance.

In recent years, the surge in computational capacity has led to the emergence of newer deep learning-based approaches. Deep learning, as a specialized case of ANN with multiple layers, has demonstrated enhanced performance compared to shallower networks [35]. Various deep learning models, including Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNNs), and Transformers, have found applications across diverse domains. CNNs, for instance, are widely used in image classification [36], [37] and video processing, as well as in natural language processing tasks such as sentence categorization [38]. LSTMs and RNNs are predominantly employed for sequential data analysis in speech processing, natural language processing, and time-series-related tasks. Transformers, initially applied to Natural Language Processing (NLP) tasks [39], have more recently demonstrated success in computer vision tasks [16], [40].

While deep CNNs and transformer-based architectures have become prevalent in the last decade, their application in financial tasks remains limited. Existing studies, such as [23], [27], [41], [42], [43], [44], [45], have explored deep learning approaches for forecasting stock markets, leveraging events data, financial time-series datasets, and news integration. Relevant studies in [23], [45] incorporate technical analysis indicators into prediction models, employing feedforward neural networks and CNNs with two-dimensional matrix characterizations of technical analysis datasets. Notably, deep learning demonstrates accurate learning and generalization across buy and sell time points over extended testing horizons in these studies.

Other studies enhance stock market prediction performance by incorporating extra strategies that can be utilized to assist in market movement prediction. Such methods are generally called hybrid methods, which are deep learning techniques integrated with extra algorithms. Those hybrid techniques have shown great promise in terms of their predictive abilities [46], [47]. For instance, the method proposed in [48] uses adversarial training to enhance neural networks generalizability by preventing overfitting while predicting stock price rise or fall. Similarly, [49] came up with a novel deep generative architecture that combines price and textual datasets to better model the stock prediction problems complexity. Different than traditional discriminative topic models, the model proposed in [49] handles the stochasticity better by introducing recursive variables where they used Variational Autoencoders (VAEs) for detailed reasoning.

On the other hand, transformer-based approaches for financial market prediction are notably limited. Studies such as [50], [51], [52] propose novel transformer architectures or methods to predict stock movements, incorporating feature engineering and multi-scale Gaussian priors. These studies also integrate textual attributes from the Twitter dataset into the prediction. The existing studies frequently utilize CNNs for image analysis and classification tasks, while deep learning methods, particularly LSTM and RNN, are commonly employed for financial time-series prediction. However, the integration of technical analysis datasets with deep learning techniques, especially the combination of CNNs with two-dimensional matrix characterizations of these datasets, is relatively rare in algorithmic trading studies [6], [23], [32], [53], [54]. These studies typically convert the financial time-series prediction problem into an image classification task by generating two-dimensional images from price indicators. Recent work [54] applies a single type of vision transformer to such image-like data for the first time.

II. DATA PREPROCESSING

A. DATASET PREPARATION

Financial data is generally analyzed by either technical analysis (TA), fundamental analysis, or quantitative analysis [4]. In fundamental analysis, investors focus on predicting an asset's price by examining the corresponding company's reported metrics such as long-term debt, short-term debt, cash flow, return on equity, return on asset, earning per share, etc [55]. On the contrary, technical analysis is based on analyzing the temporal price and volume characteristics of a company via predefined mathematical models. A huge number of technical analysis indicators have been proposed to help with financial asset price prediction over time.

Here, we have collected Open, High, Low, Close, Volume (OHLCV) time-series datasets for nine of the frequently-traded Exchange-Traded Funds (ETFs) by using the yfinance library [56] over 20 years period from 1/1/2002 to 1/1/2022 for training and testing our approaches. This 20-year duration consists of both bull and major bear markets such as the 2008 financial crisis and 2020 coronavirus crises. These 9 ETFs are summarized in Table 1. We applied 65 commonly-used technical indicators [57] which focus on different time horizons on distinct categories such as volatility, momentum, volume, overlap studies, price transformation, and statistics. We use TA-Lib (Technical Analysis Library) in Python to calculate these indicator values over time. All used indicators are summarized in Table 2.

B. GENERATING LABELS

We labeled the time-series dataset for our supervised learning problem, after preprocessing and partitioning the dataset. In this case, we assigned one of *Buy*, *Hold*, or *Sell* classes according to a predetermined threshold value. Data for periods above the threshold are tagged as *Buy*, below the threshold are tagged as *Sell*, and the remaining periods are tagged as *Hold*. The threshold value has a significant impact on the label distribution, as the generated dataset frequency

TABLE 1. ETFs used in our analysis and their attributes.

Symbol	Description	Inception Date
XLF	Financial Select Sector SPDR ETF	12/16/1998
XLU	Utilities Select Sector SPDR ETF	12/16/1998
QQQ	PowerShares QQQ ETF	10/03/1999
SPY	SPDR S&P 500 ETF	1/22/1993
XLP	Consumer Staples Select Sector SPDR ETF	12/16/1998
EWZ	iShares MSCI Brazil Capped ETF	7/10/2000
EWH	iShares MSCI Hong Kong ETF	3/12/1996
XLY	Consumer Discret Sel Sect SPDR ETF	12/16/1998
XLE	Energy Select Sector SPDR ETF	12/16/1998

TABLE 2. Technical analysis indicators used in our study.

Category	Indicators
Statistics	BETA, CORREL, LINEARREG, LINEARREG_ANGLE, LINEARREG_INTERCEPT, LINEARREG_SLOPE, STD, TSF, VAR
Price Transformation	AVGPRICE, MEDPRICE, TYPPRICE, WCLPRICE
Volatility	TRANGE
Volume	AD, ADOSC, OBV
Momentum	ADX, ADXR, APO, AROONUP, AROONDOWN, AROONOSC, BOP, CCI, CMO, DX, FASTD, FASTDRSI, FASTK, FASTKRSI, MACD, MACDEXT, MACDFIX, MFI, MINUS_DI, MINUS_DM, MOM, PLUS_DI, PLUS_DM, PPO, ROC, ROCP, ROCR, ROCR100, RSI, SLOWD, SLOWK, TRIX, ULTOSC, WILLR
Overlap Studies	BBANDSL, BBANDSM, BBANDSU, DEMA, EMA, HT_TRENDLINE, KAMA, MA, MIDPOINT, MIDPRICE, SMA, TEMA, TRIMA, WMA

is dependent on the threshold. In our analysis, we focused on 2 threshold values 0.0038 and 0.01, which correspond to balanced and imbalanced datasets respectively. Evaluating the performance on both balanced and imbalanced datasets is crucial for quality and robustness. Intuitively, 0.01 threshold means buying the asset for a price increase greater than 1% and selling the asset for a price decrease greater than 1% in a day which is a reasonable assumption even though it results in an imbalanced dataset. On the other hand, 0.0038 threshold generates a dataset of almost equal distribution among these three classes. We report our results for 0.01 threshold case unless otherwise noted.

C. MAPPING TECHNICAL INDICATORS TO IMAGES

To apply vision transformer-based and patch-based approaches, we transform our one-dimensional time-series dataset into 2D images. While converting time-series data to image dataset, we calculate MACD, CMO, PPO, WILLR, EMA, WMA, SMA, ROC, CCI, TEMA, RSI, and 54 more technical indicator signals across different horizons. These indicator ranges are between one week to three weeks. At a higher level, those technical analysis indicators might also be considered as filters applied to financial time series. For instance, the Fast Fourier Transform will also be an indicator, but we do not consider it in our application since it is not finance-specific. Those indicators are frequently utilized and ideal for medium-horizon trading, which is the focus of this research. If one is interested in high-frequency trading or longer-term trading, these technical analysis indicators should be adjusted accordingly.

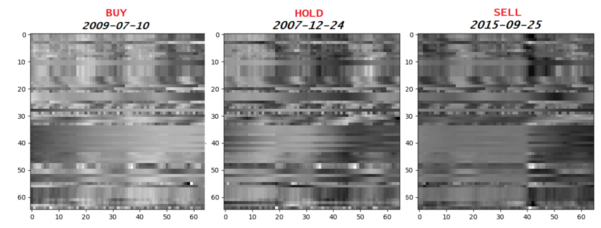


FIGURE 1. 65 × 65 labeled generated images.

After applying these technical indicators to a one-dimensional time-series dataset of 65 historical days, we generate a 65 × 65 image of each training sample. In this image matrix, each column represents a distinct day in these 65 days whereas each row corresponds to a different technical indicator. Even though columns are already ordered temporally, there is no such apparent ordering for technical indicators. The indicator ordering is a key factor in our results since we will obtain distinct images for distinct orderings. In our case, we sort the resulting set of indicators by their categories as shown in Table 2. Moreover, once we generate the images for a given ordering, we apply standardization to make the dataset more robust and consistent [58]. For instance, we plot examples of 65 × 65 pixel images generated during our preprocessing in Figure 1.

III. METHODS

We test the transformer-based and patch embedding-based algorithmic trading approaches performance and compare it with baselines including convolutional neural networks.

A. VISION TRANSFORMER (VIT)

Transformers have frequently resulted in state-of-the-art outcomes for many natural language processing tasks. However, they cannot be directly applied to image-like data. A natural adaptation is to apply attention together with convolutional networks, or to modify only certain portions of convolutional networks without modifying the overall architecture. Direct self-attention application on image-like data will result in a very high complexity since each pixel should attend to every other pixel. Different solutions have been proposed to decrease this complexity [59], [60] which apply self-attention to local neighborhoods in a scalable way.

Following such adaptations, researchers came up with ViT (Vision Transformer) [16] which can be seen as a pure transformer for image-like data. ViT applies self-attention directly to sequenced image patches, where it completely removes the dependence on CNNs. Since its proposal, ViT has been integrated into many computer vision tasks including image segmentation, and image classification. Its performance has been state-of-the-art in those domains.

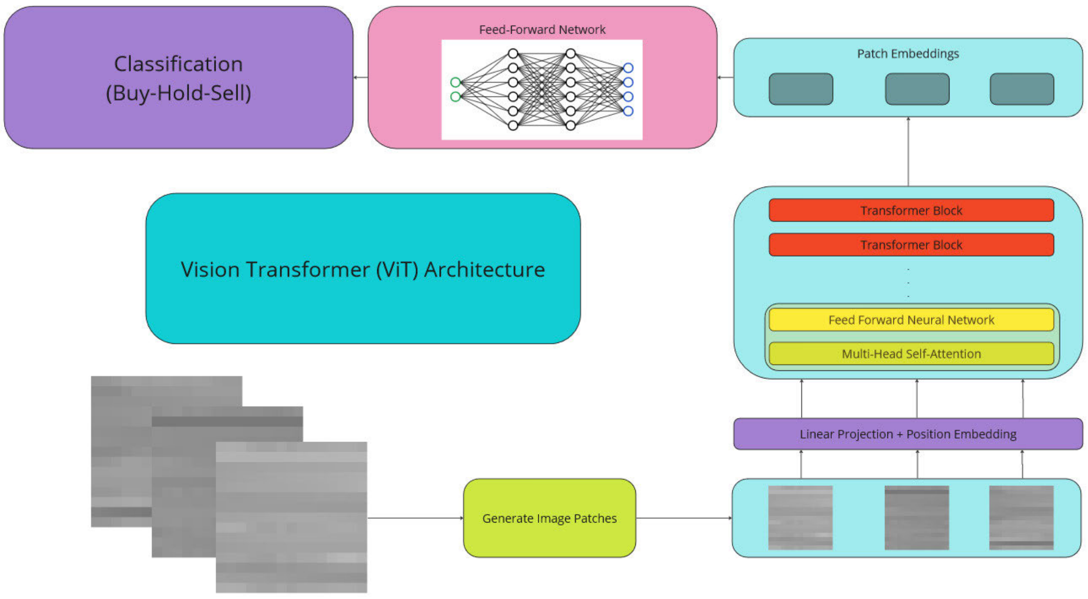

Classification
(Buy-Hold-Sell)
Feed-Forward Network
Patch Embeddings
Vision Transformer (ViT) Architecture
Transformer Block
Transformer Block
Feed Forward Neural Network
Multi-Head Self-Attention
Linear Projection + Position Embedding
Generate Image Patches

FIGURE 2. **Overview of ViT architecture in our problem.**

In ViT, we have a patch and position-based embedding layer that first partitions the image into sub-images. Once images are partitioned, it is flattened and a transformer encoder which has multi-layer attention is applied as in Figure 2.

In more detail, given an image of $W \times H$ and a patch size B, ViT first transforms the image into $\frac{W}{B} \times \frac{H}{B}$ size components. Once these $\frac{W}{B} \times \frac{H}{B}$ are flattened, a multi-layer encoder is applied. In each encoder layers, as defined in [16], the forward equations are follows:

$$z_0 = [x_{class}; x_p^1 E; x_p^2 E; \ldots; x_p^N E] + E_{pos}, \tag{1}$$

$$E \in R^{(P^2 C)\times D}, E_{pos} \in R^{(N+1)\times D} \tag{2}$$

$$z_l' = MSA(LN(z-1)) + z_{l-1}, \qquad l = 1 \ldots L \tag{3}$$

$$z_l = MLP(LN(z)) + z_l', \qquad l = 1 \ldots L \tag{4}$$

$$y = LN(z_L 0) \tag{5}$$

where MSA represents multi-headed self-attention, MLP represents multilayer perception, and LN represents layer normalization in the Transformer Encoder part. ViT adds positional embeddings to patch embeddings to keep positional knowledge.

B. CONVMIXER

While CNNs are pyramid architectures with decreasing resolution by using convolution, ConvMixer [19] is an isotropic architecture that is integrated with patch representation and convolution. As summarized in Figure 3, ConvMixer architecture is very simple and preserves locality using tensor patch embeddings. After the patch embedding step, d copies of a simple Fully Convolutional block are applied where each block includes large-kernel depthwise convolution and pointwise convolution. Lastly, global average pooling is applied which is followed by a simple fully connected layer.

In our case, ConvMixer [19] has been applied to discuss the effectiveness of transformers for asset price prediction problems. Even though ConvMixer has no attention mechanism, it trains CNN units integrated with Gaussian Linear Unit (GeLU). More formally, ConvMixer implements patch embeddings with a patch size p and embedding dimension h as a convolution with c_{in} input channels, h output channels, kernel size p, and stride p:

$$z_0 = \text{BN}(\sigma\{\texttt{Conv}_{c_{\text{in}}\to h}(X, \texttt{stride}=p, \texttt{kernel\_size}=p)\}) \tag{6}$$

where BN represents batch normalization. Patch embedding summarizes a $p \times p$ patch into an embedded vector of dimensions e. The authors implement this by a single convolution with kernel size p, stride p, and h output channels, followed by a non-linearity. This surprising trick will convert the $n \times n$ image into features of shape $h \times \frac{n}{p} \times \frac{n}{p}$. The ConvMixer block is composed of depthwise convolution (grouped convolution with groups equal to the number of channels, h) followed by pointwise convolution (kernel size 1×1). Each of these convolutions is followed by an activation and post-activation batch normalization as in:

$$z_l' = \text{BN}\left(\sigma\{\texttt{ConvDepthwise}(z_{l-1})\}\right) + z_{l-1} \tag{7}$$

$$z_{l+1} = \text{BN}\left(\sigma\{\texttt{ConvPointwise}(z_l')\}\right) \tag{8}$$

C. DATA-EFFICIENT IMAGE TRANSFORMERS (DEIT)

DeiT [17] is another type of recent vision transformer that is based on Knowledge Distillation (KD). Knowledge distillation [61] represents a training paradigm where a student model uses soft labels suggested by a strong teacher model. In this case, soft labels are the output vector of the teacher's softmax function instead of maximum scores which

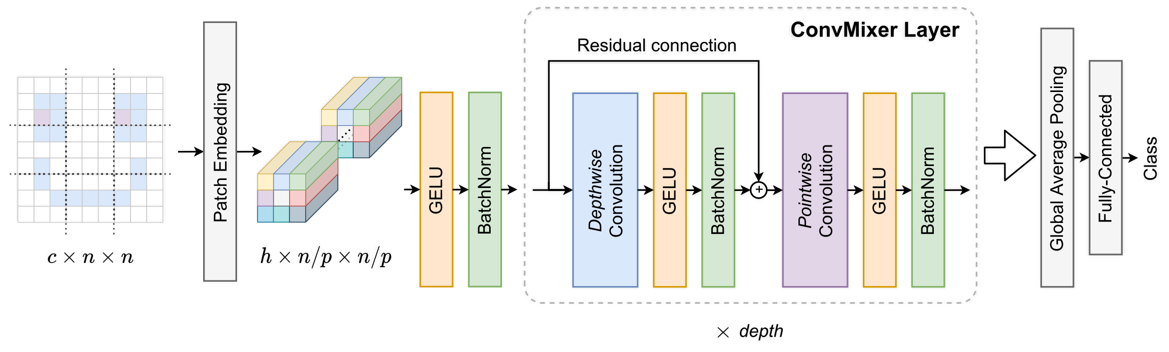


FIGURE 3. Overview of ConvMixer architecture in our problem.

can be considered as a hard label. This type of training will enhance the student model's performance since it will transfer the knowledge from a larger model to a smaller one.

DeiT studies the distillation of a transformer student by either a convolutional neural network or a transformer teacher. DeiT came up with a new distillation process that is transformer-specific. DeiT assumes to have a powerful classifier as a teacher model which can be a convolutional neural network or a mixture of classifiers. It mainly tries to learn a transformer by using the knowledge from the teacher. DeiT incorporates a new distillation token that interacts with the class token and other patch tokens via the following self-attention layers. The architecture uses the distillation token similar to the class token. One main difference between them is that the distillation token focuses on reproducing the hard label inferred by the teacher, whereas the class token's objective is to predict the true label.

DeiT includes both soft and hard distillation. Soft distillation focuses on minimizing the Kullback-Leibler (KL) divergence between the softmax of the teacher model and the softmax of the student model. More formally, let Z_t and Z_s represent the logits of the teacher and student models respectively. τ represents distillation temperature, λ balances the KL divergence loss (KL) and the cross-entropy loss ($\mathcal{L}_{CE}$) on ground truth labels y, and ψ defines the softmax function. In this case, the soft distillation objective can be defined as:

$$\mathcal{L}_{\text{global}} = (1-\lambda)\mathcal{L}_{\text{CE}}(\psi(Z_s), y) + \lambda\tau^2 \text{KL}(\psi(Z_s/\tau), \psi(Z_t/\tau)). \tag{9}$$

On the other hand, in hard distillation, the teacher's hard decision is treated as a true label. Assuming $y_t = \text{argmax}_c Z_t(c)$ be the hard decision of the teacher, the hard distillation objective can be defined as:

$$\mathcal{L}_{\text{global}}^{\text{hardDistill}} = \frac{1}{2}\mathcal{L}_{\text{CE}}(\psi(Z_s), y) + \frac{1}{2}\mathcal{L}_{\text{CE}}(\psi(Z_s), y_t). \tag{10}$$

Overall, a novel distillation token is added to the initial embeddings which consist of the class token and patches. The distillation token is similar to the class token in the way it interacts with other embeddings via self-attention and output is generated by the neural network following the last layer. While testing, both distillation and class embeddings generated by the transformer will be used by the following linear classifiers.

D. SWIN TRANSFORMER

Swin transformer [18] is one of the most recent vision transformers which can be used as a general-purpose architecture for almost all vision problems. Even though transformers have been mainly proposed across language tasks, their adaptation to the vision domain requires two main changes: They should be adapted to large changes in visual entity scale, and image pixel resolution is higher than the words in a document. Swin transformer addresses these challenges by proposing a Hierarchical Transformer whose representation is obtained via Shifted windows. In this case, shifted window-based modeling is quite efficient since it restricts self-attention computation to only local windows without an overlap.

More formally, Swin Transformer architecture is summarized in Figure 4. Image-like input is split into non-intersecting patches similar to ViT. It treats each patch as a "token" and each patch attribute is composed of raw values concatenation. Then, the Swin transformer applies a linear embedding layer to this feature for arbitrary dimension projection, denoted as C.

Swin applies many Transformer blocks with modified self-attention computation on patch tokens. The Transformer blocks keep the number of tokens as ($\frac{H}{4} \times \frac{W}{4}$). Overall, Swin transformer replaces the standard multi-head self-attention (MSA) module with a shifted windows-based module. As shown in Figure 4(b), a Swin Transformer block includes a shifted window-based MSA module which is followed by a 2-layer Multilayer Perceptron (MLP) with GELU non-linearity in between. Swin Transformer block applies a layer normalization (LN) before each MSA module and MLP, where a residual connection is applied after each module.

Quadratic complexity is the main challenge for both standard Transformer architecture [13] and its vision-based version [16] since both approaches compute the relationships between all token pairs. To efficiently model self-attention

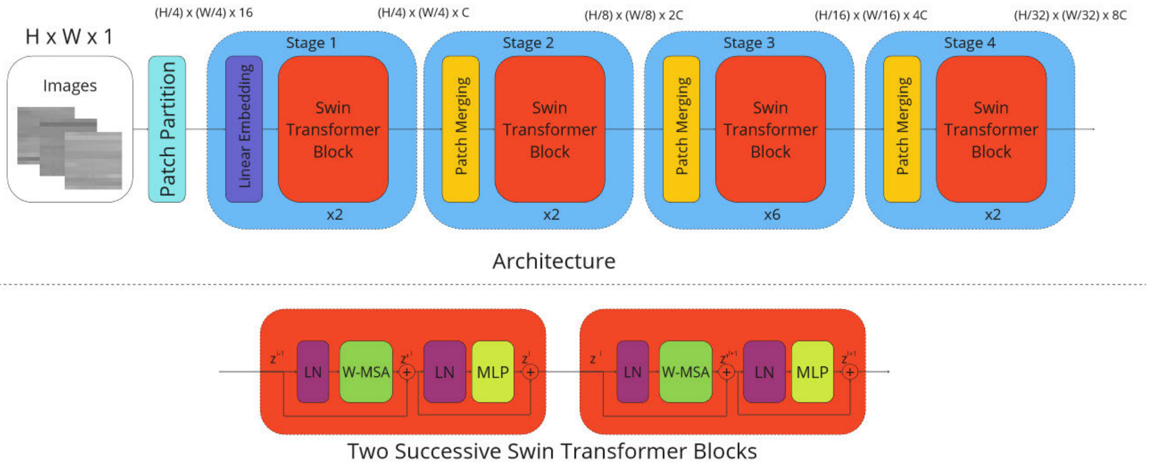


FIGURE 4. (a) Overview of Swin Transformer architecture in our problem. (b) Consecutive Swin transformer blocks where W-MSA and SW-MSA represent multi-head self-attention modules with regular and shifted window configurations, respectively.

in image-like data, Swin transformer calculates self-attention within local windows where it arranges windows for non-overlapping partition of the image-like data. Assuming each window contains $M \times M$ patches, Swin uses the shifted window partitioning technique to efficiently compute non-intersecting windows. In this case, we alternate between different 2 partition configurations in successive Swin Transformer blocks.

The initial block utilizes an orderly window partitioning approach where it starts from the left-upper pixel, and the 8×8 feature map is evenly split into 2×2 windows of size 4×4 ($M = 4$). However, the latter block uses a configuration that is shifted from that of the preceding layer, by displacing the windows by $(\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor)$ pixels from the regularly partitioned windows. With the shifted window partitioning approach, consecutive Swin Transformer blocks are computed as:

$$\hat{z}^l = \text{W-MSA}\left(\text{LN}\left(z^{l-1}\right)\right) + z^{l-1} \tag{11}$$

$$z^l = \text{MLP}\left(\text{LN}\left(\hat{z}^l\right)\right) + \hat{z}^l \tag{12}$$

$$\hat{z}^{l+1} = \text{SW-MSA}\left(\text{LN}\left(z^l\right)\right) + z^l \tag{13}$$

$$z^{l+1} = \text{MLP}\left(\text{LN}\left(\hat{z}^{l+1}\right)\right) + \hat{z}^{l+1} \tag{14}$$

where $\hat{z}^l$ and z^l represent the output features of the (S)W-MSA and the MLP modules for block l, respectively. Additionally, W-MSA and SW-MSA represent window-based multi-head self-attention utilizing regular and shifted window partitioning configurations, respectively.

IV. EXPERIMENTS

A. DATA PREPARATION

Since we are applying an evaluation on a time-series dataset, we chose the train and test datasets in continuous intervals. ETF prices between 1/1/2002 and 1/1/2022 are used in our analysis. We apply a sliding window-based with a retraining approach, where a consecutive 5-year interval is selected as the training period and the following year is selected as the testing period. Following such partition, we shift train and test dataset periods one more year forward and repeat the training process above. As a result, each year between 2007 and 2021 is tested once by such repeated retraining technique. Once the images are generated and standardization is applied, we obtain 45090 images, or 5010 images for each ETF in total. We combine all distinct ETF images into a single dataset since a joint model is trained for all ETFs.

B. BASELINE TECHNIQUES

We focused on 6 baselines while comparing the performance of distinct transformers: Buy & Hold, RSI (Relative Strength Index) (14 days, 70–30), SMA (Simple Moving Average) (50 days), LSTM [27], MLP regression [31], and Enhanced CNN-TA. We have implemented each of these techniques and analyzed associated financial metrics. Among them, Buy & Hold Strategy (BaH) baseline simply longs the assets at the beginning of the test period and it unrolls the position by selling it once the test period finishes. In the RSI model, we trade by only using the RSI indicator. In this case, we calculate RSI for each testing day and we buy if RSI is less than 30. On the other hand, if RSI is greater than 70, we trigger a sell signal. Similarly, in the SMA model, we calculate a 50-day SMA for each testing day. We generate a buy signal if the associated test data is greater than a 50-day SMA, and a sell signal if the test data is less than 50 days SMA. LSTM and MLP regression techniques are utilized as baselines in financial time series data analysis. Our LSTM baseline consists of 25 neurons (1 neuron in the input layer, 25 neurons in the latent layer, 1 neuron in the output layer). In this case, dropout is selected as 0.5 and we run LSTM for 1000 epochs. On the other hand, our MLP baseline consists of four layers which have 1, 10, 5, and 1 neurons consecutively. For MLP, dropout is again selected as 0.5 and we run MLP for 200 epochs.

Lastly, we have enhanced CNN-TA [23] and used it as our last baseline. CNN-TA architecture consists of deep convolutional neural networks, making predictions on two-dimensional data. Once it gets 2D input, it has 2 convolutional layers where each applies 3×3 kernel. After these convolution layers, max pooling is applied. It has 2 dropouts with 0, 25 and 0.5 probabilities. Lastly, the output of these convolutional steps is connected to a fully connected layer which generates the output. Even though CNN-TA is also a neural network-based baseline, its fully convolutional structure does not incorporate an attention mechanism or patch embeddings. It also uses only 15 technical indicators. To make a fair comparison between CNN-TA and vision transformers, we increase the number of technical indicators from 15 to 65 since all vision transformers use 65 indicators. In this case, this enhanced approach is called *Enhanced CNN-TA* or CNN-TA++ in short and we use 65×65 images instead of 15×15 image as input.

C. PERFORMANCE EVALUATION

We assess and compare the performance of multiple vision transformer-based approaches, and the proposed 6 baselines via traditional machine learning metrics and financial metrics. In terms of traditional metrics, we use Accuracy, Recall, Precision, and F1 score for classification performance evaluation. These metrics are defined below:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{16}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{17}$$

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{18}$$

where TP, TN, FP, and FN represent true positive, true negative, false positive, and false negative sample counts, respectively. Let $TPR = \frac{TP}{TP+FN}$ be the true positive rate and $FPR = \frac{FP}{FP+TN}$ be the false positive rate.

In addition to the traditional machine learning methods, we also evaluate the financial performance of the proposed transactions. In this case, we simulate the daily trades by utilizing the model predictions over the test dataset. We can either buy, sell, or hold an asset. When we predict the label as "Buy", we buy the asset by using all our available capital if it has not been already bought. When we predict the label as "Sell", we sell the asset completely if it has been previously bought. Lastly, when we predict the label as "Hold", we do not take any action. If we predict the same label consecutively while executing a trade, we only focus on the first label and we perform the associated transaction. If a label is repeated, we ignore the label until the label changes. We assume that we start with \$10000 cash at the beginning and the trading commission is assumed to be \$1 per transaction since we are trading only liquid high-volume EFTs. We summarize the financial evaluation scenario in Eq. 19 where "S" indicates the financial evaluation scenario, "Money" indicates the total amount of cash, and "#OfStocks" represents the number of stocks.

$$S = \begin{cases} \#OfStocks = \dfrac{tMoney}{price} & \text{if label = 'Buy'} \\ tMoney = price * \#OfStocks & \text{if label = 'Sell'} \\ \text{no action} & \text{if label = 'Hold'} \end{cases} \tag{19}$$

One common financial evaluation metric for a strategy is Sharpe Ratio [62] which divides the method's annualized return relative to risk-free rate by annualized standard deviation where a 3-month US treasury bill is used to model the risk-free rate. Our remaining evaluation metrics are defined by the following equations:

$$AR = \left(\left(\frac{totalMoney}{startMoney}\right)^{\frac{1}{numberOfYears}} - 1\right) * 100 \tag{20}$$

$$AnT = \frac{transactionCount}{numberOfYears} \tag{21}$$

$$PoS = \frac{successTransactionCount}{transactionCount} * 100 \tag{22}$$

$$ApT = \frac{totalPercentProfit}{transactionCount} * 100 \tag{23}$$

$$L = \frac{totalTransactionLength}{transactionCount} * 100 \tag{24}$$

$$IdleR = \frac{data.length - totalTransLength}{data.length} * 100 \tag{25}$$

where AR represents the annualized return, AnT represents the annualized number of transactions, PoS represents the percent of success, ApT represents the average percent profit per transaction, L represents the average transaction length in days, and IdleR represents the idle ratio. Additionally, MpT and MlT represent the maximum profit/loss percentage in transaction respectively. We have implemented transformer techniques and baseline approaches mainly by using HuggingFace [63] and Pytorch [64]. We have also employed various Java libraries, including Spark, and Hadoop among others, to facilitate the implementation and execution of the experiments. Our code and processed datasets are available at https://github.com/seferlab/price_transformer.

V. RESULTS

A. COMPARISON VIA TRADITIONAL METRICS

As indicated in Table 3, ConvMixer algorithm exhibits superior performance in terms of test accuracy. Among the tested vision transformers, the ViT architecture closely trails the ConvMixer, with the most notable divergence found in test accuracy, where there exists a substantial margin exceeding 8%. Among the other vision transformers, the hierarchical transformer Swin performs better than DeiT. Among the all considered baselines, the best-performing baseline is CNN-TA++, and it performs worse than all transformers except DeiT in terms of accuracy. We retrieved the performance of

TABLE 3. Train and test performance of the tested algorithms and baselines.

Algorithm	Train Loss	Train Accuracy	Test Accuracy	Test F1
ViT	5,27%	98,31%	79,90%	0.71
ConvMixer	1,58%	99,57%	87,94%	0.32
DeiT	14,83	60,32%	44,24%	0.12
Swin	34,15%	89,57%	77,49%	0.47
CNN-TA++	34,00%	86,26%	62,06%	0.65
BaH	-	-	58,04%	0.61
LSTM	37,12%	83,21%	61,11%	0.49
MLP	39,15%	83,12%	60,55%	0.45
RSI	-	-	58,34%	0.46
SMA	-	-	56,01%	0.47

all baselines other than CNN-TA++ from a similar analysis applied in [23].

However, in terms of F1 score, vision transformer ViT outperforms all other methods. Even though the patch embedding-based non-transformer approach ConvMixer performs quite well in terms of accuracy, this is not correct in terms of F1 score. This can be explained by the fact that there are more "Hold" signals in our datasets. Such imbalance among the classes causes problems for ConvMixer and all other transformers except ViT. These "Hold" signals are more learnable for ViT. Our dataset can be considered as limited for deeper and more powerful architectures, so it is easier for ViT to learn the True Positives of "Hold", "Buy" and "Sell" signals.

We can observe a detailed breakdown of prediction performance in terms of each ETF as in Table 4. According to this table, ViT performs the best for almost all ETFs in terms of F1 score.

B. CONFUSION MATRICES

Confusion matrices for tested architectures elucidates notable strengths and challenges inherent in methods classification performance. We report the performance of tested approaches as shown in Tables 5-8. For instance, for ConvMixer in Table 5, the model demonstrates proficiency in accurately identifying certain instances within the "Hold" class; however, a considerable number of false negatives (744) is observed. Conversely, the model encounters more pronounced challenges in classifying instances within the "Buy" and "Sell" classes. Notably, the "Buy" class is characterized by a complete absence of true positives, indicating a deficiency in the model's ability to correctly identify instances of this class. In the case of the "Sell" class, the model exhibits a noteworthy occurrence of false positives (782), signifying instances where the model incorrectly predicts the presence of "Sell" orders. These detailed findings underscore the imperative of addressing and refining ConvMixer architecture, particularly in enhancing its sensitivity to "Buy" class instances and reducing false positives in the prediction of "Sell" orders.

In terms of Swin architecture as seen in Table 6, the examination of its confusion matrix illuminates its proficiency in accurately identifying instances within the "Hold" class, underscored by a notable count of true positives (898) and a minimal occurrence of false negatives (196). However, the model's performance encounters challenges in both the "Buy" and "Sell" classes. Specifically, the "Buy" class exhibits a diminished number of true positives (2) and a comparatively elevated count of false positives (40), implying difficulties in the accurate identification of instances within this category. Similarly, in the "Sell" class, the model demonstrates limited success, with only 3 true positives and 167 false positives. This implies a notable struggle in effectively distinguishing between instances belonging to the "Sell" and "Hold" classes.

In terms of ViT architecture as seen in Table 7, its confusion matrix unveils the model's proficiency in accurately identifying instances within the "Hold" class, as underscored by a notable count of true positives (799) and comparatively low occurrences of false positives (64) and false negatives (295). However, its performance encounters challenges in the "Buy" class, characterized by 33 true positives, 42 false negatives, and 101 false positives. This suggests a difficulty in the accurate discrimination of instances between the "Buy" and "Hold" categories. Notably, the model demonstrates a contrasting strength in the "Sell" class, exhibiting 53 true positives and 194 false negatives, indicative of adeptness in identifying instances of the "Sell" category. This intriguing observation points to a particular challenge in distinguishing between instances of the "Sell" and "Hold" categories.

Lastly, the evaluation of the confusion matrix for DeiT transformer as in Table 8 reveals a suboptimal performance, characterized by a meager count of correct predictions (151) against a significantly elevated number of false predictions (1093). This disparity results in a notably low F1 score, indicative of the model's limited ability to accurately identify instances within any given category. The discerned inability of the model to effectively distinguish between classes is a prominent factor contributing to the overarching poor results observed. Consequently, the implemented trading strategy yields unfavorable outcomes, underscoring the imperative of addressing and refining DeiT architecture.

C. ALGORITHMIC TRADING PORTFOLIO ANALYSIS

We also evaluated the portfolio analysis of the financial strategies by using the results of our tested algorithms predictions. Table 9 presents a comprehensive overview of the aggregated outcomes of vision transformer approaches such as ViT, DeiT, Swin, non-transformer patch embedding-based approach ConvMixer, and the best-performing baseline CNN-TA++. Similar to more traditional machine learning metrics, ViT outperforms all remaining methods.

A detailed comparison between ViT and the best-performing enhanced baseline CNN-TA++, focusing on the Annualized Number of Transactions and Average Transaction Length, reveals distinct trading strategy approaches. Specifically, CNN-TA++ executed 80 transactions, nearly doubling ViT's 43 transactions. However, a counter-intuitive

TABLE 4. F1 score and accuracy comparison between multiple architectures.

	F1					Accuracy				
	ConvMixer	ViT	DeiT	Swin	CNN-TA++	ConvMixer	ViT	DeiT	Swin	CNN-TA++
EWH	0.32	0.47	0.42	0.43	0.48	0.87	0.66	0.87	0.56	0.67
EWZ	0.35	0.46	0.39	0.44	0.45	0.86	0.66	0.86	0.65	0.64
QQQ	0.32	0.44	0.37	0.39	0.40	0.54	0.57	0.87	0.51	0.45
SPY	0.33	0.45	0.40	0.40	0.44	0.73	0.61	0.86	0.50	0.65
XLE	0.38	0.47	0.40	0.44	0.47	0.70	0.67	0.83	0.59	0.64
XLF	0.31	0.48	0.45	0.52	0.47	0.83	0.65	0.87	0.71	0.63
XLP	0.31	0.49	0.39	0.44	0.46	0.85	0.69	0.86	0.57	0.61
XLU	0.35	0.46	0.41	0.43	0.45	0.81	0.66	0.84	0.58	0.59
XLY	0.32	0.44	0.46	0.45	0.46	0.84	0.59	0.85	0.64	0.59

TABLE 5. ConvMixer confusion matrix.

	Hold	Buy	Sell
Hold	350	0	744
Buy	37	0	38
Sell	30	0	45

TABLE 6. Swin confusion matrix.

	Hold	Buy	Sell
Hold	898	36	160
Buy	66	2	7
Sell	68	4	3

TABLE 7. ViT confusion matrix.

	Hold	Buy	Sell
Hold	799	101	194
Buy	42	33	0
Sell	22	0	53

TABLE 8. DeiT confusion matrix.

	Hold	Buy	Sell
Hold	82	111	901
Buy	6	9	60
Sell	5	10	60

relationship emerges when examining transaction lengths, with CNN-TA++ exhibiting an average length of 6 transactions compared to ViT's 10 transactions. Both algorithms remained active in their respective strategies, accumulating 480 and 430 transaction points for CNN-TA++ and ViT, respectively. This detailed portfolio analysis provides valuable insights into the divergent trading strategies employed by these algorithms.

We also tested for the impact of COVID-19 in our analysis by focusing only on the period between 1/1/2002 and 1/1/2020, removing the last 2 years of the whole dataset where COVID-19 impact has started to be seen across financial markets. In this case, Table 10 reports a comprehensive overview of the aggregated outcomes of vision transformer approaches such as ViT, DeiT, Swin, non-transformer patch embedding-based approach ConvMixer, and the best-performing baseline CNN-TA++. In this case, ViT's performance is similar to ConvMixer in terms of multiple financial evaluation criteria, slightly different than Table 9 where ViT outperforms all remaining methods. A detailed comparison between ViT and the best-performing enhanced baseline CNN-TA++, focusing on the Annualized Number of Transactions and Average Transaction Length, reveals distinct trading strategy approaches. Specifically, CNN-TA++ executed 6 transactions, nearly one-ninth of ViT's 53 transactions. However, a counter-intuitive relationship emerges when examining transaction lengths, with CNN-TA++ exhibiting an average length of 165 transactions compared to ViT's 11 transactions. Similar to Table 9, this detailed portfolio analysis provides valuable insights into the divergent trading strategies employed by these algorithms.

D. FINANCIAL EVALUATION BY WEALTH CURVE OVER TIME

Figure 5 shows wealth curves of all considered approaches and the best-performing baseline CNN-TA++. In the context of strategy performance, our evaluation identifies the ViT architecture as the optimal choice, with a narrow gap compared to the Enhanced CNN-TA architecture. Both these algorithms consistently yield returns exceeding 120%. Intriguingly, despite ConvMixer's great performance in terms of traditional machine learning metrics, it ranks second to the last when evaluated from a more financial, portfolio evaluation perspective. Consequently, the most favorable algorithm in terms of strategy profitability is the ViT architecture. Nonetheless, it is essential to acknowledge that various considerations, including average transaction length, transaction frequency, and fluctuation in capital values, can influence the optimal choice of algorithm.

According to Figure 5, DeiT algorithm demonstrates a notable frequency of transactions; however, it frequently falters in effectively identifying optimal entry and exit points. As a consequence, following an initial phase of profitability, the algorithm undergoes a gradual decrement, culminating in an ultimate negative balance relative to the initial capital. This observed outcome underscores the algorithm's limited capacity to apprehend the inherent strategic logic, as elucidated by the diminished accuracy metrics in previous tables. This discernment not only accentuates the algorithm's suboptimal performance but also suggests a deficiency in its ability to align with the fundamental principles of the underlying

TABLE 9. General comparison of ConvMixer, ViT, CNN-TA++, DeiT, and Swin architectures.

	ConvMixer	ViT	CNN-TA++	DeiT	Swin
Sharpe Ratio (Daily)	0.13	0.15	0.12	-0.02	-0.12
Ending Capital	15778.9	22098.77	22070.29	8466.28	8331.54
Annualized return	9.55%	17.19%	17.16%	-3.28%	-3.58%
Annualized number of transaction	38	43	80	71	28
Percent success of transaction	52.63%	72.09%	71.25%	56.34%	53.57%
Average percent profit per transaction	1.42%	2.14%	1.13%	-0.17%	-0.46%
Average transaction length	13	10	6	1	11
Maximum profit percent in transaction	14.55%	12.84%	8.24%	4.14%	8.85%
Maximum loss percent in transaction	-16.11%	-27.97%	-18.63%	-8.98%	-19.36%
Maximum capital value	15778.9	22098.77	22070.29	10810.12	12590.97
Minimum capital value	9334.11	9365.84	9554.23	7267.91	7639.49
Idle Ratio	60.21%	65.19%	55.63%	89.79%	73.55%

TABLE 10. General comparison of ConvMixer, ViT, CNN-TA++, DeiT, and Swin architectures during pre-COVID period by focusing only on the period between 1/1/2002 and 1/1/2020, removing the last 2 years of the whole dataset where COVID-19 impact has started to be seen across financial markets.

	ViT	Swin	DeiT	ConvMixer	CNN-TA++
Sharpe Ratio	0.30	0.02	-0.14	0.46	0.25
Ending Capital	16520.68	8.26	6324.5	15923.07	15777.46
Annualized return	10.56%	-75.82%	-8.76%	9.75%	9.55%
Annualized number of transaction	53	64	11	6	45
Percent success of transaction	83.02%	82.81%	100.0%	83.33%	77.78%
Average percent profit per transaction	0.99%	130.83%	-104.2%	9.64%	1.06%
Average transaction length	11	6	73	165	11
Maximum profit percent in transaction	3.09%	-275.56%	-167.86%	47.32%	5.78%
Maximum loss percent in transaction	-8.58%	-200.0%	-77.78%	-14.67%	-6.18%
Maximum capital value	16520.68	17619026.81	51745.06	15923.07	15777.46%
Minimum capital value	10000.0	-73530615.59	-30538.48	9982.11	10000.0
Idle Ratio	51.57%	66.93%	35.4%	19.95%	58.41%

trading strategy. Further exploration and analysis are required to elucidate the specific aspects contributing to the observed inadequacies and to inform potential enhancements for algorithmic optimization.

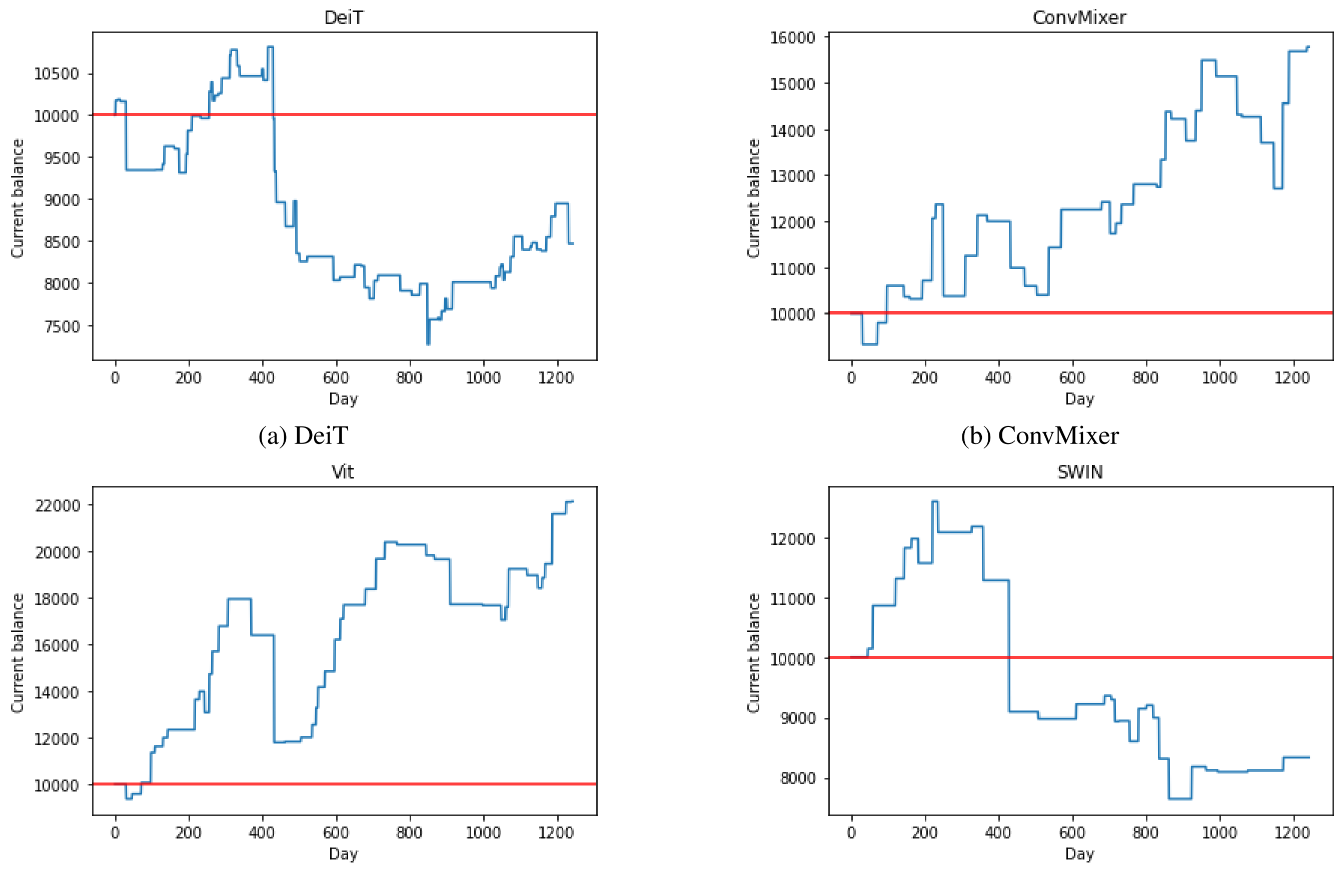
Similarly, Swin algorithm exhibits a low volume of transactions and fails to generate a profit with more than half of the transaction actions, thus resulting in a negative balance. This outcome contradicts the relatively high accuracy and F1 score metrics observed in Table 3. This discrepancy can be attributed to the algorithm's inability to identify the ideal entry and exit points, as reflected in the low precision and recall metrics for "Sell" and "Buy" orders seen in its confusion matrix. This result is further supported by the high number of false positives and false negatives in the identification of "Buy" and "Sell" orders, respectively. From a strategic standpoint, these findings indicate the need to address and reduce the prevalence of incorrect "Buy" and "Sell" orders, as these errors are recurrent and contribute significantly to capital loss.

Lastly, as seen in Figure 5, ConvMixer has a certain tendency towards "Hold" orders. This strategic inclination is indicative of a discerning recognition that financial losses often emanate from the execution of "Buy" and "Sell" orders. Consequently, ConvMixer algorithm partakes in a lower frequency of transactions relative to alternative algorithms. While this conservative approach mitigates exposure to potential losses, it simultaneously constrains the algorithm's profit-generation capacity. These empirical observations collectively show that, although the ConvMixer strategy is characterized by a risk-averse approach, it may not optimize profitability. Our comprehensive analysis of the ConvMixer algorithm's outcomes underscores the pivotal importance of striking a balance between risk mitigation and profit maximization in the quest for efficacious trading strategies.

In summary, each algorithm employs a unique approach to processing both financial data and executing trading strategies. Distinct algorithmic priorities are observed, with some algorithms placing a paramount emphasis on precision, while others prioritize profitability, sometimes at the cost of achieving the utmost precision in their predictions. Such observation underscores the inherent trade-offs and strategic decisions that deep learning algorithms must navigate when applied within the domain of financial time series analysis. The detailed interplay between precision and profitability underscores the complexity of algorithmic decision-making in financial markets, necessitating a thoughtful consideration of the strategic objectives and inherent risks associated with each algorithmic approach. These observations contribute to a deeper understanding of the intricate dynamics involved in optimizing learning algorithms for effective and strategic financial market operations.

E. THE PERFORMANCE ACROSS BALANCED DATASET

We also report portfolio performance results on balanced dataset, which is formed as defined in Section II-B. Table 11



(a) DeiT (b) ConvMixer

(c) ViT (d) Swin

FIGURE 5. Wealth curves for tested deep learning approaches.

TABLE 11. General comparison of ConvMixer, ViT, CNN-TA++, DeiT, and Swin architectures over balanced dataset.

	ConvMixer	ViT	CNN-TA++	DeiT	Swin
Sharpe Ratio (Daily)	0.14	0.16	0.12	0.11	0.13
Ending Capital	17212.3	24121.17	19898.71	17213.45	17989.11
Annualized return	10.11%	18.11%	17.55%	12.34%	12.78%
Annualized number of transaction	41	49	95	75	61
Percent success of transaction	55.14%	75.11%	65.25%	69.35%	68.23%
Average percent profit per transaction	1.35%	1.53%	0.69%	0.92%	1.12%
Average transaction length	12	10	5	11	11
Maximum profit percent in transaction	15.99%	13.99%	9.67%	11.15%	10.95%
Maximum loss percent in transaction	-15.55%	-24.10%	-17.77%	-13.02%	-14.04%
Maximum capital value	17121.1	23155.23	24111.45	19820.04	19770.00
Minimum capital value	9512.135	9565.34	9511.14	9113.77	9039.23
Idle Ratio	57.41%	62.58%	57.25%	64.99%	68.88%

presents a comprehensive overview of the aggregated outcomes of vision transformer approaches such as ViT, DeiT, Swin, non-transformer patch embedding-based approach ConvMixer, and the best-performing baseline CNN-TA++. Similar to imbalanced dataset case, ViT outperforms all remaining methods. The performance of Swin and DeiT transformers are better than their performance on imbalanced datasets.

F. STATISTICAL SIGNIFICANCE OF RESULTS

According to the statistical significance tests shown in Table 12, ViT-based asset price prediction method's behavior is not impacted by distinct periods and changing market states such as 2002–2012, and 2012-2022 time intervals. For many evaluation criteria, ViT operates quite robustly and stably. Although the performance difference between ViT-based price prediction methods under various distinct market conditions is not statistically significant, it outperforms the competing baselines like MLP, LSTM, SMA, RSI, and Buy & Hold especially during non-bull market conditions.

In addition to the summary of our strategy-based automated trading in Table 12, we also provide statistical significance results by comparing the performance of various transformers and baseline approaches as in Table 13.

TABLE 12. t-test results and average results of ViT model across different periods.

	t-test	2002-2012	2012-2022
Sharpe Ratio (Daily)	-	0.13	0.17
Annualized return	0.108	15.31%	18.65%
Annualized number of transaction	0.317	41	45
Percent success of transaction	0.076	70.65%	73.98 %
Average percent profit per transaction	0.465	2.15%	2.25%
Average transaction length	0.547	11	10
Maximum profit percent in transaction	0.653	12.92%	12.65%
Maximum loss percent in transaction	0.667	-27.62%	-28.35%
Idle Ratio	0.074	64.02%	66.35%

TABLE 13. t-test for annualized ETF returns. * corresponds to statistically significant performance differences.

Time Period	Comparison	p-value
2002–2012	ViT-Swin	0.001*
	ViT-ConvMixer	0.095
	ViT-DeiT	0.032*
	ViT-BaH	0.001*
	ViT-LSTM	0.009*
	ViT-MLP	0.013*
2012–2022	ViT-Swin	0.002*
	ViT-ConvMixer	0.023*
	ViT-DeiT	0.064
	ViT-BaH	0.001*
	ViT-LSTM	0.007*
	ViT-MLP	0.031*

A performance comparison between methods is considered to have a statistically significant difference if the p-value is less than 0.05. According to these results, ViT performs significantly better than simpler baselines such as BaH, LSTM, and MLP across different periods. However, the statistical significance of its outperformance concerning other methods depends on the considered period.

VI. CONCLUSION

In this study, we have undertaken a comprehensive investigation into the performance of various deep vision transformer-based algorithms and two-dimensional deep patch embedding-based convolutional neural networks in financial asset price and direction prediction. Our techniques integrate several well-known technical analysis indicators into the algorithmic trading prediction framework. We analyze temporally changing ETF datasets by first forming two-dimensional images out of the original price data via technical indicators. We designed meaningful and profit-making trades by inferring entry and exit points via 3 categories such as Buy, Sell, and Hold. According to our experiments, transformer-based methods consistently outperform the Buy and Hold baseline, LSTM, and the enhanced version of the only convolutional CNN-TA architecture in terms of both traditional machine learning metrics and financial portfolio analysis metrics. In addition to transformers, patch-embedding-based convolutional architecture is also effective in asset price and direction performance.

The principal objective of this research has been to gain a deeper understanding of the inherent strengths and weaknesses of each distinct transformer-based algorithm and how these attributes may be harnessed to enhance the performance of algorithmic trading strategies. Although we obtained reasonably well performance by transformer-based architectures, some more enhancements could be integrated into our framework. In future work, current analysis on ETFs can be enhanced to cryptocurrencies which are more volatile than ETFs. Such extension to multiple cryptocurrencies will also result in a massive amount of training data for our transformer-based deep learning methods, where the performance of transformers is known to be impacted by the training size. Moreover, the ordering of technical indicators while forming the image-like data could be enhanced as well. Another enhancement could be to focus more on different trading strategies instead of just long-only strategies. Lastly, we can integrate Explainable AI into our framework where Explainable AI methods for convolutional methods have been previously studied. However, similar explainable AI methods for vision transformer-based architectures have not been studied as much. Overall, the compelling prospects for future research underscore the continued importance of exploring and advancing the application of AI in the domain of financial time series analysis.

ACKNOWLEDGMENT

The authors would like to thank the editors and anonymous reviewers for providing insightful suggestions and comments to improve the quality of the research article.

REFERENCES

[1] M. Ge, S. Zhou, S. Luo, and B. Tian, "3D tensor-based deep learning models for predicting option price," in *Proc. Int. Conf. Inf. Sci. Commun. Technologie*, 2021, pp. 1–6.

[2] W. Zheng, "Exchange-traded fund price prediction based on the deep learning model," in *Proc. China Autom. Congr. (CAC)*, Oct. 2021, pp. 7429–7434.

[3] S. E. Freeda, T. C. E. Selvan, and I. G. Hemanandhini, "Prediction of Bitcoin price using deep learning model," in *Proc. 5th Int. Conf. Electron., Commun. Aerosp. Technol. (ICECA)*, Dec. 2021, pp. 1702–1706.

[4] R. C. Cavalcante, R. C. Brasileiro, V. L. F. Souza, J. P. Nobrega, and A. L. I. Oliveira, "Computational intelligence and financial markets: A survey and future directions," *Exp. Syst. Appl.*, vol. 55, pp. 194–211, Aug. 2016.

[5] B. Seyhan and E. Sefer, "NFT primary sale price and secondary sale prediction via deep learning," in *Proc. 4th ACM Int. Conf. AI Finance*. New York, NY, USA: Association for Computing Machinery, Nov. 2023, pp. 116–123, doi: 10.1145/3604237.3626896.

[6] A. M. Ozbayoglu, M. U. Gudelek, and O. B. Sezer, ''Deep learning for financial applications : A survey,'' *Appl. Soft Comput.*, vol. 93, Aug. 2020, Art. no. 106384.

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

[8] M. D. Zeiler and R. Fergus, ''Visualizing and understanding convolutional networks,'' in *Computer Vision—ECCV*. Zurich, Switzerland: Springer, 2014, pp. 818–833.

[9] K. He, X. Zhang, S. Ren, and J. Sun, ''Deep residual learning for image recognition,'' 2015, *arXiv:1512.03385*.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ''BERT: Pre-training of deep bidirectional transformers for language understanding,'' 2019, *arXiv:1810.04805*.

[11] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, ''RoBERTa: A robustly optimized BERT pretraining approach,'' 2019, *arXiv:1907.11692*.

[12] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, ''XLNet: Generalized autoregressive pretraining for language understanding,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1–11.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. U. Kaiser, and I. Polosukhin, ''Attention is all you need,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, 2017, pp. 1–11.

[14] R. E. Turner, ''An introduction to transformers,'' 2023, *arXiv:2304.10557*.

[15] N. N. Soylu and E. Sefer, ''BERT2OME: Prediction of 2'-O-methylation modifications from RNA sequence by transformer architecture based on BERT,'' *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 20, no. 3, pp. 2177–2189, May/Jun. 2023.

[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, ''An image is worth 16×16 words: Transformers for image recognition at scale,'' 2020, *arXiv:2010.11929*.

[17] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, ''Training data-efficient image transformers & distillation through attention,'' in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10347–10357.

[18] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, ''Swin transformer: Hierarchical vision transformer using shifted windows,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 10012–10022.

[19] A. Trockman and J. Z. Kolter, ''Patches are all you need?'' 2022, *arXiv:2201.09792*.

[20] Y. Santur, ''Deep learning based regression approach for algorithmic stock trading: A case study of the Bist30,'' *Gümüshane Üniversitesi Fen Bilimleri Dergisi*, vol. 10, no. 4, pp. 1195–1211, 2020.

[21] Y. K. Pardeshi and Prof. P. Kale, ''Technical analysis indicators in stock market using machine learning: A comparative analysis,'' in *Proc. 12th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT)*, Jul. 2021, pp. 1–6.

[22] E. Hossain, M. S. Hossain, P.-O. Zander, and K. Andersson, ''Machine learning with belief rule-based expert systems to predict stock price movements,'' *Exp. Syst. Appl.*, vol. 206, Nov. 2022, Art. no. 117706. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422009940

[23] O. B. Sezer and A. M. Ozbayoglu, ''Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach,'' *Appl. Soft Comput.*, vol. 70, pp. 525–538, Sep. 2018.

[24] E. C. M. Hui and K. K. K. Chan, ''Alternative trading strategies to beat 'buy-and-hold,''' *Phys. A, Stat. Mech. Appl.*, vol. 534, Nov. 2019, Art. no. 120800. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378437119304108

[25] M. R. Vargas, C. E. M. dos Anjos, G. L. G. Bichara, and A. G. Evsukoff, ''Deep leaming for stock market prediction using technical indicators and financial news articles,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[26] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[27] T. Fischer and C. Krauss, ''Deep learning with long short-term memory networks for financial market predictions,'' *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, Oct. 2018.

[28] J.-Z. Wang, J.-J. Wang, Z.-G. Zhang, and S.-P. Guo, ''Forecasting stock indices with back propagation neural network,'' *Exp. Syst. Appl.*, vol. 38, no. 11, pp. 14346–14355, Oct. 2011.

[29] Z. Liao and J. Wang, ''Forecasting model of global stock index by stochastic time effective neural network,'' *Exp. Syst. Appl.*, vol. 37, no. 1, pp. 834–841, Jan. 2010.

[30] A.-S. Chen, M. T. Leung, and H. Daouk, ''Application of neural networks to an emerging financial market: Forecasting and trading the Taiwan stock index,'' *Comput. Oper. Res.*, vol. 30, no. 6, pp. 901–923, May 2003.

[31] E. Guresen, G. Kayakutlu, and T. U. Daim, ''Using artificial neural network models in stock market index prediction,'' *Exp. Syst. Appl.*, vol. 38, no. 8, pp. 10389–10397, Aug. 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417411002740

[32] O. B. Sezer, A. M. Ozbayoglu, and E. Dogdu, ''An artificial neural network-based stock trading system using technical analysis and big data framework,'' in *Proc. SouthEast Conf.* New York, NY, USA: Association for Computing Machinery, Apr. 2017, pp. 223–226.

[33] D. Zhang and L. Zhou, ''Discovering golden nuggets: Data mining in financial application,'' *IEEE Trans. Syst., Man, Cybern., C*, vol. 34, no. 4, pp. 513–522, Nov. 2004.

[34] S. K. Chalup and A. Mitschele, *Kernel Methods in Finance*. Berlin, Germany: Springer, 2008, pp. 655–687, doi: 10.1007/978-3-540-49487-4_27.

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

[36] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ''ImageNet classification with deep convolutional neural networks,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1–9.

[37] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, ''Large-scale video classification with convolutional neural networks,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1725–1732.

[38] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, ''A convolutional neural network for modelling sentences,'' in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 655–665.

[39] T. Wolf et al., ''Transformers: State-of-the-art natural language processing,'' in *Proc. Conf. Empirical Methods Natural Lang. Processing: Syst. Demonstrations*, Oct. 2020, pp. 38–45.

[40] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, ''Transformers in vision: A survey,'' *ACM Comput. Surv.*, vol. 54, no. 10, pp. 1–4, Dec. 2021.

[41] X. Ding, Y. Zhang, T. Liu, and J. Duan, ''Deep learning for event-driven stock prediction,'' in *Proc. 24th Int. Conf. Artif. Intell.*, 2015, pp. 2327–2333.

[42] M. Längkvist, L. Karlsson, and A. Loutfi, ''A review of unsupervised feature learning and deep learning for time-series modeling,'' *Pattern Recognit. Lett.*, vol. 42, pp. 11–24, Jun. 2014.

[43] C. Krauss, X. A. Do, and N. Huck, ''Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500,'' *Eur. J. Oper. Res.*, vol. 259, no. 2, pp. 689–702, Jun. 2017.

[44] A. Yoshihara, K. Fujikawa, K. Seki, and K. Uehara, ''Predicting stock market trends by recurrent deep neural networks,'' in *PRICAI: Trends in Artificial Intelligence*, D.-N. Pham and S.-B. Park, Eds. Cham, Switzerland: Springer, 2014, pp. 759–769.

[45] O. B. Sezer, M. Ozbayoglu, and E. Dogdu, ''A deep neural-network based stock trading system based on evolutionary optimized technical analysis parameters,'' *Proc. Comput. Sci.*, vol. 114, pp. 473–480, Jan. 2017.

[46] W. Chen, C. K. Yeo, C. T. Lau, and B. S. Lee, ''Leveraging social media news to predict stock index movement using RNN-boost,'' *Data Knowl. Eng.*, vol. 118, pp. 14–24, Nov. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169023X17305839

[47] C.-Y. Lee and V.-W. Soo, ''Predict stock price with financial news based on recurrent convolutional neural networks,'' in *Proc. Conf. Technol. Appl. Artif. Intell. (TAAI)*, Dec. 2017, pp. 160–165.

[48] F. Feng, X. He, X. Wang, C. Luo, Y. Liu, and T.-S. Chua, ''Temporal relational ranking for stock prediction,'' *ACM Trans. Inf. Syst.*, vol. 37, no. 2, pp. 1–30, Mar. 2019, doi: 10.1145/3309547.

[49] Y. Xu and S. B. Cohen, ''Stock movement prediction from tweets and historical prices,'' in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, I. Gurevych and Y. Miyao, Eds. Jul. 2018, pp. 1970–1979. [Online]. Available: https://aclanthology.org/P18-1183

[50] Q. Zhang, C. Qin, Y. Zhang, F. Bao, C. Zhang, and P. Liu, ''Transformer-based attention network for stock movement prediction,'' *Exp. Syst. Appl.*, vol. 202, Sep. 2022, Art. no. 117239.

[51] Q. Ding, S. Wu, H. Sun, J. Guo, and J. Guo, "Hierarchical multi-scale Gaussian transformer for stock movement prediction," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, C. Bessiere, Ed. Jul. 2020, pp. 4640–4646.

[52] J. Liu, H. Lin, X. Liu, B. Xu, Y. Ren, Y. Diao, and L. Yang, "Transformer-based capsule network for stock movement prediction," in *Proc. 1st Workshop Financial Technol. Natural Lang. Process.*, Macao, China, Aug. 2019, pp. 66–73.

[53] N. Cohen, T. Balch, and M. Veloso, "Trading via image classification," 2019, *arXiv:1907.10046*.

[54] T. Tuncer, U. Kaya, E. Sefer, O. Alacam, and T. Hoser, "Asset price and direction prediction via deep 2D transformer and convolutional neural networks," in *Proc. 3rd ACM Int. Conf. AI Finance*. New York, NY, USA: Association for Computing Machinery, Nov. 2022, pp. 79–86, doi: 10.1145/3533271.3561738.

[55] A. S. Wafi, H. Hassan, and A. Mabrouk, "Fundamental analysis models in financial markets—Review study," *Proc. Econ. Finance*, vol. 30, pp. 939–947, Jan. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212567115013441

[56] B. Enkhsaikhan, "Yahoo finance asset price dataset," Tech. Rep., 2023, doi: 10.21227/87yd-jk77.

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

[58] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Appl. Soft Comput.*, vol. 97, Dec. 2020, Art. no. 105524. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494619302947

[59] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4055–4064.

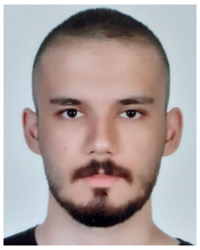
[60] H. Hu, Z. Zhang, Z. Xie, and S. Lin, "Local relation networks for image recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Jun. 2019, pp. 3464–3473.

[61] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[62] W. F. Sharpe, "Mutual fund performance," *J. Bus.*, vol. 39, pp. 119–138, Jan. 1966.

[63] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Davison, "Hugging-face's transformers: State-of-the-art natural language processing," 2020, *arXiv:1910.03771*.

[64] S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf



ABDUL HALUK BATUR GEZICI (Member, IEEE) received the bachelor's degree (Hons.) from the Computer Engineering Department, Engineering Faculty, Istanbul Aydin University, in 2021. He is currently pursuing the master's degree in artificial intelligence with Özyeğin University. His research interests include artificial intelligence and finance.

EMRE SEFER (Member, IEEE) received the B.Eng. degree from the Department of Computer Engineering, Boğaziçi University, in 2008, the M.S. degree in computer science from the University of Maryland, College Park, in 2011, and the Ph.D. degree in computational biology from Carnegie Mellon University, in 2015. After completing the Ph.D. degree, he was a short Postdoctoral Researcher with the CMU Machine Learning Department, under the supervision of Ziv-Bar Joseph. He is currently an Assistant Professor with the Computer Science Department, Özyeğin University. He has published several journals and conferences during the Ph.D. degree. His research interests include machine learning on graph and time-series datasets, especially in finance and bioinformatics application domains.