Computational Economics (2024) 64:2263–2354
https://doi.org/10.1007/s10614-023-10502-3

High-Frequency Trading in Bond Returns: A Comparison Across Alternative Methods and Fixed-Income Markets

David Alaminos[1] · **María Belén Salas**[2,3] · **Manuel A. Fernández-Gámez**[2,3]

Accepted: 16 October 2023 / Published online: 2 December 2023

Abstract

A properly performing and efficient bond market is widely considered important for the smooth functioning of trading systems in general. An important feature of the bond market for investors is its liquidity. High-frequency trading employs sophisticated algorithms to explore numerous markets, such as fixed-income markets. In this trading, transactions are processed more quickly, and the volume of trades rises significantly, improving liquidity in the bond market. This paper presents a comparison of neural networks, fuzzy logic, and quantum methodologies for predicting bond price movements through a high-frequency strategy in advanced and emerging countries. Our results indicate that, of the selected methods, QGA, DRCNN and DLNN-GA can correctly interpret the expected bond future price direction and rate changes satisfactorily, while QFuzzy tend to perform worse in forecasting the future direction of bond prices. Our work has a large potential impact on the possible directions of the strategy of algorithmic trading for investors and stakeholders in fixed-income markets and all methodologies proposed in this study could be great options policy to explore other financial markets.

Keywords Fixed-income markets · Bond returns · High-frequency trading · Deep learning · Fuzzy logic · Quantum computing

JEL Classification C63 · G12 · G14

✉ David Alaminos
alaminos@ub.edu

[1] Department of Business, University of Barcelona, Barcelona, Spain

[2] Department of Finance and Accounting, University of Málaga, Málaga, Spain

[3] Cátedra de Economía y Finanzas Sostenibles, University of Málaga, Málaga, Spain

1 Introduction

Fixed-income markets represent an important financing source for governments, domestic and international organizations, banks, and both private and public companies with access to the fixed-income market. The development of fixed-income trading can contribute to financial stability in general, and enhance financial intermediation by increasing competition and developing the associated financial infrastructure, products, and services (Nunes et al., 2019; International Monetary Fund, 2021). Government bonds are the main instrument of most fixed-income asset markets, for developed and developing economies alike. In the United States, the main data source for public securities trading activity is GovPX and MTS for Europe. The MTS is a fully electronic, quote-driven interbank market comprising multiple trading platforms. All MTS platforms use identical technology for trading; however, every platform maintains its rules set, and market participants (Biais & Green, 2019; Friewald & Nagler, 2019). Fixed-income securities are commonly traded over-the-counter (OTC), on inter-dealer wholesale platforms, and, less frequently, on retail platforms where liquidity is provided by pre-purchased dealers. Transactions are not anonymous and are bilateral; therefore, the conditions of negotiation are determined by search and trading frictions, in the absence of a focal point, dealers have to proactively seek out and negotiate with possible counterparties to get the "best" offer (Darbha & Dufour, 2013; Glode & Opp, 2020; Neklyudov, 2019).

Fixed-income trading has generally received less attention from researchers than equity market trading, even though fixed-income markets involve significantly more capital raising in comparison to equity markets. The electronic ease of bond trading, however, is on the rise. The impact of electronically supported trading on the performance of the fixed-income market will be interesting to evaluate as its contribution expands. In addition, some liquidity providers for corporate bonds offer requests for trades under specific trade size limits using algorithms instead of human participants (Bessembinder et al., 2020).

Technological advances have transformed how investors can operate in the financial markets. High-Frequency Trading (HFT), which is algorithmic trading (AT) distinguished by high-speed trade execution, exemplifies these changes in technology (Frino et al., 2020). HFT is an approach to a financial market intervention involving complex software tools, which are used to execute high-frequency trades, guided by mathematical algorithms, in the markets for stocks, options, bonds, derivative instruments, commodities, etc. (Rundo, 2019). Hendershott et al. (2011) claim that AT reduces trading costs and increases quote information. In addition, liquidity providers' revenues also increase with AT, although this effect seems to be transitory. In conclusion, financial trading demands that the AT scans the environment for suitable and prompt decisions in the absence of monitored data (Aloud & Alkhamees, 2021).

In academic research, the term "high-frequency trading" is often used to refer to asset price windows of 10 min or 60 min, for example (Christiansen & Ranaldo, 2007). For instance, a study published in the Journal of Financial

Markets in 2013 defines high-frequency trading as "trading that takes place in intervals of a few seconds to a few minutes" (Aldrich, 2013). Similarly, a paper published in the Journal of Financial Economics in 2008 found that HFT can improve liquidity provision in corporate bond markets, particularly for less liquid bonds (Mahanti et al., 2008).

However, in practice, the term "high-frequency trading" is often used more strictly to refer to price windows with even shorter times. For example, in the US equities market, the Securities and Exchange Commission defines a high-frequency trader as someone who trades at least 2 million shares or $20 million in securities in a single day, with an average holding time of less than 0.5 s (SEC, 2014).

Despite the varying definitions of high-frequency trading used in academic research and in practice, there is general agreement that this type of trading has significant implications for bond markets. Some studies have suggested that HFT can increase market efficiency and liquidity, while others have argued that it can exacerbate market volatility and lead to market instability (e.g., Frino et al., 2013; Schestag et al., 2016). As HFT continues to evolve and shape financial markets, it is likely that academic researchers and practitioners will continue to debate its effects and implications**.**

AT, both algorithms driven by fundamental and technical indicator analysis and algorithms supported by machine learning techniques, have been examined by several researchers. According to Goldblum et al., (2021), Machine Learning (ML) is playing an important and growing role in financial business applications. Besides, Deep learning (DL), which is a subclass of ML methods that study deep neural networks, develops DL algorithms that can be used to train complex data and predict the output. Today numerous financial firms, ranging from hedge firms, investment banks, and retailers to modern FinTech providers, are investing in developing expertise in data science and ML (Goodell e al., 2021).

As market turmoil and uncertainty in financial markets have increased considerably, ML algorithms are quite applicable for the analysis of financial markets and, in particular, the fixed-income market. The marketplace is very complicated, and the only forecast that can be made is its unpredictability. The financial market's unforeseeability is caused by the uncertainty of many episodes that occur in it (Goldblum et al., 2021). Deep Neural Networks draw knowledge from the data, that can then be utilised to forecast and produce further data. This feedback decreases unreliability by indicating specific problem-solving. ML is especially useful for handling problems where an analytical solution is not explicitly instructed to do so, such as complicated categorisation techniques or recognition of trends (Ghoddusi et al., 2019). The benefit of Deep Neural Network methods over those offered by classical statisticians and econometricians is that ML algorithms can handle a huge quantity of organised and non-structured information and provide quick predictions or conclusions (Milana & Ashta, 2021).

Publications on the use of ML techniques with specific applications to fixed-income markets are scarce. However, other financial areas have attracted much more interest in the research literature, particularly in the equity and foreign exchange markets (Nunes et al., 2018). Most of these studies involve the stock market, mainly for forecasting with artificial neural networks (ANN), support

vector machines (SVM), and random forests (RF) models. These methods have proven to produce excellent results for financial time series forecasting (Deng et al., 2021, 2022). For example, Kara et al. (2011) suggested an ANN-based model for predicting the daily price movement in the stock market, and it yielded high accuracy in the forecast. Akyildirim et al. (2022) compare the trading behaviour of several advanced forecasting techniques, such as ANN, autoregressive integrated moving average (ARIMA), nearest neighbors, naïve Bayes method, and logistic regression to forecast stock price movements relying on past prices. They apply these methods to high-frequency data of 27 blue-chip stocks traded on the Istanbul Stock Exchange. Their results highlight that, among the chosen methodologies, naïve Bayes, nearest neighbors, and ANN can detect the future price directions as well as percentage changes at a satisfactory level, while ARIMA and logistic regression perform worse than the random walk model. In addition, these authors establish a future line of research to test the chosen methods in other markets to achieve more accurate and widespread results.

Some authors have made predictions about the performance of fixed-income assets through neural networks. Vukovic et al. (2020) analyze the model of a neural network that forecasts the Sharpe ratio. Their results demonstrate that neural networks are accurate in predicting nonlinear series with an 82% precision in the test cases for forecasting the future Sharpe ratio dynamics and the position of the investor's portfolio. For future research, they propose analyzing more data in stronger artificial intelligence technologies, such as Long Short-Term Memory (LSTM) neural network technology. They conclude that these adaptive methodologies should provide more accurate analysis and forecasting and such an area of study requires additional attention and effort in the future. Li et al. (2021) analyze sovereign CDS to prevent investment risks and propose a hybrid ensemble forecasting model. They employ Autoregressive Integrated Moving Average (ARIMA) model to predict trend elements, meanwhile, the Relevance Vector Machine (RVM) technique is used to forecast market volatility and noise elements, correspondingly, having the model excellent robustness. They establish that although the suggested model exhibits satisfactory prediction efficiency, there is scope for further improvement and apart from sovereign CDS time series, the prediction model provided may be applied to other financial time series to test the generalizability of the model. Nunes et al. (2019) concentrate on yield curve forecasting, currently the centerpiece of the bond markets. They apply ML to fixed-income markets, specifically multilayer perceptrons (MLPs), to analyze the yield curve overall. They exhibit that MLPs could be effectively utilized to forecast the yield curve. They determine that, in terms of future work, an important area of interest is to keep exploring multitask learning, as they believe that further research is required to identify the terms and conditions under which their methodology could be applied with enhanced performance.

To fill the gap in this research area, our study aims to predict bond price movements based on past prices through high-frequency trading. We compare machine learning methods applied to the fixed-income markets (sovereign, corporate and high-yield debt) in advanced and emerging countries in the one-year bond market for the period from 15 February 2000 to 12 April 2023.

Despite the limited number of observations at 10-min frequency, the methodologies applied in this work are capable of working and finally making estimates, something that would be impossible for conventional statistical methodologies and even for some simple computational methodologies. Some previous works have found more avaibility of this data from OTC markets. Although these limitations exist, several works have also appeared investigating corporate bond data with 10-min interval observations, such as Nowak et al. (2009), Aldana (2017), Gomber and Haferkorn (2015), Holden et al. (2018), Gündüz et al. (2023).

We make at least two further contributions to the literature. First, we analyze the fixed-income market through HFT comparing a wide range of innovative computational machine learning methodologies, since most of the previous studies employ statistical and econometric methods. In addition, the prior literature deals with portfolio optimization only with fixed-income assets is not too many, and even fewer are dealing with the use of HFT. Within the ongoing advancement of financial markets, HFT proportion has increased steadily in recent years, which is generally characterized by fast update frequency and high trading speed. HFT also will produce plenty of profitable market influence, like increasing market liquidity and improving risk-handling ability (Deng et al., 2021). Second, our study has made predictions of bond price movements globally, and so not restricted to developed countries, being interesting for those responsible for the economic policies of any country in the world. Whereas the relevance of public debt markets has led to innumerable papers on these markets in the United States and other advanced countries, comparatively limited research exists on emerging bond markets (Bai et al., 2013). In addition, our study has considered not only sovereign bonds but also corporate and high-yield debt.

The rest of the paper is organized as follows. In Sect. 2, the methodologies are described. Section 3 details the sample and data involved in the research. Section 4 points out the results and findings obtained. By last, Sect. 5 finishes explaining the conclusions reached.

2 Methodologies

We have used different methods to predict bond price movements through HFT. The application of various techniques aims to obtain a robust model, which is tested not just via one categorisation technique but using those that have proven successful in prior literature and other areas. Specifically, this study applies Quantum-Fuzzy Approach, Adaptive Boosting, and Genetic Algorithm, Support Vector Machine-Genetic Algorithm, Deep Learning Neural Network- Genetic Algorithm, Quantum Genetic Algorithm, Adaptive Neuro-Fuzzy Inference System-Quantum Genetic Algorithm, Deep Recurrent Convolutional Neural Networks, Convolutional Neural Networks-Long Short Term Memory, Gated Recurrent Unit- Convolutional Neural Networks, and Quantum Recurrent Neural Networks. The techniques Deep Recurrent Convolutional Neural Networks and Quantum Genetic Algorithm have been the ones that have obtained the best results as will be shown in Sect. 4 of Results, therefore these methodologies will be explained below. The rest named are shown in the "Appendix 1" of this study.

2.1 Quantum Genetic Algorithm (QGA)

The quantum evolutionary algorithm (QEA) is an evolutionary algorithm built on the concept of quantum computing. It introduces notions like superposition states in quantum computing and incorporates the single encoding form to obtain improved experimental results in the combinatorial optimisation problem. Nevertheless, when it comes to the optimisation of multimodal functions using QEA, in the specific, high-dimensional multimodal function optimisation problem, it is likely to drop into local optimum and its computing efficiency is poor.

This study aims to improve the global optimisation capacity of the genetic algorithm and the local search ability according to the quantum probabilistic model to introduce a new type of quantum evolutionary algorithm, namely the "quantum genetic algorithm", to deal with the above deficiencies of QEA. This algorithm utilises the quantum probabilistic vector encoding mechanism and takes the crossover operator of the genetic algorithm and the updating strategy of quantum computation simultaneously to optimize the global search capacity of the quantum algorithm effectively.

The quantum genetic algorithm steps are:

2.1.1 Step 1: Population Initialisation

The lowest unit of information in QGA is a quantum bit. The state of a quantum bit can be 0 or 1, expressed as:

$$|\Psi\rangle = \alpha\,|0\rangle + \beta|1\,\rangle \tag{1}$$

being α, β two complex numbers corresponding to the likelihood of happening of the respective state: $\left(|\alpha|^2 + |\beta|^2 = 1\right), |\alpha|^2, |\beta|^2$ symbol the likelihood of the quantum bit in the 0 and 1 state accordingly.

The most commonly adopted coding techniques in EA include binary coding, decimal coding, and symbolic coding. In QGA, a new method of coding is introduced using the quantum bit, namely the use of a pair of complex numbers to describe a quantum bit. A system with m quantum bits is expressed as

$$\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \ldots & \alpha_m \\ \beta_1 & \beta_2 & \ldots & \beta_m \end{array}\right] \tag{2}$$

In the equation, $\left|\alpha_i\right|^2 + \left|\beta_i\right|^2 = 1$ (i = 1, 2, …, m). This method of display may be applied to describe any linear superposition of states. For instance, a system of three quantum bits having the next probability amplitudes:

$$\left[\begin{array}{c|c|c} \frac{1}{\sqrt{2}} & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{\sqrt{2}} & \frac{1}{2} & \frac{\sqrt{3}}{2} \end{array}\right] \tag{3}$$

The system state can be defined as

$$\begin{aligned}&\frac{\sqrt{3}}{4\sqrt{2}}|000\rangle+\frac{3}{4\sqrt{2}}|001\rangle+\frac{1}{4\sqrt{2}}|010\rangle+\frac{\sqrt{3}}{4\sqrt{2}}|011\rangle\\&+\frac{\sqrt{3}}{4\sqrt{2}}|100\rangle+\frac{\sqrt{3}}{4\sqrt{2}}|101\rangle+\frac{1}{4\sqrt{2}}|110\rangle+\frac{\sqrt{3}}{4\sqrt{2}}|111\rangle\end{aligned} \tag{4}$$

2.1.2 Step 2: Conduct Individual Coding and Measuring of the Population Generating Units

QGA is a probabilistic algorithm analogous to EA. The algorithm is $H(t)=\left\{Q_1^t,Q_2^t,\ldots Q_h^t,\ldots Q_l^t\right\}$ h = 1,2,…l) being h the size of the population, $Q_l(t)=\left\{q_1^t,q_2^t,\ldots q_j^t,\ldots q_n^t\right\}$ where n represents the number of generator units, t denotes the evolution generation,q_j^t symbols the binary coding of the generation volume of the jth generator unit. Its chromosome is shown as below:

$$q_j^t=\left[\begin{array}{c|c|c|c}\alpha_1^t & \alpha_2^t & \ldots & \alpha_m^t\\ \beta_1^t & \beta_2^t & \ldots & \beta_m^t\end{array}\right] \tag{5}$$

(j = 1, 2, …, n) (m is the length of the quantum chromosome).

During the ''initialization of H(t),'' if α_1^t, β_1^t (i = 1, 2, …, m) in $Q_l(t)$ and all the q_j^t are initialized, it denotes that all the possible linear superposition states will happen with equal likelihood. Over the step of ''generating S(t) from H(t)'', a common solution set S(t) is created through observation of the state of H(t), wherein the t_{th} generation, $S(t)=\{P_1^t,P_2^t,\ldots,P_h^t,\ldots,P_l^t\},P_l=\left\{x_1^t,x_2^t,\ldots,x_j^t,\ldots,x_n^t\right\}$. Every x_j^t (j = 1, 2, …, n) is a series, $\left(x_1,x_2,\ldots,x_i,\ldots,x_m\right)$, of length m, which are reached from the amplitude of quantum bit $\left|\alpha_i^t\right|^2$ or $\left|\beta_i^t\right|^2$ (i = 1, 2, …, m). The relevant procedure in the binary scenario is to randomly identify a number [0, 1]. Take ''1″ if it is larger than $\left|\alpha_i^t\right|^2$; take ''0″ otherwise.

2.1.3 Step 3: Make An Individual Measure for Every Item in S(t)

Employ a fitness assessment function to test each object in S(t) and maintain the best object in the generation. If you get a satisfactory solution, the algorithm stops; if not, proceed to the fourth step. When dealing with non-binary optimization problems, the chromosome is usually represented by a set of real-valued parameters rather than a binary string. In such cases, the fitness function is often a continuous function that maps the parameter values to a scalar value that represents the fitness of the solution.

Considering a non-binary optimization problem with a chromosome composed of three real-valued parameters ×1, ×2, and ×3, the fitness function for this problem would be defined as:

$$f(x1, x2, x3) = (x1 - 3)^2 + (x2 + 1)^2 + (x3 - 2)^2 \tag{6}$$

The objective in this case would be to minimize the fitness function. To accomplish this, the QGA would search for a set of parameter values that produce the minimum fitness value. The process would be similar to that for a binary problem, with the genetic operators applied to the real-valued parameters rather than binary strings.

2.1.4 Step 4: Apply Genetic Operators to Create New Individuals

The crossover operator is applied by swapping some of the qubits between two chromosomes. One of the most commonly used crossover operators in QGA is the uniform crossover, which selects each qubit from one of the two parent chromosomes with a certain probability. The crossover operator can be represented mathematically as:

$$|\psi child\rangle = \alpha|\psi parent1\rangle + \beta|\psi parent2\rangle|\psi child\rangle = \alpha|\psi parent1\rangle + \beta|\psi parent2\rangle \tag{7}$$

where $\alpha|\psi parent1\rangle + \beta|\psi parent2\rangle$ are the two parent chromosomes, $|\psi child\rangle$ is the resulting child chromosome, and α and β are complex coefficients determined by the crossover probability.

The mutation operator randomly flips some of the qubits in a chromosome. Mathematically, the mutation operator can be represented as:

$$|\psi mutated\rangle = Um|\psi original\rangle|\psi mutated\rangle = Um|\psi original\rangle \tag{8}$$

where Um is a single-qubit unitary gate that applies a random rotation around the Bloch sphere axis for the qubit to be mutated. The mutation rate determines the probability of applying the mutation operator to each qubit in a chromosome.

It's important to note that the application of genetic operators in QGA can be done in different ways, and the specific equations used can vary depending on the implementation and problem being solved.

2.1.5 Step 5: Apply An Appropriate Quantum Rotation Gate U(t) to Update S(t)

The conventional genetic algorithm utilises mating and mutation operations, etc. to keep the population diverse. The quantum genetic algorithm uses a logic gate to the likelihood amplitude of the quantum state to preserve the diversity of the

population. Hence, the method of updating by a quantum gate is the essence of the quantum genetic algorithm. The binary system, adaptation values, and the probability amplitude comparison technique are utilised for updating using a quantum gate in the classical genetic algorithm. This approach to updating via a quantum gate is adequate for solving combinatorial optimisation problems with an in-principle optimum. Nevertheless, for real optimisation problems, especially those optimisation problems of multivariable continuous functions, whose best solutions are in principle not available beforehand. Hence, a quantum rotation gate of the quantum logic gate for the new quantum genetic algorithm is assumed here.

$$U = \begin{bmatrix} \cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{9}$$

being θ the quantum gate rotation angle. Its value is shown as

$$\theta = k \cdot f\left(\alpha_i, \beta_i\right) \tag{10}$$

$$k = \pi \cdot \exp\left(-\frac{t}{iter_{\max}}\right) \tag{11}$$

We consider k as a variable linked to the evolution generation to adjust the mesh size in a self-adaptive way. Let t be the evolution generation, π is an angle, $iter_{\max}$ is a constant that relies on the complexity of the optimization problem. The aim of the function $f\left(\alpha_i, \beta_i\right)$ serves to cause the algorithm to seek the best direction. It is based on the idea of gradually bringing the actual search solution closer to the optimal solution and thus setting the direction of the quantum rotation gate.

Thus, the process of implementing the quantum rotation gate to the entire probability amplitude for the individual object in the population, namely by applying the quantum rotation gate U(t) to update S(t), in the quantum genetic algorithm may be written as:

$$S(t+1) = U(t) \times S(t) \tag{12}$$

being t the evolution generation, U(t) represents the t_{th} generation quantum rotation gate, S(t) symbols the t_{th} generation probability amplitude of a certain object, S(t + 1) denotes the $t + 1^{th}$ generation probability amplitude of the relevant object.

2.1.6 Step 6: Perturbation

Since QGA is inclined to get caught at a better local extreme value, we disturb the population. QGA analysis has shown that if the best individual of the present generation is a local extreme value, the algorithm is very difficult to free. Thus, the algorithm is stuck at the local extrema if the best individual remains unchanged in subsequent generations.

Finally, we show how is the pseudocode for the implementation of this method for the problem studied and the flowchart (Fig. 1) with the steps to follow as it has been described previously.

```
procedure QGA
begin
t←0

initiate Q(0)
make P(0) by observing Q(0)
evaluate P(0)
store the best solution among P(0)
while not termination-criterion do
t←t+1

make P(t) by observing Q(t-1) population
evaluate P(t)
update Q(t) employing quantum gates
U(θ_t)

store the best solution among P(t)
end while
end
```

Pseudocode of Quantum Genetic Algorithm

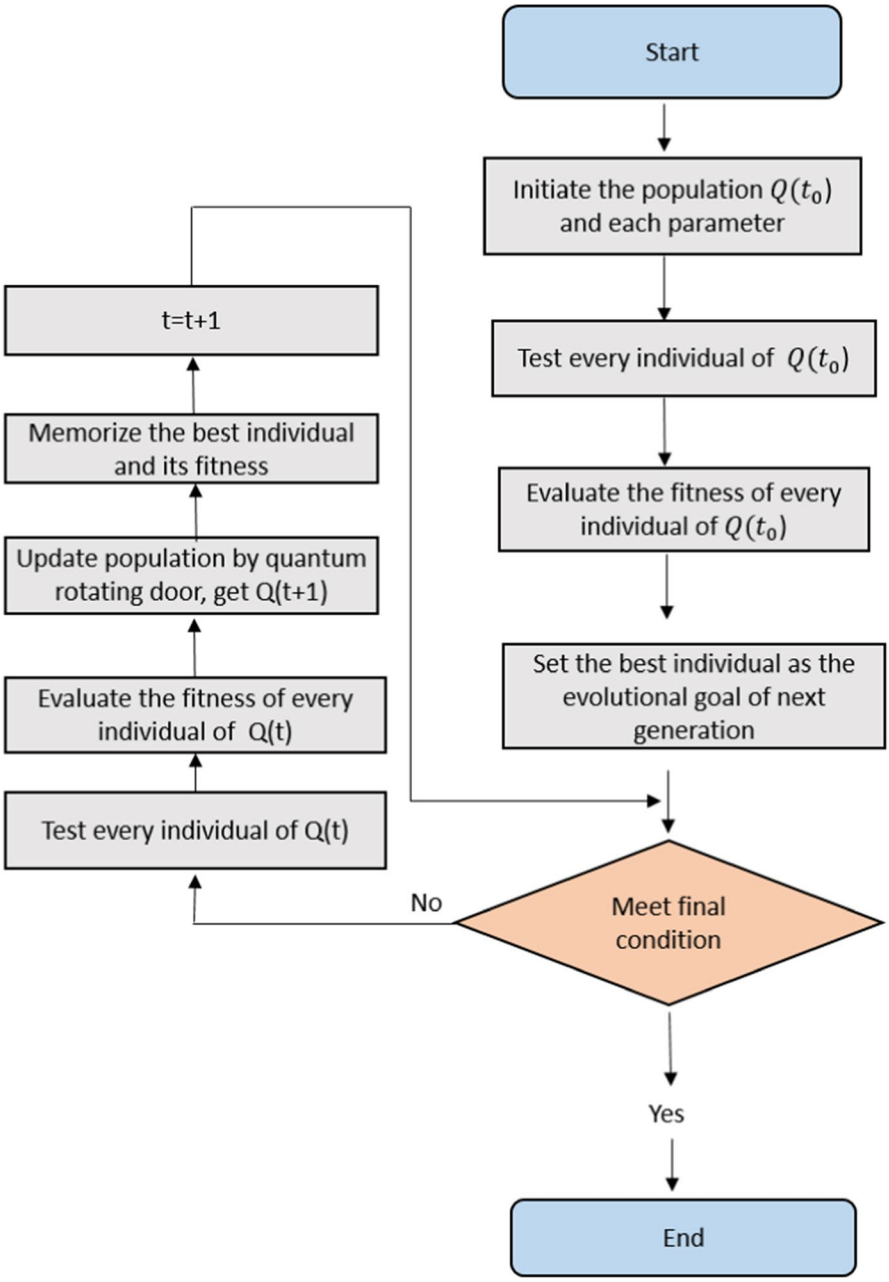

Fig. 1 Flowchart of quantum genetic algorithm

2.2 Deep Recurrent Convolutional Neural Network (DRCNN)

The RCNN model consists of a stack of RCLs and may include max pooling layers. To save on computational resources, the first layer is a standard forward convolutional layer with no recurrent connections, followed by a max pooling layer. Four RCLs are used with a max pooling layer in the middle, and there are only feed-forward links between adjacent RCLs. Both clustering operations have a stride of 2 and a size of 3. The fourth RCL's output tracks a global maximum clustering layer that produces the maximum of each feature map, resulting in a feature vector to represent the image. This approach differs from Krizhevsky et al. (2017) model, which uses fully connected layers, and Lin et al. (2013) and Szegedy et al.'s (2017) models, which use global average pooling. Finally, a softmax layer is used to classify the feature vectors into C categories, with the output consisting of:

$$\mathscr{y}_k = \frac{\exp\left(W_K^T X\right)}{\sum_{K'} \exp\left(W_K^T X\right)} (\mathrm{k} = 1, 2, \ldots, \mathrm{C}) \tag{13}$$

being $\mathscr{y}_k$ the predicted probability belonging to the kth category, and x the feature vector generated by the global max pooling.

RNNs have been deployed in many fields in time series forecasting with success owing to their enormous predictive power. The standard RNN framework is structured by the output, which depends on its past estimations (Wan et al., 2017). The standard RNN framework uses a hidden state to store information about past inputs, which is combined with the current input to make a prediction for the output at the current time step. The RCNN model incorporates the standard RNN framework by using Recurrent Convolutional Layers (RCLs) to capture the temporal dependencies in sequential data. The output of each RCL is a sequence of hidden states that can be used to make predictions about future inputs. The DRCNN model extends the RCNN model by stacking RCLs to create a deep architecture, with each layer applying a convolutional operation to the hidden states generated by the previous layer. The output of the last layer is then fed into a supervised learning layer to produce a prediction for the output at the current time step. the output of this RNN can be written as:

$$\mathrm{y_t} = \mathrm{f}\left(\mathrm{W_y} * \mathrm{s_t} + \mathrm{b_y}\right) \tag{14}$$

where yt is the output at time step t, st is the hidden state at time step t, Wy is the weight matrix connecting the hidden state to the output, by is the bias term, and f is the activation function.

An input sequence vector x, the hidden states of a recurrent layer s, and the output of a unique hidden layer y, can be obtained from formulas (14) and (15).

$$s_t = \sigma\left(W_{xs} x_t + W_{ss} s_{t-1} + b_s\right) \tag{15}$$

$$y_t = o\left(W_{so} s_t + b_y\right) \tag{16}$$

being W_{xs}, W_{ss}, and W_{so} the weights from the input layer x to the hidden layer s, the hidden layer to itself, and the hidden layer to its output layer, respectively. b_y represent the biases of the hidden layer and output layer. Formula (16) points out σ and o as a symbol of the activation functions.

$$STFT\{z(t)\}(\tau,\omega) = \int_{-\infty}^{+\infty} z(t)\omega\big(t-\tau\big)e^{-j\omega t}dt\big) \tag{17}$$

where $z(t)$ denotes the vibration signals, $\omega(t)$ symbols the Gaussian window function focused around 0. $T(\tau, \omega)$ represent a complex function defining the vibration signals over time and frequency. To compute the hidden layers with the convolutional operation formulas (17) and (18) are used.

$$S_t = \sigma\big(W_{TS} * T_t + W_{SS} * S_{t-1} + B_s\big) \tag{18}$$

$$Y_t = o\big(W_{YS} * S_t + B_y\big) \tag{19}$$

being W the convolution kernels. Below we show the pseudocode of activation function of these RCNNs:

Pseudocode for Activation function

```
# Activation function for identifying and ranking values
# Input: Characteristics from convolution layer
# Output: Removal of negative values
activation_function = lambda y: 1.0/(1.0 + np.exp(-y))
input_func = np.random.random((2, 3))
K1, a1 = np.random.random((4, 2)), np.random.random(4)
K2, a2 = np.random.random((1, 4)), np.random.random(1)
K3, a3 = np.random.random((1, 1)), np.random.random(1)
layer1 = activation_function(np.dot(K1, input_func) + a1)
layer2 = activation_function(np.dot(K2, layer1) + a2)
output = np.dot(K3, layer2) + a3
```

To establish a deep architecture, the recurrent convolutional neural network (RCNN) can be stacked and form the DRCNN (Huang & Narayanan, 2017). In this combination case, the last part of the model is a supervised learning layer, set by formula (19).

$$\hat{r} = \sigma\big(W_h * h + b_h\big) \tag{20}$$

being W_h the weight and b_h the bias, respectively. The error of predicted and actual observations in the prediction training data may be estimated and fed back into model training (Ma & Mao, 2019). Stochastic gradient descent is implemented to optimise parameter learning. Assuming that the real data at time t is r, the loss function is given in the formula (20).

$$L(r,\hat{r}) = \frac{1}{2}\|r-\hat{r}\|_2^2 \tag{21}$$

The number of filters is the number of neurons, since each neuron performs a different convolution on the input to the layer. It can also be distributed as multiples of 32, with a range limit of 32–512. The size of the filter defines how many neighboring data points there are in a convolutional layer. The most used sizes in this work have been 3×3 and 5×5. Stride and padding are a parameters of the neural network's filter

that modifies the amount of movement over observations. In this work and usually, a stride size no greater than 2×2 and a padding no greater than 1×1 have been used. Finally, we provide a flowchar of the steps to complete in order to run this DRCNN in the Fig. 2.

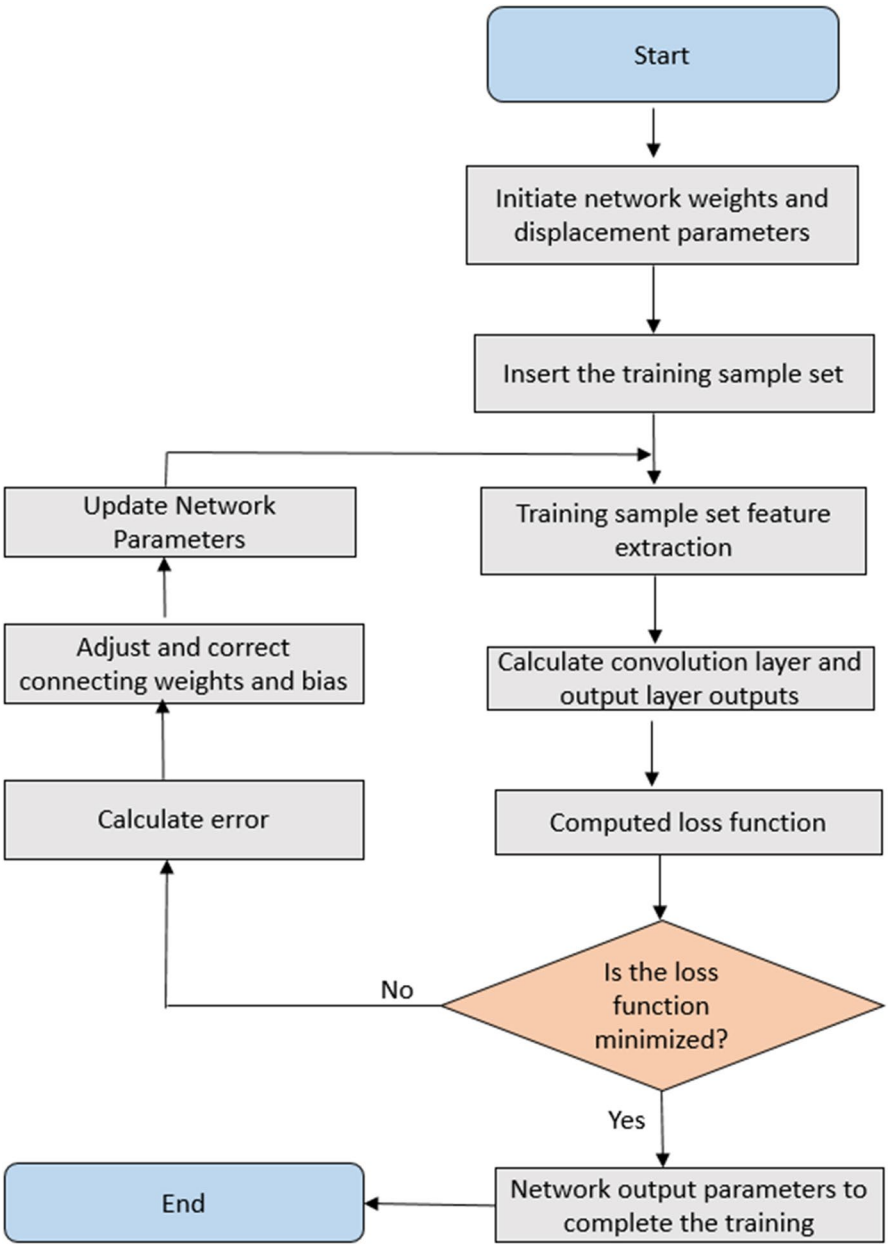

Fig. 2 Deep recurrent convolutional neural network flowchart

3 Sample and Data

We employ bond prices for a one-year bond market in the period from February 15th, 2000 to April 12th, 2023. The sample consists of ten sovereign bonds in five advanced economies (Germany, United States, Italy, Spain, and Japan) and five emerging countries (Turkey, Mexico, Indonesia, Nigeria, and Poland); ten corporate bonds in five advanced economies (Walmart, Johnson & Johnson, Verizon, Unilever PLC, Rito Tinto PLC) and in six emerging economies (Air Liquid, Ambev, Cemex, Turkish Airlines, KCE Electronics, Telekomunikacja Polska), and finally, ten high-yield bonds in five developed countries (Caesars Resort Collection LLC, Asurion LLC, Intelsat Jackson Holdings, Athenahealth Group, Great Outdoors Group LLC) and in five developing markets (Petroleos Mexicanos, Petrobas, Sands China Ltd, Indonesia Asahan Alumini, Longfor Properties). "Appendix 3" displays a detailed information about the features of every bond used in the sample. We have got data on the bond prices from the Eikon database from Refinitiv. The data on trades in Refinitiv comprises information on executed trades such as price and volume, which are timestamped up to the microsecond with tools like Refinitiv Tick History. On the other hand, the information on order book includes the limit price and order volume for both the bid and ask sides, covering limits one to ten. This information has been used by recent studies from Clapham et al. (2022), Hansen and Borch (2022), and Dodd et al. (2023). Table 1 summarizes the sample according to every category of the fixed-income market used.

We categorize all trades that happen in the continuous session across the day as "continuous trades" and build "all trades" aggregating the trades performed in the open and close sessions to the "continuous trades". To avoid dealing asynchronously, we display our data at 10, 30, and 60 min.

In addition, we measure the cost-effectiveness of our selected forecasting techniques by the following ratios.

3.1 Sign Prediction Ratio (SPR)

Correctly predicted price direction change is assigned 1, and -1 otherwise. This ratio is defined as:

$$SPR = \frac{\sum_{j=1+M/2}^{M} matches\left(Y_j, Y_j'\right)}{M/2} \tag{22}$$

being "matches" the following

$$matches\left(Y_j, Y_j\prime\right) = \begin{cases} 1 & if\ sign\left(Y_j\right) = sign\left(Y_j\prime\right) \\ 0 & otherwise \end{cases} \tag{23}$$

being the "sign function" that assigns $+1$ for positive arguments and -1 for negative arguments.

With the purpose of correcting the possible deficiencies of the model in terms of its precision regarding the direction of the trend of the movements in the prices

Table 1 Sample of bonds used

Sovereign bonds	Advanced economies	Germany
		United States
		Italy
		Spain
		Japan
	Emerging economies	Turkey
		Mexico
		Indonesia
		Nigeria
		Poland
Corporate bonds	Advanced economies	Walmart
		Johnson & Johnson
		Verizon
		Unilever PLC
		Rio Tinto PLC
	Emerging economies	Air Liquide Ambev
		Cemex
		Turkish Airlines
		KCE Electronics
		Telekomunikacja Polska
High-yield bonds	Advanced economies	Caesars Resort Collection LLC
		Asurion LLC
		Intelsat Jackson holdings
		Athenahealth group
		Great outdoors group LLC
	Emerging economies	Petróleos Mexicanos
		Petrobras
		Sands China Ltd
		Indonesia Asahan Alumini
		Longfor Properties

of the securities, we have incorporated a modification of the previous equation by adding the Moving Average Convergence Divergence (MACD) model. The MACD model is commonly calculated using the following equation:

$$\text{MACD Line} = \text{12-day Exponential Moving Average (EMA)} - \text{26-day EMA} \tag{24}$$

The MACD line represents the difference between the 12-day EMA and the 26-day EMA. The EMA is a type of moving average that gives more weight to recent data points. By subtracting the longer-term EMA from the shorter-term EMA, the MACD line aims to capture the momentum and trend direction of the underlying asset (Chong & Ng, 2008; Ramlall, 2016; Sezer & Ozbayoglu, 2018).The approach

of using MACD as a correction factor in the SPR can be modeled using the following equation:

$$\text{SPR}_{\text{Adjusted}} = \text{SPR} + (1 - \text{SPR}) * (1 - \text{MACD}) \tag{25}$$

where SPR is the original sign prediction ratio, MACD is the signal generated by the MACD model, and 1-MACD is used as a correction factor. The term (1-SPR) represents the complement of the original SPR, reflecting the portion of the original SPR that is not considered accurate. The term (1-MACD) represents the complement of the MACD signal, reflecting the extent to which the MACD signal indicates a potential reversal or correction in the market. The intuition behind this equation is that when the MACD signal is positive, it is likely that the market is trending upwards and the original SPR is more accurate. However, when the MACD signal is negative, it suggests a market reversal or a correction, and the original SPR may not be as accurate. In this case, the correction factor is used to adjust the SPR downward to reflect the possibility of a trend reversal (de Almeida & Neves, 2022; Ramlall, 2016; Slade, 2017).

By adding the correction factor to the original SPR, the adjusted $\text{SPR}_{\text{Adjusted}}$ takes into account the possibility of trend reversals or corrections indicated by the MACD signal. When the MACD signal is positive, the original SPR is considered more accurate and is only slightly adjusted. However, when the MACD signal is negative, indicating a potential trend reversal, the original SPR is adjusted more significantly downward to reflect the increased likelihood of a reversal.

This approach aims to combine the predictive power of the original SPR with the insights provided by the MACD signal, adjusting the SPR to account for potential trend changes. It recognizes that the MACD signal can act as a corrective factor when the market conditions indicate a higher likelihood of a trend reversal or correction.

3.2 Ideal Profit Ratio (IPR)

Is the ratio between the total Return and the maximum return.

$$IPR = \frac{Total\ \text{Return}}{Maximum\ \text{Return}} \tag{26}$$

The Total Return is computed in the following formula, where "sign" denotes the "sign function" and the better the forecasting approach, the higher the total return will be.

$$Total\ Return = \sum_{j=1+M/2}^{M} sign\left(Y_j'\right) * Y_j \tag{27}$$

The maximum return is determined by summing all absolute expected figures and reflects the maximum achievable return, considering a perfectly foreseeable forecast. This ratio is defined as:

$$Maximum\ Return = \sum_{j=1+M/2}^{M} abs\left(Y_j\right) \tag{28}$$

Nelson-Siegel model, which was introduced by economists Nelson and Siegel in 1987. The model is based on the idea that the yield curve can be decomposed into three factors: the level factor, the slope factor, and the curvature factor. These factors capture the average level of interest rates, the steepness of the yield curve, and the degree of curvature, respectively. The model can be expressed mathematically as follows:

$$r(t) = \beta 1 + \beta 2 * \left[1 - \exp(-t/\tau)\right]/(t/\tau) + \beta 3 * \left[(1 - \exp(-t/\tau))/(t/\tau) - \exp(-t/\tau)\right] \tag{29}$$

where r(t) represents the yield on a bond with time to maturity t, and β1, β2, β3, and τ are parameters to be estimated. The parameter β1 represents the long-term mean level of interest rates, β2 represents the slope of the yield curve at short maturities, β3 represents the curvature of the yield curve, and τ represents the time scale over which the yield curve adjusts to its long-term mean.

The rolling regression method involves estimating the relationship between the excess returns of the bond portfolio and changes in the yield curve over a specified rolling time period, such as one month or one quarter. The slope of the regression line represents the expected excess return of the portfolio for a given change in the yield curve (Grinold & Ronald, 1999; Ibbotson & Kaplan, 2000).

The equation for the rolling regression model can be written as follows:

$$\text{Excess Return} = \alpha + \beta * \text{Yield Curve Change} + \varepsilon \tag{30}$$

where Excess Return is the excess return of the bond over the risk-free rate, typically estimated using a 3-month U.S. Treasury bond as the benchmark (Campbell et al., 2001). Yield Curve Change is the change in the yield curve over the rolling time period, calculating the yields for each maturity point based on the Nelson-Siegel model for both yield curves. The term α is the intercept of the regression line, which represents the expected excess return of the bond when the yield curve change is zero. The term β is the slope of the regression line, which represents the expected excess return of the bond for a one-unit change in the yield curve. The term ε is the residual error term, which represents the deviation of the actual excess return from the predicted excess.

For its part, a modification of the Ideal Profit Ratio equation has been made following what has been done in works such as Elton et al. (1995) y Grinold and Ronald (1999). The ideal profit ratio is a measure of the performance of an investment strategy relative to a benchmark. It is calculated as the difference between the total returns of the strategy and the maximum returns of the benchmark, divided by the maximum returns of the benchmark.

To incorporate the excess return based on the yield curve into the calculation of the ideal profit ratio, you could modify the equation as follows:

$$\text{Ideal Profit Ratio} = (\text{Total Return} - \text{Expected Return})/\text{Maximum Return} \tag{31}$$

This modified equation measures the performance of the investment strategy relative to the benchmark, while taking into account the impact of the yield curve on the expected return of the portfolio. A higher ideal profit ratio indicates better performance relative to the benchmark.

Finally, after calculating the aforementioned equation of the Ideal Profit Ratio, its final result will be the net value after applying the transaction cost. In our case we used the difference between the average customer buy and the average customer sell price on each day to quantify transaction costs according to the specification of Hong and Warga (2000) and Chakravarty and Sarkar (2003):

$$TC_{AvgBidAsk} = \frac{\overline{P_t^{buy}} - \overline{P_t^{sell}}}{0.5 - \left(\overline{P_t^{buy}} - \overline{P_t^{sell}}\right)} \tag{32}$$

where $\overline{P_t^{buy/sell}}$ t is the average price of all customers buy/sell trades on day t. We calculate $TC_{AvgBidAsk}$ for each day on which there is at least one buy and one sell trade and use the monthly mean as a monthly transaction cost measure following the specifacions of previous works (Schestag et al., 2016).

4 Results

From our data described in the previous section, we collect a sample at 10, 30- and 60-min intervals, and afterward, we implement ten different methods defined in Sect. 2. The size of the training sample for the whole daily forecasting time horizon appears as 50% of the total sample size approximated to the nearest integer value, while the other 50% is used as an out-of-sample data set.

We introduce two key measures, defined in Sect. 3, of the performance of the methodologies. First, is the sign prediction rate, representing the proportion of times that the corresponding methodology accurately estimates the direction of the future price (up or down). Since correctly guessing the future price change would not ensure better results, we should contrast the performance of different prediction methodologies with a correct prediction of price changes. Thus, the ideal profit ratio is the relationship between the profitability generated by a particular method and a perfect sign forecast.

We implement the process mentioned above for "continuous operations" in the sample period and, in addition, we also apply it for "all operations" to test for robustness. Tables 2, 3, 4, 5, 6, and 7 display the results achieved for each bond at different time scales and for "continuous trades". The results for the "all trades" case scenario are presented in "Appendix 2" via Tables 10, 11, 12, 13, 14, 15.

Table 2 reports the sign prediction accuracy ratios of continuous trading for the ten techniques on the considered bonds for 10 min. We remark that QGA performs the best with 31 bonds, with an accuracy rate of over 0.772 and a mean of 0.881. DRCNN and DLNN-GA and are the second and third methods that correctly predict the change in bond price direction, with an average of 0.850 and 0.847 respectively.

Table 2 Sign Prediction Ratio (10 min) for continuous trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.837	0.876	0.894	0.906	0.929	0.878	0.912	0.873	0.885	0.892
United States	0.828	0.875	0.888	0.903	0.925	0.876	0.909	0.867	0.878	0.885
Italy	0.820	0.871	0.882	0.902	0.923	0.869	0.904	0.861	0.878	0.882
Spain	0.817	0.863	0.877	0.894	0.917	0.863	0.903	0.858	0.870	0.876
Japan	0.814	0.862	0.874	0.887	0.913	0.859	0.899	0.849	0.870	0.870
Turkey	0.813	0.861	0.868	0.881	0.904	0.854	0.897	0.845	0.862	0.866
Mexico	0.809	0.856	0.863	0.881	0.900	0.850	0.895	0.837	0.860	0.861
Indonesia	0.806	0.850	0.862	0.875	0.893	0.842	0.889	0.833	0.855	0.854
Nigeria	0.805	0.843	0.854	0.870	0.888	0.838	0.888	0.829	0.853	0.847
Poland	0.804	0.838	0.847	0.863	0.883	0.835	0.881	0.829	0.851	0.846
Walmart	0.808	0.842	0.852	0.868	0.888	0.839	0.885	0.833	0.856	0.850
Johnson & Johnson	0.798	0.832	0.841	0.857	0.877	0.828	0.874	0.823	0.845	0.840
Verizon	0.790	0.828	0.852	0.865	0.892	0.835	0.881	0.834	0.832	0.836
Unilever PLC	0.802	0.830	0.864	0.872	0.900	0.848	0.883	0.836	0.835	0.842
Rio Tinto PLC	0.792	0.820	0.854	0.861	0.889	0.837	0.872	0.826	0.825	0.831
Air Liquide	0.819	0.844	0.889	0.888	0.932	0.864	0.901	0.846	0.850	0.863
Ambev	0.831	0.845	0.895	0.902	0.942	0.873	0.908	0.852	0.854	0.855
Cemex	0.824	0.827	0.872	0.887	0.935	0.871	0.902	0.830	0.826	0.848
Turkish Airlines	0.806	0.816	0.853	0.879	0.914	0.852	0.889	0.811	0.814	0.822
KCE Electronics	0.777	0.798	0.832	0.863	0.894	0.837	0.866	0.787	0.796	0.799
Telekomunikacja Polska	0.761	0.777	0.822	0.856	0.882	0.821	0.846	0.773	0.777	0.796
Caesars Resort Collection LLC	0.761	0.757	0.822	0.845	0.871	0.807	0.836	0.748	0.764	0.793
Asurion LLC	0.732	0.740	0.821	0.829	0.868	0.784	0.808	0.742	0.749	0.771
Intelsat Jackson Holdings	0.711	0.720	0.812	0.803	0.867	0.772	0.800	0.739	0.727	0.782
Athenahealth Group	0.706	0.710	0.798	0.798	0.863	0.744	0.773	0.739	0.698	0.781
Great Outdoors Group llc	0.681	0.689	0.784	0.781	0.847	0.742	0.769	0.722	0.694	0.778

Table 2 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Petróleos Mexicanos	0.677	0.684	0.782	0.777	0.821	0.737	0.765	0.711	0.692	0.755
Petrobras	0.669	0.660	0.760	0.765	0.805	0.722	0.744	0.691	0.675	0.741
Sands China Ltd	0.662	0.641	0.739	0.748	0.792	0.697	0.741	0.662	0.651	0.740
Indonesia Asahan Alumini	0.658	0.629	0.736	0.721	0.774	0.681	0.722	0.646	0.647	0.728
Longfor Properties	0.640	0.627	0.715	0.726	0.772	0.659	0.715	0.621	0.629	0.706

Table 3 Ideal profit ratio (10 min) for continuous trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.0099	0.0127	0.0106	0.0133	0.0167	0.0138	0.0124	0.0095	0.0112	0.0138
United States	0.0119	0.0136	0.0112	0.0137	0.0178	0.0140	0.0125	0.0110	0.0122	0.0139
Italy	0.0140	0.0149	0.0116	0.0137	0.0180	0.0146	0.0127	0.0118	0.0133	0.0149
Spain	0.0150	0.0152	0.0121	0.0146	0.0194	0.0158	0.0138	0.0131	0.0138	0.0152
Japan	0.0165	0.0167	0.0128	0.0150	0.0205	0.0171	0.0150	0.0140	0.0146	0.0155
Turkey	0.0186	0.0186	0.0134	0.0158	0.0212	0.0173	0.0158	0.0154	0.0154	0.0168
Mexico	0.0168	0.0172	0.0123	0.0151	0.0200	0.0169	0.0143	0.0150	0.0153	0.0164
Indonesia	0.0159	0.0152	0.0116	0.0145	0.0200	0.0169	0.0141	0.0138	0.0147	0.0157
Nigeria	0.0157	0.0142	0.0109	0.0143	0.0189	0.0158	0.0135	0.0131	0.0145	0.0146
Poland	0.0146	0.0122	0.0095	0.0130	0.0181	0.0146	0.0128	0.0126	0.0139	0.0145
Walmart	0.0137	0.0119	0.0090	0.0116	0.0171	0.0138	0.0113	0.0114	0.0126	0.0145
Johnson & Johnson	0.0119	0.0104	0.0076	0.0114	0.0170	0.0126	0.0111	0.0102	0.0112	0.0133
Verizon	0.0110	0.0089	0.0062	0.0103	0.0158	0.0119	0.0111	0.0091	0.0104	0.0126
Unilever PLC	0.0089	0.0084	0.0054	0.0103	0.0154	0.0107	0.0103	0.0084	0.0101	0.0123
Rio Tinto PLC	0.0075	0.0082	0.0046	0.0089	0.0150	0.0097	0.0097	0.0073	0.0090	0.0119
Air Liquide	0.0081	0.0100	0.0048	0.0101	0.0156	0.0107	0.0101	0.0080	0.0093	0.0131
Ambev	0.0088	0.0111	0.0060	0.0108	0.0162	0.0117	0.0112	0.0083	0.0094	0.0138
Cemex	0.0107	0.0114	0.0068	0.0120	0.0165	0.0132	0.0118	0.0087	0.0101	0.0140
Turkish Airlines	0.0121	0.0116	0.0075	0.0133	0.0173	0.0133	0.0119	0.0101	0.0106	0.0150
KCE Electronics	0.0139	0.0120	0.0085	0.0145	0.0185	0.0137	0.0127	0.0109	0.0119	0.0155
Telekomunikacja Polska	0.0143	0.0130	0.0097	0.0158	0.0192	0.0148	0.0133	0.0113	0.0120	0.0158
Caesars Resort Collection LLC	0.0150	0.0139	0.0104	0.0160	0.0192	0.0162	0.0134	0.0122	0.0134	0.0168
Asurion LLC	0.0141	0.0121	0.0099	0.0151	0.0187	0.0147	0.0129	0.0113	0.0129	0.0165
Intelsat Jackson Holdings	0.0157	0.0100	0.0088	0.0142	0.0186	0.0143	0.0124	0.0100	0.0120	0.0155
Athenahealth Group	0.0145	0.0081	0.0082	0.0129	0.0186	0.0129	0.0123	0.0090	0.0111	0.0147
Great Outdoors Group llc	0.0144	0.0062	0.0073	0.0128	0.0179	0.0129	0.0117	0.0077	0.0109	0.0132

Table 3 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Petróleos Mexicanos	0.0136	0.0062	0.0072	0.0119	0.0166	0.0124	0.0114	0.0072	0.0101	0.0119
Petrobras	0.0122	0.0042	0.0063	0.0116	0.0163	0.0115	0.0102	0.0065	0.0098	0.0114
Sands China Ltd	0.0104	0.0035	0.0056	0.0115	0.0151	0.0105	0.0093	0.0053	0.0084	0.0103
Indonesia Asahan Alumini	0.0098	0.0026	0.0044	0.0106	0.0142	0.0094	0.0086	0.0041	0.0084	0.0101
Longfor Properties	0.0095	0.0005	0.0031	0.0094	0.0139	0.0088	0.0085	0.0028	0.0074	0.0096

Table 4 Sign Prediction Ratio (30 min) for continuous trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.816	0.855	0.872	0.897	0.906	0.857	0.890	0.851	0.864	0.878
United States	0.808	0.853	0.866	0.895	0.902	0.854	0.887	0.845	0.857	0.876
Italy	0.800	0.850	0.860	0.893	0.900	0.848	0.882	0.840	0.856	0.873
Spain	0.797	0.842	0.855	0.886	0.894	0.841	0.881	0.837	0.849	0.867
Japan	0.794	0.840	0.853	0.878	0.890	0.838	0.877	0.828	0.848	0.860
Turkey	0.793	0.839	0.847	0.872	0.882	0.833	0.874	0.824	0.841	0.856
Mexico	0.789	0.834	0.841	0.872	0.878	0.829	0.873	0.817	0.838	0.852
Indonesia	0.786	0.829	0.841	0.867	0.871	0.822	0.867	0.812	0.834	0.844
Nigeria	0.785	0.822	0.833	0.862	0.866	0.817	0.866	0.809	0.832	0.838
Poland	0.784	0.817	0.826	0.855	0.862	0.814	0.859	0.808	0.830	0.837
Walmart	0.776	0.814	0.823	0.849	0.861	0.812	0.857	0.808	0.823	0.832
Johnson & Johnson	0.768	0.802	0.815	0.840	0.859	0.804	0.844	0.804	0.811	0.822
Verizon	0.771	0.807	0.830	0.856	0.870	0.814	0.859	0.813	0.811	0.826
Unilever PLC	0.782	0.810	0.843	0.864	0.878	0.827	0.861	0.816	0.814	0.832
Rio Tinto PLC	0.790	0.814	0.853	0.873	0.894	0.835	0.864	0.820	0.818	0.843
Air Liquide	0.798	0.823	0.867	0.880	0.909	0.842	0.878	0.825	0.829	0.854
Ambev	0.810	0.824	0.873	0.893	0.918	0.851	0.886	0.831	0.832	0.846
Cemex	0.804	0.807	0.850	0.878	0.912	0.849	0.880	0.810	0.805	0.838
Turkish Airlines	0.786	0.796	0.832	0.871	0.891	0.831	0.867	0.791	0.794	0.813
KCE Electronics	0.757	0.778	0.811	0.855	0.872	0.816	0.844	0.768	0.777	0.790
Telekomunikacja Polska	0.742	0.757	0.802	0.848	0.860	0.801	0.825	0.754	0.758	0.787
Caesars Resort Collection LLC	0.742	0.738	0.802	0.837	0.849	0.787	0.815	0.730	0.745	0.784
Asurion LLC	0.714	0.721	0.800	0.821	0.847	0.764	0.788	0.724	0.731	0.762
Intelsat Jackson Holdings	0.693	0.702	0.791	0.795	0.846	0.753	0.780	0.721	0.709	0.753

Table 4 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.689	0.692	0.778	0.790	0.842	0.725	0.754	0.721	0.681	0.753
Great Outdoors Group llc	0.664	0.672	0.764	0.774	0.826	0.724	0.750	0.704	0.676	0.750
Petróleos Mexicanos	0.661	0.667	0.762	0.769	0.801	0.718	0.746	0.693	0.675	0.728
Petrobras	0.653	0.644	0.741	0.757	0.785	0.704	0.726	0.674	0.658	0.714
Sands China Ltd	0.645	0.625	0.721	0.740	0.773	0.680	0.723	0.646	0.635	0.713
Indonesia Asahan Alumini	0.642	0.613	0.718	0.719	0.755	0.664	0.704	0.630	0.631	0.701
Longfor Properties	0.624	0.612	0.697	0.719	0.753	0.643	0.697	0.605	0.613	0.680

Table 5 Ideal profit ratio (30 min) for continuous trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.0074	0.0091	0.0137	0.0158	0.0182	0.0146	0.0120	0.0104	0.0101	0.0136
United States	0.0080	0.0093	0.0149	0.0158	0.0207	0.0153	0.0138	0.0109	0.0115	0.0157
Italy	0.0085	0.0096	0.0156	0.0159	0.0216	0.0158	0.0143	0.0120	0.0120	0.0164
Spain	0.0097	0.0117	0.0174	0.0160	0.0221	0.0174	0.0163	0.0130	0.0137	0.0182
Japan	0.0110	0.0119	0.0185	0.0168	0.0238	0.0190	0.0184	0.0144	0.0138	0.0195
Turkey	0.0132	0.0124	0.0191	0.0189	0.0252	0.0200	0.0201	0.0165	0.0148	0.0196
Mexico	0.0119	0.0118	0.0175	0.0181	0.0235	0.0188	0.0185	0.0155	0.0133	0.0176
Indonesia	0.0100	0.0096	0.0169	0.0160	0.0229	0.0181	0.0166	0.0154	0.0117	0.0170
Nigeria	0.0097	0.0085	0.0164	0.0143	0.0217	0.0162	0.0152	0.0153	0.0097	0.0163
Poland	0.0080	0.0077	0.0143	0.0129	0.0222	0.0157	0.0135	0.0132	0.0079	0.0150
Walmart	0.0072	0.0075	0.0133	0.0127	0.0208	0.0144	0.0117	0.0111	0.0059	0.0137
Johnson & Johnson	0.0063	0.0069	0.0116	0.0112	0.0201	0.0135	0.0108	0.0105	0.0048	0.0122
Verizon	0.0047	0.0053	0.0112	0.0109	0.0181	0.0135	0.0092	0.0096	0.0037	0.0112
Unilever PLC	0.0030	0.0039	0.0094	0.0097	0.0172	0.0127	0.0075	0.0077	0.0019	0.0101
Rio Tinto PLC	0.0024	0.0035	0.0077	0.0094	0.0171	0.0123	0.0067	0.0058	0.0002	0.0098
Air Liquide	0.0036	0.0055	0.0082	0.0112	0.0173	0.0127	0.0079	0.0058	0.0015	0.0104
Ambev	0.0056	0.0066	0.0085	0.0131	0.0183	0.0130	0.0079	0.0068	0.0027	0.0119
Cemex	0.0078	0.0070	0.0102	0.0144	0.0190	0.0135	0.0080	0.0072	0.0042	0.0122
Turkish Airlines	0.0079	0.0078	0.0111	0.0146	0.0197	0.0144	0.0094	0.0085	0.0044	0.0128
KCE Electronics	0.0101	0.0093	0.0128	0.0148	0.0208	0.0158	0.0104	0.0093	0.0061	0.0131
Telekomunikacja Polska	0.0121	0.0108	0.0132	0.0159	0.0217	0.0178	0.0112	0.0100	0.0073	0.0137
Caesars Resort Collection LLC	0.0136	0.0123	0.0146	0.0176	0.0227	0.0186	0.0129	0.0108	0.0079	0.0137
Asurion LLC	0.0119	0.0111	0.0133	0.0171	0.0225	0.0170	0.0123	0.0103	0.0076	0.0116
Intelsat Jackson Holdings	0.0136	0.0098	0.0117	0.0153	0.0206	0.0152	0.0108	0.0101	0.0070	0.0112

Table 5 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.0129	0.0097	0.0114	0.0142	0.0190	0.0152	0.0104	0.0089	0.0050	0.0092
Great Outdoors Group llc	0.0121	0.0081	0.0106	0.0126	0.0170	0.0131	0.0086	0.0072	0.0031	0.0084
Petróleos Mexicanos	0.0109	0.0071	0.0092	0.0120	0.0158	0.0118	0.0078	0.0051	0.0025	0.0070
Petrobras	0.0102	0.0053	0.0077	0.0111	0.0138	0.0117	0.0072	0.0034	0.0014	0.0066
Sands China Ltd	0.0082	0.0036	0.0056	0.0104	0.0127	0.0101	0.0069	0.0016	0.0005	0.0057
Indonesia Asahan Alumini	0.0070	0.0025	0.0054	0.0098	0.0112	0.0084	0.0060	0.0015	0.0004	0.0056
Longfor Properties	0.0059	0.0023	0.0044	0.0078	0.0098	0.0069	0.0039	0.0006	0.0000	0.0048

Table 6 Sign Prediction Ratio (60 min) for continuous trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.796	0.834	0.850	0.875	0.883	0.835	0.883	0.830	0.842	0.879
United States	0.788	0.832	0.845	0.872	0.880	0.833	0.880	0.824	0.836	0.877
Italy	0.780	0.829	0.839	0.871	0.878	0.827	0.875	0.819	0.835	0.874
Spain	0.777	0.821	0.834	0.864	0.872	0.821	0.874	0.816	0.828	0.868
Japan	0.775	0.820	0.831	0.857	0.868	0.817	0.870	0.808	0.827	0.862
Turkey	0.774	0.819	0.826	0.851	0.860	0.812	0.868	0.804	0.820	0.858
Mexico	0.770	0.814	0.821	0.851	0.856	0.809	0.866	0.797	0.818	0.853
Indonesia	0.766	0.809	0.820	0.845	0.850	0.801	0.860	0.792	0.813	0.846
Nigeria	0.766	0.802	0.813	0.841	0.844	0.797	0.860	0.789	0.811	0.839
Poland	0.764	0.797	0.806	0.834	0.840	0.794	0.852	0.788	0.810	0.838
Walmart	0.757	0.794	0.803	0.828	0.840	0.791	0.851	0.788	0.802	0.833
Johnson & Johnson	0.749	0.783	0.795	0.819	0.838	0.784	0.837	0.784	0.791	0.823
Verizon	0.751	0.787	0.810	0.835	0.849	0.794	0.853	0.793	0.791	0.828
Unilever PLC	0.763	0.790	0.822	0.842	0.856	0.806	0.855	0.796	0.794	0.834
Rio Tinto PLC	0.771	0.794	0.832	0.851	0.871	0.815	0.858	0.799	0.798	0.844
Air Liquide	0.779	0.802	0.845	0.858	0.886	0.822	0.872	0.804	0.808	0.856
Ambev	0.790	0.803	0.851	0.871	0.896	0.830	0.879	0.810	0.812	0.847
Cemex	0.784	0.787	0.829	0.856	0.889	0.828	0.873	0.790	0.785	0.840
Turkish Airlines	0.766	0.776	0.811	0.849	0.869	0.810	0.860	0.771	0.774	0.814
KCE Electronics	0.739	0.759	0.791	0.834	0.850	0.796	0.838	0.749	0.757	0.791
Telekomunikacja Polska	0.724	0.739	0.782	0.827	0.839	0.781	0.819	0.735	0.739	0.789
Caesars Resort Collection LLC	0.723	0.720	0.782	0.816	0.828	0.767	0.809	0.712	0.726	0.786
Asurion LLC	0.696	0.704	0.780	0.801	0.826	0.745	0.782	0.706	0.713	0.764
Intelsat Jackson Holdings	0.676	0.685	0.772	0.775	0.825	0.735	0.774	0.703	0.691	0.744

Table 6 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.672	0.675	0.759	0.770	0.821	0.707	0.748	0.703	0.664	0.743
Great Outdoors Group llc	0.648	0.656	0.746	0.755	0.806	0.706	0.744	0.687	0.660	0.740
Petróleos Mexicanos	0.644	0.651	0.743	0.750	0.781	0.701	0.741	0.676	0.658	0.718
Petrobras	0.637	0.628	0.723	0.738	0.765	0.687	0.720	0.657	0.642	0.705
Sands China Ltd	0.629	0.610	0.703	0.722	0.754	0.663	0.717	0.630	0.619	0.704
Indonesia Asahan Alumini	0.626	0.598	0.700	0.701	0.737	0.648	0.699	0.615	0.615	0.693
Longfor Properties	0.608	0.597	0.680	0.701	0.735	0.627	0.692	0.590	0.598	0.672

Table 7 Ideal profit ratio (60 min) for continuous trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.0087	0.0111	0.0121	0.0174	0.0182	0.0153	0.0133	0.0077	0.0032	0.0095
United States	0.0101	0.0131	0.0139	0.0177	0.0187	0.0155	0.0137	0.0087	0.0044	0.0100
Italy	0.0112	0.0148	0.0147	0.0195	0.0202	0.0169	0.0150	0.0095	0.0048	0.0107
Spain	0.0121	0.0150	0.0153	0.0204	0.0211	0.0185	0.0162	0.0114	0.0059	0.0119
Japan	0.0125	0.0159	0.0153	0.0212	0.0213	0.0187	0.0174	0.0117	0.0070	0.0139
Turkey	0.0140	0.0172	0.0172	0.0227	0.0217	0.0206	0.0185	0.0125	0.0090	0.0150
Mexico	0.0138	0.0166	0.0169	0.0207	0.0214	0.0186	0.0164	0.0116	0.0083	0.0129
Indonesia	0.0137	0.0155	0.0164	0.0187	0.0193	0.0174	0.0155	0.0100	0.0064	0.0122
Nigeria	0.0118	0.0140	0.0162	0.0180	0.0188	0.0168	0.0150	0.0078	0.0057	0.0107
Poland	0.0111	0.0137	0.0154	0.0161	0.0174	0.0167	0.0149	0.0068	0.0035	0.0103
Walmart	0.0109	0.0129	0.0143	0.0156	0.0158	0.0161	0.0129	0.0065	0.0025	0.0089
Johnson & Johnson	0.0092	0.0127	0.0124	0.0145	0.0156	0.0160	0.0121	0.0051	0.0010	0.0085
Verizon	0.0075	0.0117	0.0121	0.0123	0.0154	0.0158	0.0105	0.0030	0.0007	0.0074
Unilever PLC	0.0071	0.0104	0.0113	0.0103	0.0150	0.0140	0.0085	0.0019	0.0010	0.0072
Rio Tinto PLC	0.0066	0.0086	0.0102	0.0087	0.0142	0.0118	0.0083	0.0001	0.0024	0.0067
Air Liquide	0.0079	0.0105	0.0119	0.0093	0.0147	0.0132	0.0094	0.0001	0.0011	0.0088
Ambev	0.0088	0.0105	0.0137	0.0098	0.0164	0.0136	0.0115	0.0002	0.0002	0.0090
Cemex	0.0104	0.0121	0.0138	0.0107	0.0174	0.0141	0.0117	0.0017	0.0005	0.0090
Turkish Airlines	0.0115	0.0133	0.0147	0.0107	0.0178	0.0150	0.0124	0.0024	0.0012	0.0102
KCE Electronics	0.0124	0.0148	0.0163	0.0108	0.0197	0.0165	0.0143	0.0043	0.0028	0.0114
Telekomunikacja Polska	0.0138	0.0161	0.0172	0.0116	0.0214	0.0181	0.0157	0.0046	0.0037	0.0121
Caesars Resort Collection LLC	0.0150	0.0168	0.0190	0.0131	0.0235	0.0190	0.0173	0.0052	0.0048	0.0124
Asurion LLC	0.0139	0.0148	0.0188	0.0122	0.0218	0.0178	0.0166	0.0034	0.0038	0.0124
Intelsat Jackson Holdings	0.0142	0.0144	0.0171	0.0117	0.0205	0.0157	0.0145	0.0022	0.0020	0.0108

Table 7 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.0123	0.0136	0.0149	0.0097	0.0204	0.0146	0.0131	0.0016	0.0018	0.0086
Great Outdoors Group llc	0.0107	0.0127	0.0143	0.0085	0.0202	0.0140	0.0130	0.0005	0.0003	0.0070
Petróleos Mexicanos	0.0091	0.0114	0.0124	0.0075	0.0187	0.0139	0.0111	0.0007	0.0004	0.0070
Petrobras	0.0074	0.0107	0.0112	0.0060	0.0166	0.0133	0.0101	0.0012	0.0014	0.0053
Sands China Ltd	0.0062	0.0091	0.0101	0.0052	0.0148	0.0130	0.0100	0.0016	0.0025	0.0038
Indonesia Asahan Alumini	0.0044	0.0085	0.0100	0.0036	0.0132	0.0117	0.0084	0.0022	0.0027	0.0025
Longfor Properties	0.0044	0.0074	0.0079	0.0018	0.0131	0.0105	0.0084	0.0037	0.0045	0.0011

SVM-GA may also be regarded as a reference model for the other machine learning algorithms, being the fourth best in the comparison. We notice that the fuzzy approach ise the worst-performing techniques, with an overall mean of 0.770 for the Qfuzzy method.

Table 3 shows the results of the ideal profit ratios for the selected bonds and for a time scale of 10 min for every methodology. It is noted that, in line with the success rates in Table 2, QGA is again the best-performing method, as all bonds have a positive ideal profit ratio with a mean value of 0.0175. Nevertheless, in contrast to the results of the accuracy rates, QRNN becomes the second-best performing method this time, as all bonds also have a positive ideal profit ratio and a mean of 0.0140. QRNN is followed by the ANFIS-QGA method with an average of 0.0134. In this case, SVM-GA and CNN-LSTM are the worst-performing models regarding profit generation for continuous 10-min trades. The maximum value of the ideal profit ratio among sovereign bonds is 0.0212 and is reached by QGA in Turkey. Among corporate bonds, Telelomunikacja Polska is the one that reaches a maximum value, being 0.0192, again in the QGA method. Finally, among high-yield bonds, Caesars Resort Collection LLC stands out with a value of 0.0192, also in the QGA method.

Table 4 presents the results concerning the success ratios of continuous operations with a frequency of 30 min. We note that, as in the case of the 10-min frequency, the QGA method is once again the most performing in terms of mean bond success ratio, with an average of 0.860. In the QGA method, among sovereign bonds, Germany has the highest ratio at 0.906. Looking at corporate bonds, also in the QGAs method, Ambev's is the highest value at 0.918, and among high-yield bonds, Caesars Resort Collection LLC ranks highest with a value of 0.849, also in the QGA method. The next methods that present a correct forecast of the future direction of bond prices are DLNN-GA and DRCNN, with a mean of 0.839 and 0.829 respectively. SVM-GA could also be accepted as a good model for the sign prediction ratio. On the other hand, we notice that, as with the 10-min frequency, the method Qfuzzy show low sign prediction capacity, with mean value of 0.750.

If we examine the results of the ideal profit ratio for continuous 30-min trades in Table 5, we observe that QGA emerges as the best model yielding the greatest profit ratio, with an average of 0.0193 with all bonds having a positive ratio. This result is following the success rates of the QGA method in Table 4. However, if we look at the techniques that worst predict the future direction of bond prices, in this case, this is not the fuzzy one but GRU-CNN and AdaBoost-GA. Both have a mean value of 0.0063 and 0.0080 respectively.

On considering a sampling frequency of 60 min, Table 6 reveals that, in line with the previous results, the QGA is better than the other methods regarding the mean sign prediction ratio, with an average of 0.838. As in the case of the 10-min and 30-min time scales, the maximum success ratio for all bonds through this QGA method is achieved for the corporate bond "Ambev". Moreover, DRCNN and DLNN-GA correctly predict all bonds with a rate above 0.701. Furthermore, we remark that, as with the 10-min frequency and the 30-min frequency, the Qfuzzy method displays a weak sign prediction ability, with mean values of 0.732.

If we analyze Table 7, it is evident that the genetic algorithms are the ones that obtain the best results for the 60-min frequency in the ideal profit ratio. QGA,

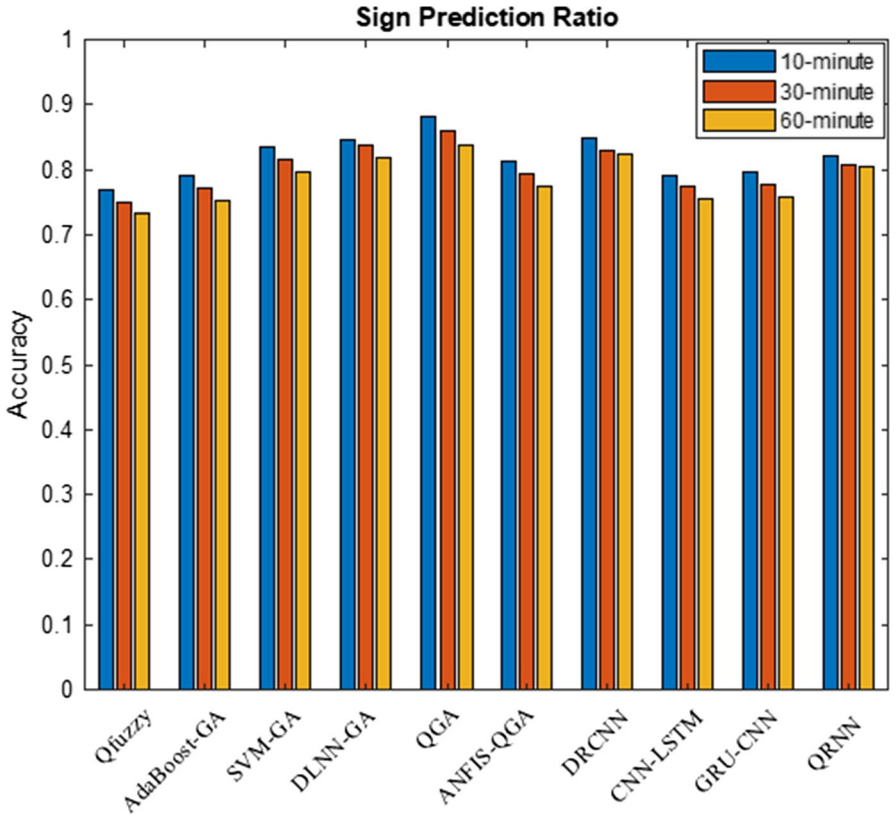

Fig. 3 Sign Prediction Ratio for continuous trades at different sampling frequencies of each method

ANFIS-QGA, and SVM-GA are those that reflect, in this order, the highest ideal profit ratio with all of the bonds having a positive ratio. DRCNN and DLNN-GA come after with an average ratio of 0.0131 and 0.0128, respectively. The lowest values, in contrast to the previous table, are those achieved in the GRU-CNN and CNN-LSTM methods, with an average value of 0.0032 and 0.0048 respectively.

When we examine continuous trading time series at 10, 30, and 60 min, we can reveal the impact of the sampling frequency on the prediction. We note in Fig. 3 that all methods show better results concerning the Sign Prediction Ratio at lower frequencies. Nevertheless, the case is otherwise, as illustrated in Fig. 4, for the Ideal Benefit Ratio, as AdaBoost-GA, SVM-GA, ANFIS-QGA and DRCNN perform the best model for trading strategy setting at 60 min of sampling, and QGAperforms the best for 30 min of sampling. Following our results, not only the bonds and methodology but also the prediction intervals are important. As a consequence, we may conclude that one method is not suitable for everything. While a method may be suitable for raw data, it may not be appropriate for fine data.

Tables 8 and 9 show the results of Sovereign bonds at 1 and 5 min frequency intervals. If we observe Table 8, the methodology with the best results is QGA

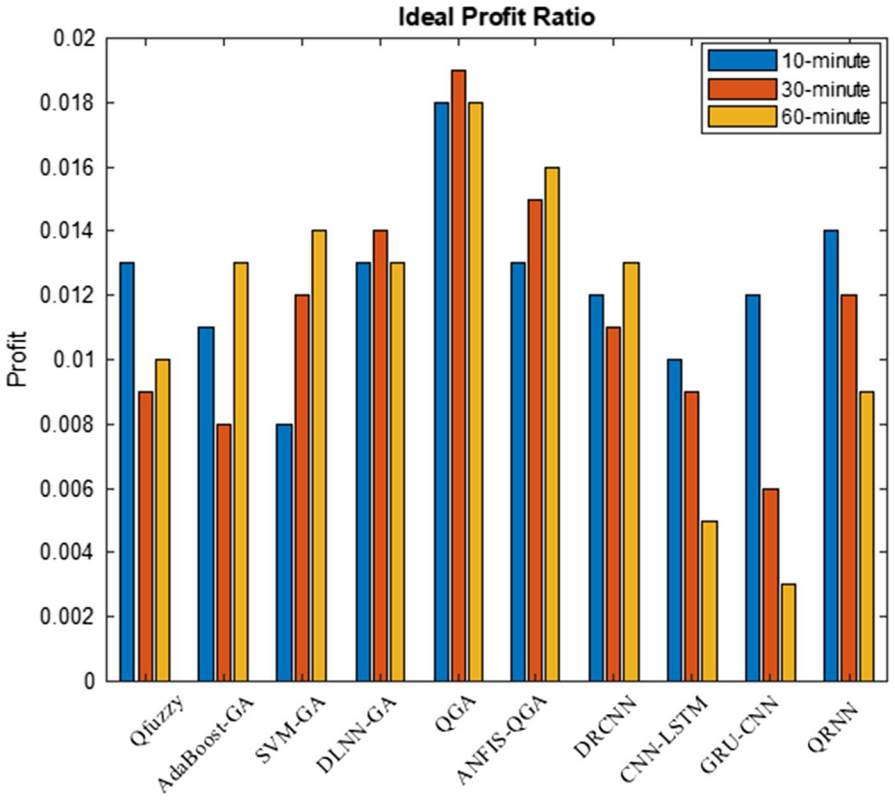

Fig. 4 Ideal profit ratio for continuous trades at different sampling frequencies of each method

for both ratios, sign prediction and ideal profit. In the case of the sign prediction ratio for a one-minute frequency, German sovereign bonds are the ones that reach the highest value (0.924) and for the 5-min frequency case, Italian sovereign bonds with a ratio of 0.946. With reference to the ideal profit ratio, the best result is obtained by Turkey for a frequency of 1 min (0.0229), and Spain for a frequency of 5 min (0.0247). For all trades, Table 9 illustrates that the best sovereign bond performance in sign prediction ratio at 1-min frequency is Japan (0.930) in the Qfuzzy method. However, at a frequency of 5 min, the DLMM-GA method performs best, with Germany having the highest ratio. Regarding the ideal profit ratio, QRNN is the best methodology, with Turkey obtaining the highest values at both frequencies, 0.0244 at a 1-min frequency and 0.0195 at a 5-min frequency.

In comparison with other works, Vukovic et al. (2020) obtain an accuracy of 82% on test cases for predicting future Sharpe ratio dynamics with neural networks. Nunes et al. (2019) achieve RMSE reductions compared to the model without synthetic data in the range of 11% to 70% (mean values, for forecast horizons of 15 and 20 days) for predicting the bond market yield curve, using the

Table 8 Sign prediction and ideal profit ratios in small frequency for continuous trades of sovereign bonds

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Sign prediction ratio (1 min)										
Germany	0.833	0.872	0.890	0.916	0.924	0.874	0.908	0.869	0.881	0.897
United States	0.824	0.871	0.884	0.913	0.920	0.872	0.905	0.862	0.874	0.895
Italy	0.817	0.867	0.878	0.911	0.919	0.865	0.900	0.857	0.874	0.892
Spain	0.813	0.859	0.872	0.904	0.912	0.859	0.899	0.854	0.866	0.886
Japan	0.810	0.858	0.870	0.896	0.908	0.855	0.895	0.845	0.866	0.879
Turkey	0.809	0.857	0.864	0.890	0.900	0.850	0.892	0.841	0.858	0.875
Mexico	0.805	0.851	0.859	0.890	0.896	0.846	0.891	0.833	0.856	0.871
Indonesia	0.802	0.846	0.858	0.884	0.889	0.838	0.885	0.829	0.851	0.863
Nigeria	0.801	0.839	0.850	0.879	0.883	0.834	0.884	0.825	0.849	0.856
Poland	0.800	0.834	0.843	0.872	0.879	0.831	0.876	0.825	0.847	0.855
Sign prediction ratio (5 min)										
Germany	0.814	0.838	0.878	0.899	0.921	0.860	0.891	0.844	0.843	0.870
United States	0.822	0.848	0.893	0.906	0.936	0.868	0.905	0.850	0.854	0.881
Italy	0.835	0.849	0.899	0.920	0.946	0.877	0.912	0.856	0.858	0.873
Spain	0.828	0.831	0.876	0.905	0.939	0.875	0.907	0.834	0.829	0.865
Japan	0.809	0.820	0.857	0.897	0.918	0.856	0.893	0.815	0.818	0.839
Turkey	0.780	0.801	0.836	0.880	0.898	0.841	0.870	0.791	0.800	0.815
Mexico	0.765	0.780	0.826	0.873	0.886	0.825	0.850	0.777	0.781	0.813
Indonesia	0.764	0.761	0.826	0.862	0.875	0.810	0.840	0.752	0.767	0.809
Nigeria	0.736	0.743	0.824	0.846	0.872	0.787	0.811	0.746	0.753	0.786
Poland	0.714	0.723	0.815	0.819	0.871	0.776	0.804	0.743	0.730	0.766
Ideal profit ratio (1 min)										
Germany	0.0092	0.0115	0.0126	0.0180	0.0189	0.0159	0.0137	0.0080	0.0034	0.0099

Table 8 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
United States	0.0107	0.0138	0.0147	0.0186	0.0197	0.0164	0.0144	0.0092	0.0046	0.0106
Italy	0.0118	0.0156	0.0155	0.0206	0.0213	0.0178	0.0158	0.0100	0.0051	0.0113
Spain	0.0128	0.0158	0.0161	0.0216	0.0222	0.0195	0.0171	0.0120	0.0062	0.0125
Japan	0.0132	0.0168	0.0162	0.0224	0.0225	0.0198	0.0183	0.0123	0.0074	0.0147
Turkey	0.0148	0.0182	0.0181	0.0239	0.0229	0.0218	0.0195	0.0132	0.0095	0.0158
Mexico	0.0145	0.0176	0.0178	0.0218	0.0226	0.0196	0.0173	0.0122	0.0088	0.0136
Indonesia	0.0145	0.0164	0.0173	0.0197	0.0204	0.0184	0.0164	0.0105	0.0068	0.0129
Nigeria	0.0124	0.0148	0.0171	0.0190	0.0199	0.0178	0.0158	0.0082	0.0060	0.0113
Poland	0.0107	0.0145	0.0162	0.0170	0.0183	0.0176	0.0157	0.0072	0.0037	0.0108
Ideal PROFIT RATIO (5 min)										
Germany	0.0076	0.0094	0.0142	0.0163	0.0209	0.0151	0.0124	0.0107	0.0105	0.0140
United States	0.0083	0.0096	0.0154	0.0163	0.0231	0.0158	0.0142	0.0112	0.0119	0.0162
Italy	0.0088	0.0099	0.0162	0.0165	0.0242	0.0163	0.0148	0.0124	0.0124	0.0169
Spain	0.0100	0.0120	0.0180	0.0165	0.0247	0.0180	0.0168	0.0134	0.0141	0.0188
Japan	0.0114	0.0123	0.0191	0.0173	0.0266	0.0196	0.0190	0.0148	0.0142	0.0202
Turkey	0.0136	0.0128	0.0197	0.0196	0.0281	0.0207	0.0207	0.0171	0.0153	0.0202
Mexico	0.0123	0.0121	0.0181	0.0187	0.0262	0.0194	0.0191	0.0160	0.0138	0.0182
Indonesia	0.0103	0.0099	0.0174	0.0166	0.0255	0.0187	0.0172	0.0159	0.0121	0.0175
Nigeria	0.0100	0.0088	0.0169	0.0148	0.0243	0.0167	0.0157	0.0158	0.0100	0.0168
Poland	0.0082	0.0080	0.0148	0.0133	0.0229	0.0162	0.0139	0.0136	0.0082	0.0155

Table 9 Sign prediction and ideal profit ratios in small frequency for all trades of sovereign bonds

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Sign prediction ratio (1 min)										
Germany	0.796	0.811	0.827	0.860	0.885	0.926	0.911	0.890	0.922	0.894
United States	0.830	0.831	0.821	0.858	0.878	0.922	0.905	0.884	0.915	0.888
Italy	0.867	0.849	0.817	0.856	0.872	0.916	0.897	0.880	0.908	0.885
Spain	0.903	0.850	0.810	0.855	0.869	0.908	0.891	0.876	0.906	0.881
Japan	0.930	0.881	0.807	0.853	0.864	0.904	0.886	0.868	0.901	0.873
Turkey	0.929	0.878	0.802	0.849	0.857	0.899	0.881	0.862	0.898	0.872
Mexico	0.927	0.861	0.800	0.847	0.856	0.893	0.878	0.857	0.892	0.871
Indonesia	0.913	0.847	0.792	0.844	0.852	0.889	0.869	0.856	0.887	0.870
Nigeria	0.912	0.834	0.788	0.843	0.847	0.888	0.867	0.855	0.884	0.867
Poland	0.912	0.833	0.782	0.835	0.841	0.882	0.863	0.848	0.882	0.861
Sign prediction ratio (5 min)										
Germany	0.815	0.847	0.872	0.913	0.898	0.877	0.909	0.866	0.902	0.913
United States	0.809	0.845	0.866	0.909	0.892	0.871	0.902	0.865	0.894	0.907
Italy	0.805	0.844	0.860	0.903	0.885	0.867	0.895	0.860	0.889	0.907
Spain	0.799	0.843	0.856	0.895	0.879	0.863	0.893	0.854	0.886	0.903
Japan	0.796	0.841	0.852	0.892	0.873	0.855	0.888	0.846	0.885	0.895
Turkey	0.791	0.837	0.845	0.886	0.869	0.849	0.885	0.844	0.884	0.890
Mexico	0.789	0.835	0.844	0.881	0.865	0.845	0.879	0.842	0.876	0.886
Indonesia	0.780	0.832	0.840	0.876	0.857	0.843	0.875	0.841	0.869	0.882
Nigeria	0.776	0.831	0.835	0.875	0.855	0.843	0.871	0.837	0.867	0.876
Poland	0.771	0.824	0.829	0.869	0.851	0.836	0.869	0.833	0.859	0.874
Ideal profit ratio (1 min)										
Germany	0.0094	0.0110	0.0137	0.0144	0.0157	0.0122	0.0074	0.0103	0.0135	0.0186

Table 9 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
United States	0.0096	0.0128	0.0141	0.0161	0.0166	0.0140	0.0090	0.0108	0.0156	0.0208
Italy	0.0104	0.0130	0.0161	0.0171	0.0173	0.0158	0.0103	0.0125	0.0172	0.0217
Spain	0.0115	0.0136	0.0168	0.0173	0.0194	0.0175	0.0120	0.0142	0.0181	0.0229
Japan	0.0123	0.0152	0.0174	0.0175	0.0196	0.0172	0.0126	0.0144	0.0179	0.0237
Turkey	0.0141	0.0159	0.0196	0.0188	0.0220	0.0185	0.0132	0.0157	0.0204	0.0244
Mexico	0.0137	0.0145	0.0193	0.0186	0.0212	0.0172	0.0124	0.0152	0.0187	0.0224
Indonesia	0.0136	0.0133	0.0186	0.0185	0.0202	0.0163	0.0117	0.0150	0.0170	0.0211
Nigeria	0.0122	0.0124	0.0184	0.0165	0.0183	0.0161	0.0106	0.0145	0.0170	0.0202
Poland	0.0115	0.0119	0.0178	0.0157	0.0164	0.0143	0.0104	0.0137	0.0160	0.0191
Ideal profit ratio (5 min)										
Germany	0.0072	0.0093	0.0095	0.0111	0.0114	0.0101	0.0076	0.0057	0.0102	0.0122
United States	0.0091	0.0101	0.0104	0.0120	0.0118	0.0114	0.0082	0.0072	0.0113	0.0141
Italy	0.0101	0.0118	0.0125	0.0126	0.0130	0.0117	0.0097	0.0075	0.0121	0.0162
Spain	0.0103	0.0123	0.0131	0.0138	0.0144	0.0121	0.0101	0.0095	0.0129	0.0180
Japan	0.0101	0.0132	0.0137	0.0142	0.0154	0.0126	0.0116	0.0112	0.0146	0.0179
Turkey	0.0119	0.0145	0.0157	0.0165	0.0165	0.0139	0.0126	0.0135	0.0160	0.0195
Mexico	0.0107	0.0129	0.0141	0.0158	0.0156	0.0125	0.0121	0.0121	0.0143	0.0192
Indonesia	0.0095	0.0113	0.0141	0.0151	0.0156	0.0107	0.0106	0.0101	0.0143	0.0178
Nigeria	0.0083	0.0100	0.0133	0.0141	0.0152	0.0089	0.0105	0.0087	0.0134	0.0174
Poland	0.0077	0.0097	0.0127	0.0126	0.0149	0.0071	0.0098	0.0069	0.0128	0.0159

Multilayer Perceptrons method. In summary, our study has high precision, and also exceeds the accuracy level of previous work, being the genetic algorithms the ones that obtain the best results, especially the QGA method. Moreover, previous literature dealing with fixed-income assets is not concerned with the use of HFT. The results of our study show that bond market transactions through HFT are executed faster and trading volume increases considerably, enhancing the liquidity of the bond market.

Finally, we analyse the cumulative net profits for each bond market (sovereign, corporate and high-yield) and according to each price window (10-min, 30-min, 60-min and 1-min, 5-min). These results are presented in "Appendix 4" via Figs. 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. Over the span of two decades under examination, all models encountered drawdowns of varying magnitudes, ranging from 5 to 15% at different points in time. On average, these drawdowns persisted for approximately 2.5 months. Instances of model underperformance became evident during periods of extreme market volatility, exemplified by the 2008 financial crisis, which witnessed model losses surpassing the 20% mark. Similarly, unexpected geopolitical events posed challenges, with losses reaching up to 18%. Our models demonstrated a propensity to falter when confronted with 'black swan' events of exceptional magnitude that surpassed historical data, as exemplified by the impact of the COVID-19 pandemic (Papadamou et al., 2021). Moreover, these models reduce their performance a little in forecasting abrupt market shifts induced by unprecedented occurrences, such as major regulatory changes. Despite these limitations in predicting and achieving profits, our models achieve a higher and more consistent level of cumulative profits over time than previous work on algorithmic trading models, especially high-frequency trading models (Dixon et al, 2018; Rundo, 2019; Lahmiri & Bekiros, 2021; Goudarzi & Bazzana, 2023).

Sovereign Bonds exhibit a fluctuating pattern over the years, with a negative start in 2001 but a significant shift towards positive gains in 2002. This positive trend continued until 2006, followed by intermittent fluctuations. By 2023, net gains had stabilized at a relatively positive level, demonstrating the resilience of these bonds. Corporate Bonds also had a negative start in 2001 but saw notable improvements in 2002, with consistent gains until around 2006. There was volatility in the subsequent years, with lucrative moments such as in 2010 but also difficulties. By 2023, corporate net gains appear to have regained a positive trajectory. High-Yield Bonds started in the negative in 2001 and remained mostly so until 2003. They then experienced a period of consistent gains until around 2011, followed by volatility. In 2023, they maintain positive cumulative gains, albeit more moderate.

At the beginning of the 2000s, central banks, particularly the U.S. Federal Reserve, had a more neutral monetary policy stance. Interest rates were relatively higher compared to the 2010s (Jarrow, 2019). However, following the burst of the dot-com bubble and the September 11 attacks in 2001, central banks, including the Federal Reserve, lowered interest rates to stimulate economic growth. These rate cuts resulted in lower yields on government bonds (Fabozzi & Fabozzi, 2021).

Bond yields, especially in the U.S., remained relatively low during the first half of the decade but started to rise as the economy improved. The latter part of the 2000s

was marked by the U.S. housing bubble and the subsequent global financial crisis of 2008. These events led to a flight to safety, with investors seeking refuge in government bonds, particularly U.S. Treasuries (Gilchrist et al., 2019). This increased demand for government bonds drove prices up and yields down.

Corporate bonds in the early 2000s offered higher yields compared to government bonds, reflecting the risk premium associated with corporate debt. However, during the financial crisis, corporate bond yields rose significantly as investors became concerned about the creditworthiness of corporations (Jarrow, 2019). Bond spreads, which measure the difference in yields between corporate bonds and government bonds, widened substantially during this period. Emerging market bonds experienced mixed performance during the 2000s. Some emerging market economies attracted foreign investment, leading to lower yields on their bonds. However, there were instances of bond market turmoil in emerging markets, driven by factors such as currency devaluations and political instability (Beirne and Sugandi, 2023).

Regarding the 2010s decade, central banks, particularly in developed economies like the United States, Europe, and Japan, implemented accommodative monetary policies in response to the global financial crisis of 2008 (Albagli et al., 2018). These policies included near-zero or negative interest rates and large-scale bond-buying programs (quantitative easing) aimed at stimulating economic growth. As a result, yields on government bonds, which serve as benchmarks for other fixed-income securities, remained historically low (Blanchard, 2023). The low yield environment prompted investors to seek higher-yielding assets, which sometimes led to increased demand for riskier bonds, such as high yield or corporate bonds. This increased demand pushed up bond prices and drove yields lower. The global economy experienced a prolonged period of low inflation and, at times, deflationary pressures during the 2010s. Low inflation expectations are often associated with lower yields on fixed-income securities (Fabozzi & Fabozzi, 2021).

Regulatory changes in the financial industry, such as Basel III banking regulations, encouraged financial institutions to hold more high-quality liquid assets, including government bonds (Ranaldo et al., 2019). This increased demand for government bonds also contributed to lower yields. While low yields were a prominent feature of the 2010s bond market, it's essential to note that not all bonds experienced the same level of yield compression. The extent of yield compression varied among different types of bonds, and some segments of the bond market, like high yield or emerging market bonds, offered higher yields to compensate for increased risk (Fabozzi & Fabozzi, 2021).

5 Conclusions

This study has developed a comparison of methodologies to predict bond price movements based on past prices through high-frequency trading. We compare ten machine learning methods applied to the fixed-income markets in sovereign, corporate and high-yield debt, in both developed and emerging countries, in the one-year bond market for the period from 15 February 2000 to 12 April 2023. Our results indicate that QGA, DRCNN and DLNN-GA can correctly interpret the

expected bond future price direction and rate changes satisfactorily. Curiously, QFuzzy is not adequate for forecasting high-frequency returns and dealers ought to avoid these models in their trading decisions, for the sample bond market.

Our study shows that all methods show better results concerning the Sign Prediction Ratio at lower frequencies. Thus, considering 10 min of frequency, the QGA method is the best performer with all the bonds, with an accuracy rate higher than 0.772 and a mean of 0.881. DRCNN and DLNNN-GA are the second and third methods that correctly predict the change in bond price direction, with an average of 0.850 and 0.847 respectively. However, for the Ideal Profit Ratio, not all methods show better results at the highest frequency. Some methods such as SVM-GA, ANFIS-QGA and DRCNN, perform the best model for the trading strategy configuration at 60-min sampling, and QGA performs the best for 30 min of sampling. Therefore, it is important to consider that depending on the sampling frequency and the objective of the approach, one method does not fit all, and a mixture of different alternative techniques must be examined.

In contrast to previous research, this study has achieved better accuracy results and has made a comparison of innovative methods of ML with the use of HFT, not been applied in the bond market so far. ML algorithms have become widely available for fixed-income market analysis, especially since uncertainty in the financial markets has risen sharply. In addition, our study has made predictions of bond price movements globally, hence it is not exclusively focused on industrialized countries. Finally, our study includes not only sovereign bonds but also corporate and high-yield debt, making it of interest to policymakers in any country.

Our study provides important benefits in the field of finance. From an insider trading perspective, it strengthens the implementation of reliable and fast forecast systems on the bond prices, including the pursuit of returns and volatility targeting, and can analyse the information of indirect market-based monetary policy instruments and the macro environment. Adequate bond price predictability can reduce medium- and long-term debt servicing costs through the development of a deep and liquid market for government securities. At the microeconomic level, the development of a robust bond price prediction model can increase overall financial stability and enhance financial intermediation via increased competition and the development of related financial infrastructure, products, and services. In addition, more generally, financial crises tend to arise in credit markets. Our model has the potential to provide financial institutions with information on the effects of policy measures on the credit market's fragility and to provide a better understanding of how market trends influence liquidity provision, implementation costs, and the impact on transaction prices.

In summary, our paper has a great perspective impact It can facilitate the work of professionals from financial institutions dedicated to trading as well as possible private investors and other stakeholders. This research makes an important contribution to high-frequency trading, as the conclusions have important implications both for investors and market participants as they seek to derive economic and financial profits from the bond market.

Our work has limitations in data availability for 10- and 30-min price frequencies for corporate debt securities. In order for this type of research to have greater

generalizability for fixed income market practitioners, greater data availability would be necessary. We leave this issue as a reason to explore future research in which more complex trading strategies can be organized to test and demonstrate the effectiveness of the techniques presented in this work for trading in debt securities.

Besides, further research should broaden the scope of the comparative analysis of methodologies to cover the field of crypto-assets, such as cryptocurrencies and fan tokens, since, in recent years, financial institutions have increasingly incorporated crypto-assets in their portfolios.

Appendix 1: Other Methodologies

Quantum-Fuzzy Approach (QFuzzy)

Singh et al. (2018) and Singh and Huang (2019) proposed recently the quantum optimization algorithm (QOP) modeled on the ''entanglement'' concept of quantum mechanics. In the present research, QOP is enhanced to resolve the multi-objective optimization problem (MOOP) and is called QFuzzy. In the case of MOOP, the major goal of QFuzzy is to choose the optimal solution set. For the operation of selecting a set of solutions, all the solutions are placed in a memory where they could be utilized to acquire the Pareto-optimal front screening out all the non-dominated optimal solutions. The concept of an archive is developed in this process, which stores all the non-dominated Pareto-optimal solutions (AONDPS). Subsequently, a selection criterion is assumed to choose the most prominent solution regarding the position of the quantum of the file. We then formulate a generalized MOOP to prove the application of QFuzzy toward seeking optimized solutions for the MOOP.

A MOOP has two goals formulated:

$$Optimize\ (Max.\,or\,Mim.)\Theta(x) = D_m(x), \quad x \in \mathbb{Q}^n \tag{33}$$

$$Optimize\ (Max.\ or\ Mim.)\beta(x) = Y_n(x), \quad x \in \mathbb{Q}^n \tag{34}$$

Under linear restrictions

$$D_m(x) \geq 0, m = 1, 2, \ldots, M \tag{35}$$

$$Y_n(x) \geq 0, n = 1, 2, \ldots, N \tag{36}$$

$$x_i^{(LB)} \leq x_i \leq x_i^{(UB)}, \quad i = 1, 2, \ldots, n \tag{37}$$

In Eq. (33), there are M objective functions: $\Theta(x) = \left(D_1(x), D_2(x), \ldots, D_M(x)\right)^T$. In Eq. (34), there are N objective functions: $\beta(x) = \left(Y_1(x), Y_2(x), \ldots, Y_N(x)\right)^T$. In this case, the objective function must be minimized or maximized. A solution x is a vector of n decision variables, where $x = (x_1, x_2, x_3 \ldots, x_m)^T$. The space extended by

the x_i is called the quantum system $\mathbb{Q}^n$, whereas the space created by the $\Theta(x)$ and $\beta(x)$ values are named the solution space. Finally, the last condition stated here indicates the restriction of the variable that limits the value of each x_i. For these restrictions, a condition set for a decision variable x_i is described as $G^{(LB)} \leq x_i \leq G^{(UB)}$ that constrains the value of each x_i within the lower bound ($G^{(LB)}$) and the upper bound ($G^{(UB)}$). To optimize the above MOOP with QFuzzy we have the following steps.

Step 1 Quantum initialization in the quantum system: Start every search agent according to the next equation of Schrodinger (1935) as follows:

$$Q_k(e) = \emptyset \cdot Q1_k(e) + \left(1 - \emptyset\right) \cdot Q2_k(e) \tag{38}$$

In Eq. (6), $Q_k(e)$ denotes the k-th quantum with an epoch e, and k = 1; 2; …; q; being q the total number of quanta in the $\mathbb{Q}^n$. $Q1_k(e)$ and $Q2_k(e)$ are two wave functions for the k-th quantum; $\emptyset = a + ib$ indicates a complex number, a and b denote real numbers in [0,1] and i is the imaginary unit $i = \sqrt{-1}$. In the representation of a complex number, multiplication by -1 refers to a 180-degree rotation about the origin of the k-th quantum. Therefore, the multiplication by i refers to a 90-degree rotation of the k-th quantum in the ''positive", in the counterclockwise direction (Berezin & Shubin, 2012). Because the complex number ∅ cannot be used straightforwardly to start the quantum in the search space, its absolute value is employed in the computational procedure, which is defined as $|\emptyset| = \sqrt{a^2 + b^2}$. $Q1_k(e)$ and $Q2_k(e)$ could be related as:

$$Q1_k(e) = \left\{G^{UB} + r_1 \cdot \left(G^{UB} - G^{LB}\right)\right\} \tag{39}$$

$$Q2_k(e) = \left\{G^{LB} + r_2 \cdot \left(G^{UB} - G^{LB}\right)\right\} \tag{40}$$

where $r_1 \in [0, 1]$ and $r_2 \in [0, 1]$ are two different random functions, correspondingly.

Step 2 Quantum localization: The acquired localization of the $Q_k(e)$ is given by $M_k(e)$, and may be stated as follows:

$$L_k(e) = \frac{1}{Q_k(e)} e^{-2/} Q_k(e) \tag{41}$$

Step 3 The motion of the quantum: The motion displayed by the $Q_k(e)$ is indicated by $M_k(e)$, and may be described as:

$$M_k(e) = \left| Q_k(e) - \frac{L_k(e)}{2} \ln\left(1/m_f\right) \right| \tag{42}$$

being m_f the "quantum movement factor", which can be in [0,1].

Step 4 Quantum shift: The displacement of $Q_k(e)$ is indicated by $D_k(e)$, and may be defined as:

$$D_k(e) = 2 \cdot \left| L_k(e) - M_k(e) \right| \tag{43}$$

Step 5 Evaluation of the suitability of the shift: A fitness value is established for $D_k(e)$, and updated if a solution better than the preceding one is available.

Step 6 Extension of the quantum search range: The extension of the motion, that is, $M_k(e)$ for the next epoch, e + 1 is expressed as $M_k(e+1)$, and can be stated as:

$$M_k(e+1) = M_1 + M_2 + M_3 \tag{44}$$

being

$$M_1 = \alpha \cdot M_k(e) \tag{45}$$

$$M_2 = \ln\left(1/m_f\right) \cdot r_3 \cdot \left[pBD_k(e) - D_k(e)\right] \tag{46}$$

$$M_3 = \ln\left(1/m_f\right) \cdot r_4 \cdot \left[gBD_k(e) - D_k(e)\right] \tag{47}$$

In this case, α is named the "quantum acceleration factor", expressed as:

$$\alpha = \alpha_{\max} - itr \times \frac{\left|\alpha_{\max} - \alpha_{\min}\right|}{Itr} \tag{48}$$

Being itr = 1,2,…,Itr. Itr indicates the maximum number of iterations set for the algorithm. Here, $\alpha_{\min}$ and $\alpha_{\max}$ can be taken in [0.1,0.9], where $\alpha_{\max} > \alpha_{\min}$. In formula (14), $pBD_k(e)$ is the personal best displacement that $D_k(e)$ has reached since the first epoch. In formula (15), $gBD_k(e)$ is the global best displacement achieved so far among the displacements. In formulas (14) and (15), $r_3 \in [0, 1]$ and $r_4 \in [0, 1]$ show two different random functions, correspondingly.

Step 7 Updating the quantum offset: The offset adjustment, that is, $D_k(e)$ for the next epoch is denoted by $D_k(e+1)$, and may be stated as:

$$D_k(e+1) = D_k(e) + M_k(e+1) \tag{49}$$

Step 8 Put AONDPSs into the file. Employing the AONDPSs, this algorithm starts by scanning the Pareto Optimal Front. Two components are integrated with the file to examine the optimal solutions as Knowles and Corne (2000): Controller and Grid. The addition of a particular solution to the file is determined by the controller. The information in the file is regarded as up to date according to one of the below conditions:

Condition 1: If the optimal non-dominated Pareto solution is missing and the file is empty, then the present solution should be adopted.

Condition 2: If an optimal solution is mastered by any other factor within the file, that particular solution must be rejected.

Condition 3: If a Pareto optimal solution is not mastered by the external factor, then the particular solution should be adopted and kept in the file.

Condition 4: If the optimal solutions are controlled by the new element, they are removed from the file.

Finally, the other component is Grid: When the AONDPS are positioned in the file, a solution space is generated for every objective. Recursively bisecting the solution space produces an individual placement called a grid location. A grid location helps identify how many non-dominated Pareto solutions are in a grid and where they are located.

Step 9 Stop the algorithm if the stop condition is satisfied; if not, go back to step 7.

Adaptive Boosting and Genetic Algorithm (AdaBoost-GA)

According to the conventional AdaBoost algorithm, every base classifier's weight is set after being computed; and the classifier adaptivity of each base classifier is not regarded (Wang et al., 2011). Hence, in this research, the GA is utilised in the adaptive integration procedures of the base classifiers. The number of decision groups is the number of weak classifiers Adaboost, and the weight of every weak classifier is the starting population of GA.

Both crossover probability and mutation probability significantly impact the algorithm's optimization effect (Cheng et al., 2019). To choose the suitable crossover likelihood and mutation likelihood, based on the literature (Drezner and Misevicius, 2013), we describe the crossover likelihood and mutation likelihood as:

$$P_c = \gamma \tag{50}$$

$$P_m = 0.1(1 - \gamma) \tag{51}$$

being γ a regulatory factor.

The fitness function was described as:

$$fit = \frac{\sum_{i=1}^{N} I\left(y\left(X_i\right) = y_i\right)}{N} \tag{52}$$

AdaBoost-GA

The function of Adaboost is to build various base classifiers using training the data distribution and thereafter allocate weights to these base classifiers by the error rate. Adaboost uses the decision group as the base classifier to enlarge the system diversity of the ensemble set and uses the GA algorithm to maximise the weight of each base classifier by combining all the base classifiers.

Given $\{\omega_{m,j}|i = 1, 2, \ldots, N; m = 1 \ldots, M\}$, which is the weight of every sample in the base classifier. $\omega_{m,j}$ symbols the weight of the ith sample in m_{th} base classifier. Let $y_m(x)$ and Y(x) be a base classifier and strong classifier, respectively. α_m denotes the weight of mth classifier. ε_m represents the error function of mth base classifier. And M constitutes the number of base classifiers. The AdaBoost-GA suggested algorithm could be explained below.

Input:

-Training sets

-Validation sets

-$\omega_{1,i}$: The weight of each training sample

Output:

Y(.): The final strong classifier:

$$Y(x) = sign\left(\sum_{j=1}^{M} \alpha_j y_j(x)\right) \tag{53}$$

1. Initialize $\omega_{1,i} = 1/N$.

2. For i = 1 to N do.

3.

$$\varepsilon_m = \sum_{i=1}^{N} \omega_{m,i} I\left(y_m\left(X_i\right) \neq y_i\right) \tag{54}$$

4.

$$\alpha_m = 1n\left\{\frac{1 - \varepsilon_m}{\varepsilon_m}\right\} \tag{55}$$

if $\alpha_m \geq 0$ then α_m increases with the decrement of ε_m. End if

5.

$$\omega_{m+1,i} = \frac{\omega_{m,i}}{Z_m} \exp\left(-\alpha_m y_i y_m(x_i)\right) \tag{56}$$

where

$$Z_m = \sum_{i=1}^{N} \omega_{m,i} \exp\left(-\alpha_m y_i y_g(x_i)\right) \tag{57}$$

6. end for

$$\alpha_m = GA(\alpha) \tag{58}$$

$$Y_x = sign\left(\sum_{j=1}^{M} \alpha_j y_j(x)\right) \tag{59}$$

Return Y(x).

Support Vector Machine- Genetic Algorithm (SVM-GA)

The problem of SVM parameter setting has been addressed by several approaches ranging from raw force to more refined metaheuristics, one of the best known of which is genetic algorithms (GA). The major benefit of GAs compared to simpler methods is their ability to deliver stochastic near-optimal solutions at modest cost, while simultaneously optimising multiple parameters with no prior knowledge (Goldberg, 1990). Similarly, every parameter in the search space is encoded as an allele or gene in a GA, and the complete configuration of a specific solution has termed a chromosome. The native formulation of GAs encodes every gene in binary form (binary genetic algorithms, BGAs) so that multiple evolutionary operators can be successfully implemented. Such encoding, however, leads to needless computational expense because it transforms real values into their binary value, and conversely; the storage expense is raised because operators such as mutation and crossover require only a single pair of bits to fulfil their function (Chih-Hung et al., 2009). Another method that improves storage and computational costs is called real-valued genetic algorithms (RGA). The latter type of coding is employed in the present work.

However, GAs suffer from some disadvantages, such as early conversion to local optima owing to their genetic operators, and their robustness is only obtained when parameters such as the population size or the number of generations are adjusted. In

addition, GA outcomes become less reproducible and need a statistical process to assure solution configuration conversion. Moreover, by keeping the optimal solution in the population, a GA is capable of converging to the global optimum (Sivaraj & Ravichandran, 2011).

Thus, key features in the development of evolutionary algorithms are the genetic operators like selection, crossover, and even the random number generator (RNG) they employ. The choice operator sets the search neighbourhoods, whereas the operators of recombination and mutation search for a particular space. In this work, we suggest a new Boltzmann operator that similarly describes a cooling scheme that the linear schedule suggested by Kirkpatrick et al. (1983). Besides, the convergence properties of a genetic algorithm may be improved by applying the chaotic number generators (Caponetto et al., 2003). In this study, three chaotic sequences have been also employed by SV RGBC genetic operators.

SVR_{GBC} Approach

A major stage in developing an efficient support vector model for ranking or regression is the adjustment of the parameters. Typically, this procedure accounts for the error compensation constant (C), the choice of a kernel, and its related kernel-specific constants. The easiest way to conduct this fitting procedure is called grid search (GS). This approach involves generating multiple SVM models starting from the learning step with a set step for the parameter values. Best values are achieved through a test of each model on a validation set and the choice of the best result. However, this method has various disadvantages, the most important of which are: a priori knowledge, large computational expense, local optima, and the uncertain distribution of the parameter values. An alternative method for SVM parameter adjustment is GA techniques; these are capable of coping with the above detractors owing to their demonstrated efficiency in the contextual handling of the model constants. GAs performs a non-linear query of the solution space based on no knowledge of the model's characteristics.

For the SVR hyperparameter setting in the volatility prediction task, our novel approach considers a triplet of parameters composed of the kernel type and a set of kernel-specific constants. Starting with an original population of chromosomes produced by a pseudo-random or chaotic distribution, the process begins. Every chromosome is composed of an integer-valued and a real-valued part. The method of choosing individuals for the mating pool in each generation can be elitism, roulette, or the suggested Boltzmann choice method. The rest of the mating pool is filled with an n-point crossover operator and a boundary mutation technique (Chih-Hung et al., 2009; Ping-Feng et al., 2006), employing pseudorandom and chaotic sequences.

Genetic Operators

SVR_{GBC} is a mix of an integer-valued and a real-valued genetic algorithm; it uses various genetic operators such as selection, crossover, and mutation to generate offspring from the population of real solutions. Three methods of selection are included

in our process: Elitism, the roulette wheel, and a new selection method named Boltzmann selection. Such systems are employed to select the best offspring to proceed with the development cycle. The chosen individuals are then gathered into a breeding pool and the crossover and mutation operators are performed on them. An inconvenience of the GA crossover operation in the SVR parameter setting problem is chromosomal heterogeneity. A solution to this problem is the employment of a dominance scheme, as indicated by Lewis et al. (1998). Every gene's value is determined by its kernel together with an upper and lower limit of the allele. The mutation is the following stage, where several chromosomes are targeted for a mutation to yield changed clones that will be incorporated into the new population. The present work is based on a uniform mutation, represented as:

$$\begin{aligned} c^{old} &= \left\{c_1, c_2, \ldots, c_i, \ldots c_n\right\} \\ c_i^{new} &= LB_i + r * \left(UB_i - LB_i\right) \\ c^{new} &= \left\{c_1, c_2, \ldots, c_i^{new}, \ldots c_n\right\} \end{aligned} \tag{60}$$

being c^{old} and c^{new} a chosen chromosome before and after mutation accordingly; n enumerates the number of genes in the chromosome structure; c_i^{new} represents the new value for the i allele after mutation; r symbols a random number in the range [0,1], produced by one of the available probability distributions; LB_i and UB_i are the lower and upper bound of the i allele. The various ranks and kinds of SVR parameters require an integer-valued mutation and a real-valued mutation (Chih-Hung et al., 2009). The former is employed to handle the kernel type K_T owing to its integer encoding, whereas the latter changes P_1 or P_2 values. For these ends, a minor alteration of the displayed mutation operator is needed: a rounding function is introduced right after the perturbation of allele i to guarantee correct values.

Boltzmann Selection

As Kirkpatrick et al. (1983) found, the solution acceptance function constitutes an essential feature of simulated annealing (SA). Once the SA procedure is looped for a sufficient amount of time, the solution acceptance distribution function follows the Boltzmann distribution. SA is comprised of three items: A probabilistic acceptance criterion, a neighbourhood exploration approach, and a cooling function to reach thermal balance.

Goldberg (1990) suggested that the SA-like mechanism will improve the capabilities of the GA given the thermal equilibrium afforded by the cooling function, and the effectiveness of the Boltzmann distribution demonstrated to be an exploration heuristic (Goldberg, 1990; Kirkpatrick et al., 1983). Guided by the work of Goldberg (1990), we suggest Boltzmann selection, used to choose the surviving set of mats based on the temperature of the system given by a cooling schedule. Every solution is either accepted or refused following the Boltzmann distribution, defined by (29)

$$P_{(x_i)} = e^{\left(\frac{-\Delta E}{KT}\right)} \tag{61}$$

being k the Boltzmann constant, ΔE represents the energy among the best and the present solution, and T denotes the actual system temperature. The latter parameter is achieved in SA by utilising a cooling function from the classical exponential or linear options of annealing schemes (Kirkpatrick et al., 1983). In this work, a cooling scheme linked to the linear cooling function for the AG, which correlates the temperature with the present generation and the number of total generations of the AG, is presented.

Fitness Function

The optimisation procedure of the SVR parameters via the GA needs a fitness function to assess and choose the chromosomes for the matting set. In this paper, x MSE is employed to evaluate the quality of every solution to maintain a good genetic material. This is determined by (30), being σ_t the observed volatility for period t,$\hat{\sigma}_t$ represents the forecasted volatility for period t, and n symbols the total forecasted time frame.

$$MSE = \frac{\sum_{i=1}^{n} \left(\sigma_t - \hat{\sigma}_t\right)^2}{n} \tag{62}$$

In addition, the MSE is computed using a statistical estimate of the generalisation error named k-fold cross-validation (CV). This refers to a method of calculating the parameter values of a model based on a training sample (Kohavi, 1995; Refaeilzadeh et al., 2009). The sample, taken in the simplest case, is split into k independent subsets of equal size. Using $k-1$ of these, a model is trained and 42 is computed over the residual subset. This procedure is repeated for the remaining $k-1$ left samples, and then the mean is calculated. At last, the model that reduces the CV value to the minimum is the optimal one (Refaeilzadeh et al., 2009).

Deep Learning Neural Network- Genetic Algorithm (DLNN-GA)

The procedure of multiple forward dispersion is defined in the next linear model linking the incident optical modes and the transferred optical modes (Vellekoop & Mosk, 2008).

$$E_m = \sum_{n=1}^{N} t_{mn} E_n = \sum_{n=1}^{N} \left|t_{mn}\right| \exp\left(i\phi_{mn}\right) \left|E_n\right| \exp\left(i\phi_n\right) \tag{63}$$

being E_n the nth complex incident mode with amplitude $\left|E_n\right|$ and phase ϕ_n, while E_m represents the mth complex optical mode transferred from the dispersion media. t_{mn} symbols one element in the complex transmission matrix which constitutes light

scattering paths. The phase values fulfil $\phi_n = -\phi_{mn}$ (Bossy & Gigan, 2016). The light will focus perfectly on the selected location when it is set to this condition.

The procedure of using the GA for wavefront modelling involves five steps: initialisation, classification, reproduction, mutation, and iteration. First, a given number G of phase patterns is generated, each phase value being selected from a uniform pseudo-random distribution. Next, these standards are marked using a specially designed fitness function. Following formula (32), the fitness function is described as the intensity of the light at a given location (Conkey et al., 2012)

$$I_m = \left|E_m\right|^2 = \frac{1}{N}\left|\sum_{n=1}^{N} t_{mn} A_n \exp\left(i\phi_n\right)\right|^2 \tag{64}$$

being A_n the amplitude of E_n. Phase patterns are classified according to the results of the fitness function assessment. High scores lead to higher rankings. The second step is breeding. Offspring is produced by offspring = T × ma + (1 − T) × pa, with T being a random binary template and ma and pa being parental parents. Both parents are sorted under the rule that higher-ranking parents are more likely to be adopted. After reproduction, certain sections of the offspring become mutated and changed by chance. The mutation rate R declines with growing generations n to prevent over mutation, as follows $R = \left(R_0 - R_{end}\right) \times \exp\left(-n/\lambda\right) + R_{end}$, being R_0, R_{end}, and λ the initial mutation rate, the final mutation rate, and the decay factor, for each of them (Conkey et al., 2012). The progeny shall also be assessed by the fitness function. In each generation, a certain number of offspring will be replicated to substitute already existing patterns with inferior scores (Conkey et al., 2012). Then, the G-stage patterns are all reclassified based on their scores. The previous reproduction and mutating proceedings will be repeated several times before the final condition is fulfilled. Usually, the iteration ceases when a prespecified amount of generations is replayed or the result of the fitness function evaluation meets a certain level of threshold.

The advantages of the AG are important. The AG manages to identify a suitable solution quickly. In addition, GA is robust to noise, as it updates the largest number of pixels rather than adjusting pixels one by one. While GA results are significantly affected by many factors, such as the mutation and reproduction rate, the fitness function, and particularly the number of phase patterns employed in every generation, namely the size of the population, matching the right parameters is not trivial and needs time and experience. Furthermore, several scenarios haphazard introduce start patterns, potentially in the neighbourhood of one or more local minima. GA is susceptible to getting locked into a local minimum, as it is a staged optimisation process and descendants reproduce themselves by replicating and mutating existing patterns. This involves the risk that likely better solutions cannot be probed. Consequently, the use of a good initialisation is crucial to reach global optima (Bodenhofer, 2003).

Deep learning, which is a data-driven process, employs a separate strategy to estimate the phase pattern for focusing the light. The poor proposal and nonlinearity of inverse scattering problems show that direct inversion is impractical, which makes

the requirements of iterative algorithms with regulation (Wei & Chen, 2018) to be as below:

$$arg\min_{p} \; y - HW_{p2}^{2} + \lambda p_1 \tag{65}$$

being H the forward scattering model, y represents the recorded speckle intensity, W symbols a transformation, and p denotes transformation coefficients so that $\hat{x} = W_p$ is the desired reconstruction.

Nearly every state-of-the-art iterative algorithm for back-scattering problems are cascades of linear convolutions and pointwise nonlinear transactions, (McCann et al., 2017), resembling the structure of convolutional neural networks (CNNs). Examples of a representative implementation are the well-known iterative shrinkage-thresholding algorithms (ISTAs) founded on model blocks as follows:

$$p^{m+1} = A_\theta\left[\frac{1}{L}W * H * y + \left(I - \frac{1}{L}W * H * HW\right)p^m\right] \tag{66}$$

being L the Lipschitz constant. The iterative optimization governed by formula (60) can be handled as a convolutional procedure with kernel $I - \frac{1}{L}W * H * HW$ and bias (1/L)W*H*y, followed by a nonlinear activation function A_θ. So, CNNs can be regarded as inherently suitable for solving the problem of reverse dispersion (Li et al., 2018; Wei & Chen, 2018).

Conventional iterative algorithms are successful and warrant the use of CNNs as a way to approach light via scattering media. Deep CNNs (DCCNs) have been demonstrated powerful in solving inverted problems (Li et al., 2018; Lucas et al., 2018).

Therefore, DCNNs can model the H -1 reverse scattering process through supervised learning, disclosing the connection between the transferred speckles and the incident x optical phase patterns. The inputs of the DCNN are the measured intensity distributions of the transferred speckles captured with a camera, and the outputs are their incident phase patterns modified by a spatial light modulator (SLM). Upon training, the DCNN will accurately map the speckle to the incident phase patterns, and therefore the DCNN can forecast the phase pattern needed to target the light via a particular dispersion media.

Using the DL approach is easier, as the relation of speckle to incident phase patterns is acquired directly via training. However, its output is influenced strongly by the training samples. Accordingly, these samples are typical to represent the complete scattering processes only when the training sample size is large enough, and thus, the DCNN can forecast the best phase pattern for the light focus following the training. However, the supervised learning method experiences the dilemma of achieving the global optimum because it is imprudent to incorporate every conceivable phase pattern and speckle for training, and sample topics are hard to estimate as well. Despite this, the DCNN results can be considered a good initialisation for the GA. Concerning the GA, maximum global convergence is obtained under the constraint that the initial figure is close to the global optima (Bodenhofer, 2003).

The suggested GeneNNN contains two parts. The first part consists of gathering samples to train a DCNN. Following the training, an early focus mote with the phase pattern predicted by the DCNN can be obtained. The second part is the adoption of

the GA for optimising the focused process. We present two methods for building early phase patterns, each employing the DCNN results. The first method, called GeneNNv1, adopts the pattern foreseen by the DCNN for one of the starting patterns, whereas other patterns are generated according to a uniform pseudo-random distribution (Conkey et al., 2012). The DCNN pattern will undoubtedly have the strongest classification, thus having the highest chance of being selected for breeding. With the other method, called GeneNNNv2, all starting patterns are built according to the DCNN patterns predicted by the DCNN but adding several extra phase patterns to this general basis, whereas all additional patterns are generated with a uniform pseudo-random distribution as well.

Adaptive Neuro-Fuzzy Inference System-Quantum Genetic Algorithm (ANFIS-QGA)

For this method, the likelihood amplitudes of every qubit are considered to be two genes, each chromosome comprises two gene strings, and every gene string stands for an optimisation solution. The number of genes is fixed by the number of optimisation parameters. With each qubit of the optimal chromosome as a target, singles are actualised by quantum rotation gates and mutated by non-quantum gates so that population diversity is increased. The mutation procedure was performed by the non-quantum gates and the crossover and selection operations were performed by the quantum rotation gates. The varying tendency of the fitness function at the search point is transferred into the rotation angle calculation function design. If the change ratio of the fitness function at a certain search point is larger than that at other points, the rotation angle is decreased appropriately. With this method, each chromosome can be made to advance in the flatness of the search process to quicken the conversion and hurry in the sweep of the searching operation to prevent losing the globally optimal solution (Cao & Shang, 2010).

The key steps of the suggested model are outlined in Algorithm 1.

Algorithm 1: Suggested model

Step 1: Data Pre-processing (Phase one) Step 2: Generate random population Step 3: Calculate the value of Radii Step 4: Initialize Anfis model Step 5: Perform optimization algorithm using DCQGA (Phase two). Step 6: Finding the best accuracy and performance. Step 7: Stop.

The application of the suggested model comprises two major steps.

Phase one: Data Pre-processing: This stage consists of a procedure that transforms the raw inputs and outputs into an acceptable form before the training process. It is mainly used to decrease the dimensionality of the input data and to improve the performance of the generalisation (Bishop, 1995).

The original data are assigned to [0,1] using min–max normalisation:

$$X(i) = \frac{x(i) - m}{M - m} \tag{67}$$

for the time series data x, m = min{x}, M = max(x). Four types of time series, for example, the opening price, the closing price, the highest price, and the lowest price are separately standardised in the experiment.

Phase two: Optimization algorithm: The algorithm used in this step is inspired by the double-stranded quantum genetic algorithm (Cao & Shang, 2010) and is designed as below:

1. Produce the original angle to create the double strand from it.

$$P_{i,1} = \left(\cos\left(y_{i,1}\right), \cos\left(y_{i,2}\right), \ldots, \cos\left(y_{i,n}\right)\right) \tag{68}$$

$$P_{i,2} = \left(\sin\left(y_{i,1}\right), \sin\left(y_{i,2}\right), \ldots, \sin\left(y_{i,n}\right)\right) \tag{69}$$

being $y_{i,n}$ a random number between 0 and 2π, $P_{i,1}$ named cosine solution and $P_{i,2}$ called sine solution.

2. Execute solution space transform

$$X_{(i,j)c} = 0.5^{*}\left[b_i\left(1+\alpha_{i,j}\right) + a_i\left(1-\alpha_{i,j}\right)\right] \tag{70}$$

$$X_{(i,j)s} = 0.5^{*}\left[b_i\left(1+\beta_{i,j}\right) + a_i\left(1-\beta_{i,j}\right)\right] \tag{71}$$

where i = 1: m, j = 1: n, m is the number of qubits, and n is the population size

3. Compute fitness value being equivalent to

$$\frac{1}{1+MSE} \tag{72}$$

4. Renovate the best composition.

$$\Delta\theta_{i,j} = -sgn(A)\Delta\theta_0 exp\left(-\frac{\left|\nabla \mathrm{f}\left(X_{i,j}\right)\right| - \nabla f_i min}{\nabla f_i max - \nabla f_i min}\right) \tag{73}$$

5. Execute quantum rotation gates to renovate corresponding qubits on the actual chromosome for every chromosome.

Being $\Delta\theta_0$ the initial value of rotation angle,

$$A = \begin{vmatrix} \alpha_0 & \alpha_1 \\ \beta_0 & \beta_1 \end{vmatrix} \tag{74}$$

Therefore, the rotation angle Δθ can be fixed by rules such as the following: if A ≠ 0, then sgn (Δθ) = −sgn(A), else if A = 0, the direction of Δθ is arbitrary,

$$\nabla \mathrm{f}\left(X_{i,j}\right) = \frac{\mathrm{f}\left(X_{i,p}\right) - \mathrm{f}\left(X_{i,c}\right)}{\left(X_{i,j}\right)^{p} - \left(X_{i,j}\right)^{c}} \tag{75}$$

$$\nabla f_i max = max\left(\left|\frac{f(X_{1,p}) - f(X_{1,c})}{(X_{1,j})^p - (X_{1,j})^c}\right|, \left|\frac{f(X_{2,p}) - f(X_{2,c})}{(X_{2,j})^p - (X_{2,j})^c}\right|, \ldots, \left|\frac{f(X_{n,p}) - f(X_{n,c})}{(X_{n,j})^p - (X_{n,j})^c}\right|\right) \quad (76)$$

$$\nabla f_i min = min\left(\left|\frac{f(X_{1,p}) - f(X_{1,c})}{(X_{1,j})^p - (X_{1,j})^c}\right|, \left|\frac{f(X_{2,p}) - f(X_{2,c})}{(X_{2,j})^p - (X_{2,j})^c}\right|, \ldots, \left|\frac{f(X_{n,p}) - f(X_{n,c})}{(X_{n,j})^p - (X_{n,j})^c}\right|\right) \quad (77)$$

being $X_{i,p}$ and $X_{i,c}$ ith vector in solution space like the parent colony and child colony, and $(X_{1,j})^p$, and $(X_{1,j})^c$, denotes the jth variable of the vector $X_{i,p}$ and $X_{i,c}$, correspondingly.

6. Mutate qubits based on the likelihood of mutation (Pm = 0.1) equivalent to 1 over the population size (pop size = 10) and these figures are derived experimentally for acceptable speed and efficiency for the model.

The above-listed methods have been used to determine the optimal value for the optimisation parameter utilizing the double-string quantum genetic algorithm and to compute the fitness function by the ANFIS model employing subtractive clustering to cause initial FIS for the ANFIS system.

Convolutional Neural Networks-Long Short-Term Memory (CNN-LSTM)

CNN is characteristic of attending to very evident properties within the sight line, therefore, it is extensively applied in engineering. LSTM features the characteristic to expand based on the time sequence, and it makes a large use in the time series. Following the characteristics of CNN and LSTM, a value forecasting model based on CNN-LSTM is constructed.

CNN was developed as a network model by Lecun et al. in 1998. CNN is a type of feed-forward neural network, which performs well in both image and natural language processing (Kim & Kim, 2019). CNN could successfully be implemented in time-series forecasting. CNN local sensing and distribution of weights may reduce the parameter number to a large extent, thereby increasing the learning efficiency of the model (Qin et al., 2018). CNN consists of two parts mainly: the convolution layer and the clustering layer. The convolution layer each holds a multiplicity of convolution kernels, and its formula of calculation is given in Eq. (46). Following the convolution operation of the convolution layer, features are removed from the data, however, the dimensions of the separated characteristics become high, therefore, to resolve this issue and decrease the training cost of the network, a clustering layer is inserted directly after the convolution layer for reducing the dimension of the characteristics:

$$l_t = tnh(x_t * k_t + b_t) \quad (78)$$

being l_t the output value after convolution, t_{nh} represents the activation function, x_t stands for the input vector, k_t means the weight of the convolution kernel, and b_t symbols the bias of the convolution kernel.

LSTM is aimed at overcoming the long-standing explosion and vanishing gradient problems in Recurrent Neural Networks (RNNs) (Ta et al., 2020). It has been largely employed in speech detection, sentiment analysis, and text processing since it has a unique memory and can make relatively precise predictions (Gupta & Jalal, 2020). The LSTM contains three parts: the forgetting gate, the input gate, and the output gate.

The computational procedure of the LSTM follows:

1. The output last time value and the input current time value are entered into the forgetting gate, and the output forgetting gate value is determined after calculation, according to the equation below:

$$f_t = \sigma\left(W_f.\left[h_{t-1}, x_t\right] + b_f\right) \tag{79}$$

being the value range of f_t (0,1), W_f symbols the weight of the forget gate, and b_f represents the bias of the forget gate, x_t is the input current time value, and h_{t-1} is the output last time value.

2. The output last time value and the input current time value are entered in the input gate, and the output value and the status of the input gate candidate are derived after the calculation, illustrated by the below equations:

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right) \tag{80}$$

$$\tilde{C}_t = \tanh\left(W_c \cdot \left[h_{t-1}, x_t\right] + b_c\right) \tag{81}$$

Being the value range of i_t (0,1), W_i represents the weight of the input gate, b_i symbols the bias of the input gate, W_c is the weight of the candidate input gate, and b_c represents the bias of the candidate input gate.

3. Update the current cell state as follows:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{82}$$

where the value range of C_t is (0,1).

4. The output h_{t-1} and input x_t are taken at time t as the input values of the output gate, and the output o_t of the output, the gate is determined by:

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right) \tag{83}$$

being the value range of o_t (0,1), W_o symbols of the weight of the output gate, and b_o represents the bias of the output gate.

5. The LSTM output value is achieved by computing the output of the output gate and the cell status, based on the formula below:

$$h_t = o_t * \tanh\left(C_t\right) \tag{84}$$

The principal stages of the CNN-LSTM training and prediction procedure are listed below:

1. Enter the necessary data for the training of the CNN-LSTM.
2. Since the gap in the input data is large, for better training of the model, the z-score standardisation approach to standardise the input data is assumed, illustrated by the formulas below:

$$y_i = \frac{x_i - \overline{x}}{s} \tag{85}$$

$$x_i = y_i * s + \overline{x} \tag{86}$$

being y_i the standardized value, x_i represents the input data, $\overline{x}$ symbols the average of the input data, and s is the standard deviation of the input data.
3. Initiate the weighting of each layer of the CNN-LSTM and biases.
4. Input data passes successively via the convolution layer and the clustering layer in the CNN layer, the input data is feature extracted and the output value is acquired.
5. The CNN layer output data is computed via the LSTM layer, and the output value is given.
6. LSTM layer output value is fed into the complete connection layer to obtain the output value.
7. The output value computed via the output layer is checked against the actual value of this dataset, resulting in the respective error.
8. The completion conditions are that a specified number of cycles are reached, that the weight is below a specified limit and that the forecast error rate is below a specified limit. If one of the end conditions is achieved, the training shall be fulfilled, the entire CNN-LSTM network shall be updated and go to step 10; if not, proceed to step 9.
9. Propagates the computed error in reverse, maintains the weight and bias of every layer, and proceeds to step 4 to further train the network.
10. Save the model trained for the forecast.
11. Enter input data needed for the forecast.
12. The input data are normalised by Eq. (40).
13. Input the calibrated data into the CNN-LSTM trained model, and subsequently obtain an output value for it.
14. The output value provided by the CNN-LSTM model is the normalised value, and the normalised value is subtracted from the original value. As given by formula (41), where x_i the restored normalised value, y_i is the CNN-LSTM output value, s is the standard deviation of the input data, and x is the mean value of the input data.
15. Issue the results restored for completing the prediction procedure.

Gated Recurrent Unit- Convolutional Neural Networks (GRU-CNN)

RNN is a type of artificial neural network suitable for processing and analysing temporal data sequences, in contrast to classical neural networks, which rely on the weighting connection between the layers. The RNN implements the hidden layers to retain the information of the prior time, and the output is affected by the present states and the memories of the prior time. The structure of the unrolled RNN is presented as follows:

$$\begin{aligned} a^t &= g_1\left(\omega_{aa}a^{t-1} + \omega_{ax}x^{t-1} + b_a\right) \\ \hat{y}^t &= g_2\left(\omega_{ay}a^t + b_y\right) \end{aligned} \tag{87}$$

being a^t the output of one single hidden layer at time t, and ω^t_{aa}, ω^t_{ax}, and ω^t_{ay} are the hidden layers' weight matrixes, the input weight matrixes, and the output weight matrixes, respectively. b_a and b_y symbol the bias vectors of one single hidden layer and the output, respectively, and g_1 and g_2 represent the nonlinear activation function.

The RNN works properly if the output is near its related inputs; nevertheless, with a long-time-interval and a great number of weights, the input shall not have much effect on the output owing to the problem of disappearing gradient. To resolve the gradient vanishing problem and the simple structure of the RNN hidden layer, we proposed a particular kind of RNN called GRU.

The GRU consists of a variation of the LSTM with a closed RNN structure, and in comparison, to the LSTM, there are two gates (update gate and reset gate) in the GRU and three gates (forget gate, entry gate, and exit gate) in the LSTM (Alaminos et al., 2022a; Gao et al., 2019).

The GRU equations are:

$$\begin{aligned} \Gamma_u &= \sigma\left(\omega_u\left[c^{\langle t-1\rangle}, x^{\langle t\rangle}\right] + b_u\right), \\ \Gamma_r &= \sigma\left(\omega_r\left[c^{\langle t-1\rangle}, x^{\langle t\rangle}\right] + b_r\right), \\ \tilde{c}^{\langle t\rangle} &= tanh\left(\omega_c\left[\Gamma_r * c^{\langle t-1\rangle}, x^{\langle t\rangle}\right] + b_c\right), \\ c^{\langle t\rangle} &= \left(1 - \Gamma_u\right) * c^{\langle t-1\rangle} + \Gamma_u * \tilde{c}^{\langle t\rangle}, \end{aligned} \tag{88}$$

being ω_u, ω_r, and ω_c the training weight matrix of the update gate, the reset gate, and the candidate activation $\tilde{c}^t$, respectively and b_u, b_r, and b_c represent the bias vectors.

The Establishment of CNN Module

CNN is often employed in visual image and video recognition, and text categorisation. To keep the spatial information of the data registered by sensors and smart appliances in the power system, the spatio-temporal matrix was suggested. Its data are based on the sensors' location and time sequence. The spatio-temporal matrix appears like this:

$$x = \begin{bmatrix} X_1(1) & \cdots & X_n(n) \\ \vdots & \ddots & \vdots \\ X_k(1) & \cdots & X_k(n) \end{bmatrix} \tag{89}$$

being k the k th smart sensor, n is the n th time sequence, and $X_k(n)$ symbols of the data recorded by the k th a smart sensor at n time. For extracting the charging characteristic from the spatio-temporal matrix, CNN was utilised for processing the spatio-temporal matrix.

First, numerous two-dimensional space–time matrices are piled into blocks of three-dimensional matrices, followed by applying these blocks with a convolution operation. The convolution operation aims to obtain a strongly abstract feature, then after the convolution operation, the results of the convolution operation are applied to the grouping operation. The pooling operation makes no change to the entry matrix depth, though it may decrease the size of the matrices as well as the number of nodes so that the parameters in the complete neural networks are reduced. Following the repeated convolution and pooling operations, the highly abstract feature was extracted and smoothed to a one-dimensional vector, hence it can be linked to the whole layer connected. Next, the weights and bias parameters within the globally connected layer can be computed iteratively. Lastly, the forecasting outcomes are provided by the output of the activation function.

The GRU-CNN Hybrid Neural Networks

Our proposed GRU-CNN hybrid neural networks framework is composed of a GRU module and a CNN module. Inputs are the time sequence data information and spatio-temporal matrices recorded from the power system; outputs represent the forecasting of the future load value. As for the CNN module, it is good for processing two-dimensional data, such as spatio-temporal matrices and images. CNN module employs local connection and weight sharing to extract local characteristics of the data directly from the spatio-temporal matrices and get an efficient presentation using the convolution layer and the clustering layer. CNN module structure contains two convolution layers and one flattening operation, with each convolution layer containing one convolution operation and one clustering operation. Following the second clustering operation, high-dimensional data is smoothed into one-dimensional data, and the outputs of the CNN module become linked to the connected layer. Moreover, the purpose of the GRU module is to catch the long-term dependency, and the GRU module could gather helpful information in the historical data over a long period via the memory cell, while the useless data will be ignored by the oblivion gate. GRU module inputs are the temporal sequence data; the GRU module holds plenty of closed recurrent units, and the outputs of all these closed recurrent units are linked with the connected layer. At last, the load forecasting outcomes may be achieved with the average value of all the neurons in the gated layers.

Quantum Recurrent Neural Network (QRNN)

A quantum system on n qubits exists in the n-fold Hilbert space of tensor product $\mathcal{H} = \left(\mathbb{C}^2\right)^{\otimes d}$ with resulting dimension 2^d. A quantum state represents a unit vector $\psi \in \mathcal{H}$, commonly described in quantum computing in bra-ket notation $|\psi\rangle \in \mathcal{H}$ |; its conjugate transpose with $\langle\psi| = |\psi\rangle^{\dagger}$; then the inner product $\langle\psi|\psi\rangle = \|\psi\|_2^2$ means the square of the 2-norm of $\psi \cdot \left[\psi\rangle\langle\psi\right]$ then denominates the outer product, yielding a tensor of rank 2. The computational ground conditions correspond to $|0 = (1, 0), |1\rangle = (0, 1)$, and compound ground states are for example set by $|01\rangle = |0\rangle \otimes |1 = (0, 1, 0, 0)$.

Thus a quantum gate becomes a unitary operation $\mathbb{U}$ on $\mathcal{H}$; where the operation nontrivially operates on a subset $\mathbb{S} \subseteq [n]$ of qubits, then $\mathbb{U} \epsilon \mathbb{S} \mathbb{U}\left(2^{|\mathbb{S}|}\right)$; to operate on $\mathcal{H}$ we expand $\mathbb{U}$ to operate as identity on the remainder of the space, i.e. $\mathbb{U}_{\mathbb{S}} \otimes 1_{[n]\setminus\mathbb{S}}$. This extension is usually ignored, and indicates if the gate operates in a quantum circuit: the first gate R(θ) represents a unitary of a qubit that operates on the second qubit from below, and which depends on the parameter θ. The dotted line extending from the gate designates a "controlled" operation, if the control, for example, acts only on a single qubit denominates the single block-diagonal unitary map $|0\rangle\langle 0| \otimes 1 + |1\rangle\langle 1| \otimes \mathbb{R}(\theta) = 1 \oplus \mathbb{R}(\theta)$ it stands for "if the control qubit is in state $|1$ apply $\mathbb{R}(\theta)$". The gate sequences are computed as matrix products, and the circuits.

The projective measures of a single qubit are provided by a hermitian 2×2 matrix P, such as $\mathrm{M}\ |1\rangle\langle 1| = \mathrm{diag}(0, 1)$; the complementary outcome is then $M^{\perp} = 1 - \mathrm{M}$. They are measured by metres in the circuit. Considering a quantum state $|\psi\rangle$, the post-measurement state is $M|\psi\rangle/p$/p with probability $p = \|M|\psi\|_2$. This is also the post-selection likelihood to ensure a measured result M; this likelihood may be extended close to 1 using $\sim \sqrt{1/p}$ rounds of amplitude amplification (Alaminos et al., 2022b; Grover, 2005).

The quantum recurrent neural networks within this proposal are all runnable on classical hardware in which the "hidden state" on n qubits is expressed by an array of size 2^n, and the set of parameters is provided by the collection of all parameterized quantum gates in the process, leading to matrices with parameterised inputs. To run a QRNN conventionally, we employ a series of matrix–vector multiplications for the gates, and matrix–vector multiplications with subsequent renormalisation of the status for norm 1 for the measure and post-select transactions. Running on quantum hardware, matrix multiplications are "free", and the hidden state in n qubits, which classically requires exponential memory, may be contained in ~ n qubits only.

Parametrized Quantum Gates

Quantum VQE circuits are very compact, meaning that they alternate single-qubit parameterised gates with entangled gates, such as controlled-no transactions. Hence, this offers the advantage of packing many parameters in a rather dense circuit.

Moreover, although these circuits are known to form a universal family, their high entanglement gate density, as well as the missing correlation between the parameters, results in very over-parameterised models which are difficult to train in sorting tasks for inputs of more than a few bits (Benedetti et al., 2019).

We build a highly structured parameterised quantum circuit in which a few parameters are reused again and again. It is mainly based on a new type of quantum neuron that spins its target lane following a non-linear activation function attached to the polynomials of its binary inputs. The cell consists of a composite of an input stage that, at every step, puts the actual input into the state of the cell. Multiple work steps follow which calculate the input and the cell state, plus a concluding output step that generates a density of probability on possible forecasts. The application of these QRNN cells in an iterative fashion over the input sequence in a recurrent model is very similar to traditional RNNs.

In training we implement quantum amplitude amplification (Guerreschi, 2019) on the output vias, to make sure we measure the right token of the training data at all steps. Although the measures are usually non-unitary operations, using the amplitude amplification step ensures that the measures while training remains as close to unitary as we want them to be.

A Higher-Degree Quantum Neuron

The power of classical neural networks arises from the implementation of non-linear activation factors to the related converse transformations in the layers of the network. Instead, because of the nature of quantum mechanics, any quantum circuit would inevitably be a linear operation.

Nevertheless, nonlinear behaviour does not happen anywhere in quantum mechanics: a simple example is a single-qubit gate R(θ) = exp(iYθ) for the Pauli matrix Y (Nielsen & Chuang, 2001), acting as a

$$R(\theta) = \exp\left(i\theta\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}\right) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \tag{90}$$

namely like a rotation within the two-dimensional space covered by the computational basis vectors of a single qubit, {|0, |1}. Meanwhile, the rotation matrix itself remains linear, we observe that the state amplitudes —cos θ and sin θ—depend non-linearly on the angle θ. Lifting the rotation to a checked operation cR(i, θ_i) conditional on the i^{th} qubit of a state $|x$ for $x \in \{0,1\}^n$, we obtain the map

$$\begin{aligned} &R\left(\theta_0\right)cR\left(1,\theta_1\right)\ldots cR\left(n,\theta_n\right)|x\rangle|0 = |x\rangle(cos(\eta)|0\rangle + sin(\eta)|1\rangle) \\ &\text{for} \quad \eta = \theta_0 + \sum_{i=1}^{n}\theta_i x_i \end{aligned} \tag{91}$$

Hence, this corresponds to a rotation by an infinite transformation of the basis vector $|x\rangle$ with $x = \left\{x_{1,\ldots,}x_n\right\} \in \{0,1\}^n$, by a parameter vector $\theta = \left(\theta_0, \theta_1, \ldots, \theta_n\right)$. The process is linearly expanded to the base and target state superpositions, and owing to the form of R(θ) all changes in amplitude just introduced are true-valued.

This transformation of the cosine of the amplitudes through a checked transaction is already non-linear; however, a sine function is not especially sharp, lacking also a sufficient "flat" region where the activation stays constant, as is the case of a linear rectifier unit. Cao et al. (2017) suggested an approach to implement a linear map into a set of qubits that produces amplitudes that exhibit these steeper slopes and plateaus, in a manner very similar to a sigmoidal activation function. The activation has a parameter of order ord ≥ 1 governing the tilt, the circuit resulting in the activation amplitude. This quantum neuron in pure states is rotated by an angle $f(\theta) = \arctan\left(tan(\theta)^{2ord}\right)$, where ord ≥ 1 is the order of the neuron. Assuming an affine transformation η for the input bitstring x_i as shown in formula (93), this rotation is translated into the amplitudes.

$$\cos\left(f(\eta)\right) = \frac{1}{\sqrt{1+\tan\left(\eta\right)^{2\times 2^{ord}}}} \quad \text{and} \quad \sin\left(f(\eta)\right) = \frac{\tan\left(\eta\right)^{2^{ord}}}{\sqrt{1+\tan\left(\eta\right)^{2\times 2^{ord}}}} \tag{92}$$

which arises by standardising the transform $|0\rangle \mapsto \cos\left(\theta\right)^{2^{ord}}|0\rangle + \sin\left(\theta\right)^{2^{ord}}|1\rangle$ as can be seen clearly. For ord = 1, the circuit is shown on the left; for ord = 2 on the right. Superior orders are recursively buildable.

A so-called repetition-to-success (RUS) circuit is this quantum neuron, indicating that the measured ring signals if the circuit has been performed correctly. If the result is zero, the neuron has been committed. A correction circuit returns the state to its original configuration when the result is one. Beginning with a pure state (e.g. $|x\rangle$ for $x \in \{0,1\}^2$ and recurring every time a 1 is measured, an arbitrarily high probability of success is reached.

Alas, for control in superposition, such as a state $|x\rangle + |y/\sqrt{2}$, this does not work for $x \neq y$ two bit-strings of length n. The amplitudes within the overlap, in this case, will rely on the success story. A technique called fixed-point oblique amplitude amplification (Tacchino et al., 2019), essentially post-selects in the measurement of result 0 while preserving the unitarity of the operation with arbitrary precision. There is the additional cost of multiple rounds of these quantum circuits, whose number will depend on the chance of a zero being measured in the first place. This depends obviously on the parameters of the neuron, θ, and the input state is given. We stress that by selecting sufficiently large individual post-selection probabilities, there is no exponential reduction in the overall probability of success across the number of quantum neurons employed.

We extend this quantum neuron in this paper with an increase in the number of check terms. More precisely, η as provided in formula (89) is an affine transform of the boolean vector $x = \left\{x_{1,\ldots,}x_n\right\}$ for $x_i \in \{0,1\}$. When we introduce multi-control gates—having their own parameterised rotation, labelled by a multi-index θ_I that varies depending on the qubits i ∈ I on which the gate conditions—we get the option of incorporating higher-degree polynomials, i.e.

$$\eta' = \theta_0 + \sum_{i=1}^{n} \theta_i x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} \theta_{ij} x_i x_j + \cdots = \sum_{\substack{I \subseteq [n] \\ |I| \leq d}} \theta_I \prod_{i \in I} x_i \tag{93}$$

being d the degree of the neuron; for d = 2 and n = 4 an example of a checked rotation that increase to this higher-order transformation η' on the bit string x_i. So, higher degree boolean logic operations could be directly encrypted inside a unique conditional rotation: an AND operation between two bits x_1 and x_2 is simply $x_1 x_2$.

QRNN Cell and Sequence to Sequence Model

The identified quantum neuron becomes the central component in building our quantum recurrent neural network cell. As for conventional RNNs and LSTMs, we provide such a cell to be applied successively to the input submitted to the network. In particular, the cell consists of input and output lanes that are restored following each step, plus a cell-internal state that is transmitted into the next iteration of the network.

To implement the constructed QRNN cell, we require an iterative application of the QRNN cell to a sequence of input words $in_{1,} in_{2,} \ldots, in_L$.

The outgoing lanes out_i label a discrete distribution measuring p_i over the class labels. The distribution could be entered into an assigned loss function, like cross-entropy or CTC loss.

Appendix 2: Results for the 'All Trades' Scenario

This supplement reports the results of the main forecasting techniques implemented to "all trades" rather than just those produced in the continuous trading situation.

See Tables 10, 11, 12, 13, 14, 15.

Table 10 Sign prediction ratio (10 min) for all trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.810	0.842	0.866	0.907	0.893	0.872	0.903	0.860	0.896	0.885
United States	0.804	0.840	0.860	0.903	0.887	0.866	0.896	0.859	0.888	0.880
Italy	0.800	0.839	0.854	0.897	0.879	0.862	0.889	0.854	0.884	0.880
Spain	0.794	0.837	0.851	0.889	0.873	0.858	0.887	0.848	0.880	0.876
Japan	0.791	0.836	0.846	0.886	0.868	0.850	0.883	0.841	0.879	0.869
Turkey	0.786	0.832	0.839	0.880	0.863	0.844	0.879	0.839	0.878	0.864
Mexico	0.783	0.829	0.839	0.875	0.859	0.839	0.873	0.836	0.870	0.860
Indonesia	0.775	0.827	0.835	0.870	0.851	0.838	0.869	0.836	0.864	0.855
Nigeria	0.771	0.826	0.829	0.870	0.849	0.837	0.866	0.831	0.862	0.850
Poland	0.766	0.818	0.823	0.863	0.845	0.830	0.863	0.827	0.853	0.848
Walmart	0.764	0.818	0.817	0.862	0.838	0.826	0.863	0.827	0.846	0.841
Johnson & Johnson	0.763	0.803	0.808	0.861	0.823	0.824	0.846	0.821	0.844	0.830
Verizon	0.771	0.808	0.815	0.863	0.832	0.824	0.851	0.831	0.857	0.845
Unilever PLC	0.786	0.825	0.831	0.863	0.846	0.841	0.866	0.845	0.873	0.849
Rio Tinto PLC	0.793	0.836	0.841	0.871	0.852	0.853	0.867	0.846	0.888	0.851
Air Liquide	0.796	0.842	0.848	0.876	0.858	0.866	0.883	0.847	0.898	0.860
Ambev	0.804	0.859	0.856	0.885	0.865	0.873	0.890	0.861	0.901	0.847
Cemex	0.802	0.848	0.833	0.863	0.850	0.853	0.878	0.852	0.890	0.844
Turkish Airlines	0.783	0.838	0.825	0.852	0.830	0.839	0.855	0.849	0.875	0.840
KCE Electronics	0.768	0.820	0.806	0.850	0.806	0.828	0.840	0.842	0.868	0.823
Telekomunikacja Polska	0.761	0.793	0.802	0.839	0.781	0.817	0.836	0.823	0.850	0.808
Caesars Resort Collection LLC	0.760	0.790	0.795	0.835	0.755	0.812	0.826	0.822	0.821	0.804
Asurion LLC	0.760	0.785	0.771	0.809	0.735	0.789	0.813	0.817	0.812	0.793
Intelsat Jackson Holdings	0.746	0.764	0.769	0.801	0.718	0.769	0.800	0.801	0.804	0.794

Table 10 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.730	0.756	0.746	0.794	0.696	0.767	0.781	0.782	0.788	0.769
Great outdoors Group llc	0.704	0.744	0.737	0.774	0.681	0.752	0.753	0.762	0.785	0.767
Petróleos Mexicanos	0.687	0.734	0.729	0.765	0.659	0.731	0.741	0.734	0.776	0.752
Petrobras	0.680	0.728	0.717	0.760	0.635	0.729	0.735	0.728	0.758	0.748
Sands China Ltd	0.674	0.715	0.695	0.738	0.614	0.722	0.712	0.700	0.735	0.746
Indonesia Asahan Alumini	0.656	0.693	0.694	0.714	0.597	0.718	0.690	0.678	0.726	0.723
Longfor Properties	0.635	0.683	0.693	0.709	0.590	0.708	0.680	0.663	0.720	0.720

Table 11 Ideal profit ratio (10 min) for all trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.0091	0.0106	0.0132	0.0139	0.0152	0.0118	0.0086	0.0097	0.0128	0.0170
United States	0.0091	0.0121	0.0134	0.0153	0.0157	0.0133	0.0102	0.0100	0.0145	0.0187
Italy	0.0099	0.0123	0.0152	0.0162	0.0164	0.0149	0.0116	0.0116	0.0159	0.0196
Spain	0.0109	0.0129	0.0159	0.0164	0.0184	0.0166	0.0136	0.0132	0.0168	0.0207
Japan	0.0116	0.0144	0.0165	0.0166	0.0186	0.0163	0.0143	0.0133	0.0166	0.0213
Turkey	0.0133	0.0151	0.0185	0.0178	0.0208	0.0175	0.0150	0.0146	0.0189	0.0220
Mexico	0.0129	0.0137	0.0183	0.0176	0.0201	0.0163	0.0141	0.0141	0.0174	0.0202
Indonesia	0.0129	0.0126	0.0176	0.0176	0.0191	0.0155	0.0132	0.0139	0.0158	0.0190
Nigeria	0.0115	0.0118	0.0175	0.0156	0.0174	0.0153	0.0120	0.0134	0.0157	0.0182
Poland	0.0109	0.0113	0.0169	0.0149	0.0156	0.0135	0.0118	0.0128	0.0149	0.0172
Walmart	0.0098	0.0093	0.0152	0.0134	0.0150	0.0122	0.0094	0.0121	0.0135	0.0157
Johnson & Johnson	0.0088	0.0078	0.0145	0.0115	0.0141	0.0117	0.0086	0.0102	0.0126	0.0153
Verizon	0.0082	0.0060	0.0134	0.0100	0.0130	0.0110	0.0075	0.0086	0.0110	0.0147
Unilever PLC	0.0080	0.0056	0.0130	0.0084	0.0112	0.0100	0.0053	0.0083	0.0108	0.0146
Rio Tinto PLC	0.0080	0.0043	0.0129	0.0078	0.0096	0.0091	0.0040	0.0081	0.0100	0.0137
Air Liquide	0.0100	0.0045	0.0147	0.0097	0.0101	0.0099	0.0048	0.0096	0.0107	0.0141
Ambev	0.0119	0.0063	0.0162	0.0107	0.0112	0.0111	0.0050	0.0112	0.0114	0.0147
Cemex	0.0131	0.0066	0.0174	0.0116	0.0122	0.0119	0.0053	0.0127	0.0130	0.0157
Turkish Airlines	0.0136	0.0081	0.0182	0.0134	0.0126	0.0125	0.0077	0.0141	0.0142	0.0175
KCE Electronics	0.0153	0.0084	0.0199	0.0147	0.0146	0.0140	0.0082	0.0148	0.0145	0.0182
Telekomunikacja Polska	0.0164	0.0097	0.0201	0.0158	0.0153	0.0160	0.0101	0.0162	0.0154	0.0201
Caesars Resort Collection LLC	0.0169	0.0098	0.0204	0.0161	0.0154	0.0163	0.0124	0.0166	0.0169	0.0216
Asurion LLC	0.0163	0.0080	0.0203	0.0160	0.0151	0.0161	0.0110	0.0158	0.0162	0.0210
Intelsat Jackson Holdings	0.0182	0.0075	0.0196	0.0160	0.0134	0.0152	0.0110	0.0150	0.0155	0.0194

Table 11 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.0172	0.0070	0.0183	0.0147	0.0133	0.0135	0.0106	0.0140	0.0148	0.0176
Great Outdoors Group llc	0.0157	0.0056	0.0175	0.0136	0.0132	0.0127	0.0105	0.0125	0.0139	0.0165
Petróleos Mexicanos	0.0145	0.0037	0.0159	0.0133	0.0126	0.0119	0.0091	0.0112	0.0137	0.0149
Petrobras	0.0127	0.0031	0.0148	0.0122	0.0110	0.0103	0.0091	0.0101	0.0134	0.0145
Sands China Ltd	0.0108	0.0026	0.0145	0.0114	0.0104	0.0101	0.0075	0.0085	0.0132	0.0143
Indonesia Asahan Alumini	0.0102	0.0013	0.0140	0.0094	0.0096	0.0087	0.0052	0.0078	0.0123	0.0131
Longfor Properties	0.0094	0.0010	0.0138	0.0081	0.0079	0.0077	0.0050	0.0059	0.0113	0.0118

Table 12 Sign prediction ratio (30 min) for all trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.793	0.824	0.848	0.888	0.874	0.853	0.884	0.842	0.877	0.892
United States	0.787	0.822	0.842	0.884	0.868	0.847	0.877	0.841	0.869	0.887
Italy	0.783	0.821	0.836	0.878	0.860	0.843	0.871	0.836	0.865	0.887
Spain	0.777	0.820	0.833	0.871	0.854	0.840	0.869	0.831	0.861	0.883
Japan	0.774	0.818	0.828	0.867	0.849	0.832	0.864	0.823	0.861	0.876
Turkey	0.769	0.814	0.822	0.861	0.845	0.826	0.861	0.821	0.859	0.871
Mexico	0.767	0.812	0.821	0.856	0.841	0.821	0.855	0.819	0.852	0.867
Indonesia	0.759	0.809	0.817	0.852	0.833	0.820	0.850	0.818	0.845	0.862
Nigeria	0.755	0.808	0.812	0.851	0.831	0.820	0.847	0.814	0.843	0.856
Poland	0.750	0.801	0.806	0.845	0.828	0.813	0.845	0.810	0.835	0.854
Walmart	0.748	0.801	0.799	0.844	0.820	0.809	0.844	0.809	0.828	0.848
Johnson & Johnson	0.747	0.786	0.791	0.843	0.805	0.806	0.828	0.803	0.827	0.836
Verizon	0.755	0.791	0.798	0.845	0.815	0.807	0.833	0.814	0.839	0.852
Unilever PLC	0.770	0.807	0.813	0.845	0.828	0.823	0.848	0.827	0.854	0.856
Rio Tinto PLC	0.776	0.818	0.823	0.853	0.834	0.835	0.848	0.828	0.869	0.858
Air Liquide	0.779	0.824	0.830	0.858	0.839	0.848	0.865	0.829	0.879	0.867
Ambev	0.787	0.841	0.838	0.867	0.847	0.855	0.871	0.842	0.882	0.853
Cemex	0.785	0.830	0.815	0.844	0.832	0.835	0.860	0.834	0.871	0.851
Turkish Airlines	0.767	0.821	0.808	0.834	0.813	0.821	0.837	0.831	0.857	0.846
KCE Electronics	0.752	0.802	0.789	0.833	0.789	0.811	0.822	0.824	0.849	0.829
Telekomunikacja Polska	0.745	0.777	0.785	0.821	0.764	0.800	0.818	0.806	0.832	0.815
Caesars Resort Collection LLC	0.744	0.773	0.779	0.817	0.739	0.795	0.809	0.805	0.804	0.810
Asurion LLC	0.744	0.768	0.755	0.792	0.719	0.772	0.796	0.800	0.795	0.799
Intelsat Jackson Holdings	0.730	0.748	0.753	0.784	0.703	0.753	0.783	0.784	0.788	0.779

Table 12 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.715	0.740	0.730	0.777	0.681	0.751	0.764	0.766	0.771	0.779
Great Outdoors Group llc	0.689	0.728	0.721	0.757	0.667	0.736	0.737	0.746	0.768	0.777
Petróleos Mexicanos	0.672	0.719	0.714	0.749	0.645	0.716	0.725	0.718	0.760	0.762
Petrobras	0.666	0.713	0.702	0.743	0.622	0.714	0.719	0.712	0.742	0.758
Sands China Ltd	0.660	0.700	0.680	0.722	0.601	0.706	0.697	0.685	0.720	0.756
Indonesia Asahan Alumini	0.642	0.678	0.679	0.698	0.584	0.703	0.675	0.664	0.711	0.733
Longfor Properties	0.622	0.668	0.678	0.694	0.578	0.694	0.665	0.649	0.705	0.729

Table 13 Ideal profit ratio (30 min) for all trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-SVM	GRU-CNN	QRNN
Germany	0.0070	0.0090	0.0092	0.0108	0.0110	0.0098	0.0086	0.0055	0.0099	0.0118
United States	0.0088	0.0098	0.0101	0.0116	0.0114	0.0110	0.0094	0.0069	0.0110	0.0137
Italy	0.0098	0.0114	0.0121	0.0122	0.0126	0.0114	0.0110	0.0073	0.0117	0.0157
Spain	0.0100	0.0119	0.0127	0.0134	0.0139	0.0117	0.0115	0.0092	0.0125	0.0174
Japan	0.0098	0.0128	0.0132	0.0138	0.0149	0.0122	0.0132	0.0109	0.0141	0.0174
Turkey	0.0115	0.0141	0.0152	0.0160	0.0160	0.0135	0.0143	0.0130	0.0155	0.0189
Mexico	0.0104	0.0125	0.0137	0.0153	0.0151	0.0121	0.0137	0.0118	0.0139	0.0186
Indonesia	0.0092	0.0109	0.0136	0.0146	0.0151	0.0104	0.0120	0.0098	0.0138	0.0172
Nigeria	0.0081	0.0097	0.0129	0.0137	0.0148	0.0086	0.0120	0.0084	0.0130	0.0169
Poland	0.0074	0.0094	0.0123	0.0122	0.0144	0.0069	0.0111	0.0067	0.0124	0.0154
Walmart	0.0069	0.0084	0.0108	0.0114	0.0130	0.0061	0.0096	0.0059	0.0120	0.0136
Johnson & Johnson	0.0051	0.0071	0.0106	0.0101	0.0124	0.0053	0.0078	0.0056	0.0111	0.0133
Verizon	0.0071	0.0086	0.0111	0.0118	0.0134	0.0063	0.0099	0.0061	0.0124	0.0140
Unilever PLC	0.0053	0.0074	0.0109	0.0104	0.0127	0.0054	0.0080	0.0057	0.0114	0.0137
Rio Tinto PLC	0.0052	0.0071	0.0098	0.0098	0.0122	0.0042	0.0061	0.0037	0.0099	0.0119
Air Liquide	0.0052	0.0052	0.0079	0.0081	0.0116	0.0025	0.0057	0.0020	0.0089	0.0100
Ambev	0.0048	0.0042	0.0076	0.0060	0.0101	0.0023	0.0041	0.0010	0.0069	0.0083
Cemex	0.0052	0.0046	0.0089	0.0079	0.0105	0.0031	0.0055	0.0018	0.0071	0.0097
Turkish Airlines	0.0056	0.0050	0.0103	0.0086	0.0109	0.0050	0.0070	0.0035	0.0090	0.0116
KCE Electronics	0.0061	0.0055	0.0104	0.0098	0.0110	0.0056	0.0083	0.0037	0.0090	0.0136
Telekomunikacja Polska	0.0064	0.0061	0.0114	0.0105	0.0123	0.0067	0.0106	0.0052	0.0097	0.0143
Caesars Resort Collection LLC	0.0069	0.0070	0.0122	0.0121	0.0130	0.0068	0.0126	0.0072	0.0106	0.0158
Asurion LLC	0.0088	0.0071	0.0132	0.0125	0.0134	0.0079	0.0130	0.0081	0.0112	0.0171
Intelsat Jackson Holdings	0.0092	0.0084	0.0138	0.0135	0.0150	0.0090	0.0148	0.0094	0.0127	0.0178

Table 13 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-SVM	GRU-CNN	QRNN
Athenahealth Group	0.0076	0.0063	0.0138	0.0132	0.0136	0.0084	0.0147	0.0091	0.0110	0.0164
Great Outdoors Group llc	0.0083	0.0054	0.0120	0.0130	0.0135	0.0067	0.0128	0.0086	0.0094	0.0147
Petróleos Mexicanos	0.0069	0.0044	0.0119	0.0117	0.0126	0.0059	0.0118	0.0069	0.0088	0.0142
Petrobras	0.0050	0.0040	0.0108	0.0111	0.0108	0.0040	0.0110	0.0055	0.0073	0.0139
Sands China Ltd	0.0040	0.0022	0.0100	0.0101	0.0092	0.0029	0.0104	0.0043	0.0072	0.0123
Indonesia Asahan Alumini	0.0021	0.0028	0.0087	0.0082	0.0080	0.0017	0.0097	0.0038	0.0070	0.0118
Longfor Properties	0.0016	0.0010	0.0071	0.0069	0.0061	0.0016	0.0077	0.0019	0.0069	0.0098

Table 14 Sign prediction ratio (60 min) for all trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.773	0.804	0.827	0.866	0.908	0.832	0.892	0.865	0.884	0.909
United States	0.768	0.802	0.821	0.862	0.902	0.826	0.885	0.864	0.877	0.903
Italy	0.764	0.801	0.816	0.856	0.894	0.823	0.878	0.859	0.872	0.903
Spain	0.758	0.799	0.812	0.849	0.888	0.819	0.876	0.853	0.869	0.899
Japan	0.755	0.798	0.808	0.846	0.883	0.811	0.872	0.845	0.868	0.891
Turkey	0.750	0.794	0.801	0.840	0.878	0.806	0.869	0.843	0.867	0.886
Mexico	0.748	0.792	0.801	0.835	0.874	0.801	0.863	0.841	0.859	0.882
Indonesia	0.740	0.789	0.797	0.831	0.866	0.800	0.858	0.840	0.853	0.878
Nigeria	0.736	0.788	0.792	0.830	0.864	0.800	0.855	0.836	0.851	0.872
Poland	0.732	0.781	0.786	0.824	0.860	0.792	0.853	0.832	0.842	0.870
Walmart	0.729	0.781	0.780	0.823	0.852	0.789	0.852	0.831	0.835	0.863
Johnson & Johnson	0.728	0.767	0.772	0.822	0.837	0.786	0.836	0.825	0.834	0.851
Verizon	0.736	0.772	0.778	0.824	0.847	0.787	0.840	0.836	0.846	0.868
Unilever PLC	0.751	0.787	0.793	0.824	0.861	0.803	0.855	0.849	0.862	0.872
Rio Tinto PLC	0.757	0.798	0.803	0.832	0.867	0.815	0.856	0.850	0.877	0.873
Air Liquide	0.759	0.804	0.810	0.837	0.872	0.827	0.873	0.851	0.886	0.882
Ambev	0.768	0.820	0.817	0.845	0.880	0.833	0.879	0.865	0.890	0.869
Cemex	0.766	0.810	0.795	0.824	0.865	0.814	0.868	0.856	0.878	0.866
Turkish Airlines	0.748	0.800	0.788	0.814	0.845	0.801	0.845	0.854	0.864	0.862
KCE Electronics	0.734	0.783	0.770	0.812	0.820	0.791	0.830	0.846	0.857	0.844
Telekomunikacja Polska	0.727	0.757	0.766	0.801	0.794	0.780	0.825	0.828	0.839	0.829
Caesars Resort Collection LLC	0.726	0.754	0.759	0.797	0.768	0.775	0.816	0.826	0.811	0.825
Asurion LLC	0.725	0.749	0.736	0.773	0.797	0.753	0.803	0.822	0.801	0.814
Intelsat Jackson Holdings	0.712	0.730	0.734	0.764	0.778	0.735	0.790	0.806	0.794	0.793

Table 14 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.697	0.721	0.712	0.758	0.754	0.733	0.771	0.786	0.778	0.768
Great Outdoors Group llc	0.672	0.710	0.703	0.739	0.738	0.718	0.743	0.766	0.775	0.766
Petróleos Mexicanos	0.655	0.701	0.696	0.730	0.714	0.698	0.731	0.738	0.766	0.752
Petrobras	0.650	0.695	0.685	0.725	0.734	0.696	0.726	0.732	0.748	0.747
Sands China Ltd	0.644	0.683	0.663	0.704	0.709	0.689	0.704	0.704	0.726	0.745
Indonesia Asahan Alumini	0.626	0.661	0.662	0.681	0.689	0.685	0.681	0.682	0.717	0.723
Longfor Properties	0.606	0.652	0.661	0.677	0.681	0.676	0.671	0.666	0.711	0.719

Table 15 Ideal profit ratio (60 min) for all trades

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Germany	0.0114	0.0066	0.0064	0.0085	0.0093	0.0050	0.0051	0.0037	0.0068	0.0082
United States	0.0129	0.0070	0.0068	0.0103	0.0111	0.0053	0.0051	0.0042	0.0061	0.0074
Italy	0.0135	0.0079	0.0072	0.0104	0.0111	0.0057	0.0055	0.0058	0.0065	0.0079
Spain	0.0143	0.0072	0.0075	0.0107	0.0123	0.0076	0.0067	0.0061	0.0074	0.0090
Japan	0.0157	0.0072	0.0080	0.0112	0.0128	0.0093	0.0083	0.0075	0.0083	0.0100
Turkey	0.0160	0.0082	0.0087	0.0117	0.0144	0.0111	0.0099	0.0076	0.0099	0.0120
Mexico	0.0150	0.0078	0.0084	0.0114	0.0125	0.0104	0.0097	0.0074	0.0077	0.0093
Indonesia	0.0130	0.0071	0.0080	0.0107	0.0123	0.0100	0.0086	0.0065	0.0074	0.0090
Nigeria	0.0127	0.0062	0.0073	0.0091	0.0123	0.0096	0.0072	0.0051	0.0071	0.0087
Poland	0.0109	0.0058	0.0070	0.0090	0.0121	0.0075	0.0052	0.0043	0.0066	0.0080
Walmart	0.0105	0.0051	0.0061	0.0073	0.0111	0.0061	0.0051	0.0042	0.0060	0.0073
Johnson & Johnson	0.0099	0.0047	0.0056	0.0055	0.0096	0.0058	0.0028	0.0034	0.0047	0.0057
Verizon	0.0091	0.0049	0.0056	0.0057	0.0090	0.0050	0.0022	0.0019	0.0041	0.0050
Unilever PLC	0.0069	0.0040	0.0046	0.0048	0.0083	0.0047	0.0017	0.0002	0.0028	0.0035
Rio Tinto PLC	0.0049	0.0036	0.0040	0.0048	0.0068	0.0045	0.0011	0.0008	0.0027	0.0033
Air Liquide	0.0050	0.0047	0.0036	0.0067	0.0090	0.0048	0.0036	0.0005	0.0034	0.0042
Ambev	0.0052	0.0050	0.0035	0.0070	0.0105	0.0058	0.0054	0.0009	0.0049	0.0059
Cemex	0.0057	0.0056	0.0024	0.0078	0.0109	0.0076	0.0069	0.0016	0.0070	0.0085
Turkish Airlines	0.0067	0.0060	0.0015	0.0086	0.0117	0.0093	0.0075	0.0029	0.0079	0.0096
KCE Electronics	0.0073	0.0071	0.0008	0.0089	0.0129	0.0106	0.0100	0.0033	0.0079	0.0096
Telekomunikacja Polska	0.0092	0.0079	0.0007	0.0100	0.0144	0.0110	0.0112	0.0049	0.0084	0.0102
Caesars Resort Collection LLC	0.0112	0.0090	0.0011	0.0116	0.0154	0.0114	0.0096	0.0033	0.0108	0.0131
Asurion LLC	0.0103	0.0084	0.0007	0.0116	0.0154	0.0104	0.0095	0.0033	0.0085	0.0104
Intelsat Jackson Holdings	0.0119	0.0076	0.0014	0.0115	0.0134	0.0090	0.0075	0.0032	0.0077	0.0094

Table 15 (continued)

	Qfuzzy	AdaBoost-GA	SVM-GA	DLNN-GA	QGA	ANFIS-QGA	DRCNN	CNN-LSTM	GRU-CNN	QRNN
Athenahealth Group	0.0103	0.0065	0.0022	0.0107	0.0116	0.0082	0.0050	0.0018	0.0056	0.0068
Great Outdoors Group llc	0.0082	0.0064	0.0026	0.0104	0.0109	0.0080	0.0025	0.0015	0.0039	0.0047
Petróleos Mexicanos	0.0063	0.0058	0.0026	0.0098	0.0094	0.0068	0.0017	0.0054	0.0080	0.0086
Petrobras	0.0046	0.0056	0.0032	0.0077	0.0074	0.0065	0.0017	0.0061	0.0057	0.0060
Sands China Ltd	0.0043	0.0048	0.0035	0.0075	0.0052	0.0052	0.0003	0.0066	0.0057	0.0058
Indonesia Asahan Alumini	0.0031	0.0042	0.0037	0.0057	0.0049	0.0038	0.0019	0.0077	0.0056	0.0057
Longfor Properties	0.0026	0.0032	0.0039	0.0049	0.0037	0.0035	0.0020	0.0085	0.0075	0.0079

Appendix 3: Features About Every Bonds Used in the Sample

Bond Issuer	ISIN code	Issue Date	Maturity	Coupon (%)	Volume Issued (in USD)	Volume Executed (in USD)
Germany	DE0001141836	07/04/2010	07/04/2042	3.250	27,458,500,000	2,532,011,575
	DE0001102317	05/15/2013	05/15/2023	1.500	19,770,120,000	1,695,369,335
	DE0001102325	09/11/2013	08/15/2023	2.000	19,770,120,000	1,647,715,293
	DE0001102333	01/29/2014	02/15/2024	1.750	19,770,120,000	1,813,871,000
	DE0001104875	08/15/2017	08/15/2048	1.250	21,417,630,000	2,111,752,498
	DE0001141794	01/23/2019	04/05/2024	0.000	23,065,140,000	2,131,391,960
	DE0001102358	05/21/2014	05/15/2024	1.500	24,712,650,000	2,064,163,436
	DE0001104883	05/17/2022	06/14/2024	0.200	18,671,780,000	1,520,070,072
	DE0001102366	09/10/2014	08/15/2024	1.000	19,648,260,000	1,509,855,113
	DE0001135366	07/23/2008	07/04/2040	4.750	17,573,440,000	1,447,789,640
United States	US9128283W81	2/15/2018	2/15/2028	2.750	70,572,104,500	5,499,761,156
	US912828P469	02/15/2016	02/15/2026	1.625	64,940,659,900	4,700,899,049
	US9128285W63	01/15/2019	01/15/2029	1.032	13,000,026,300	1,100,035,194
	US912810SG40	02/15/2019	02/15/2049	1.181	15,385,015,600	1,449,539,188
	US91282CDD02	10/31/2021	10/31/2023	0.375	66,099,908,400	4,592,193,508
	US9128285C00	09/30/2018	09/30/2025	3.000	31,000,000,000	2,243,332,249
	US912810QP66	02/15/2011	02/15/2041	2.884	23,984,657,100	1,704,139,189
	US912810QF84	02/15/2010	02/15/2040	2.922	15,171,280,100	1,148,366,429
	US912810RY64	08/15/2017	08/15/2047	2.750	43,512,330,700	3,776,592,369
	US912810QX90	08/15/2012	08/15/2042	2.750	41,995,432,300	3,414,186,998
	US912810QW18	05/15/2012	05/15/2042	3.000	43,918,685,600	3,472,679,035
	US912810PX00	05/15/2008	05/15/2038	4.500	25,500,122,800	1,873,596,980
Italy	IT0005436693	02/01/2021	08/01/2031	0.600	21,300,000,000	1,545,041,693
	IT0005240350	09/01/2016	09/01/2033	2.450	16,808,228,000	1,379,642,560
	IT0005210650	08/01/2016	12/01/2026	1.250	18,891,843,000	1,374,124,929
	IT0005127086	08/01/2015	12/01/2025	2.000	19,427,596,000	1,262,818,368
	IT0004953417	08/01/2013	03/01/2024	4.500	23,264,571,000	1,450,240,610
	IT0005321325	09/01/2017	09/01/2038	2.950	14,963,750,000	1,141,990,363
	IT0004644735	09/01/2010	03/01/2026	4.500	21,999,898,000	1,321,215,545
	IT0005327306	03/15/2018	05/15/2025	1.450	15,419,125,000	965,491,378
	IT0005090318	03/02/2015	06/01/2025	1.500	19,786,723,000	1,195,137,809
	IT0005386245	10/01/2019	02/01/2025	0.350	19,468,306,000	1,233,633,129

Bond Issuer	ISIN code	Issue Date	Maturity	Coupon (%)	Volume Issued (in USD)	Volume Executed (in USD)
Spain	ES00000128H5	07/26/2016	10/31/2026	1.300	26,346,638,000	1,555,103,387
	ES00000127G9	06/09/2015	10/31/2025	2.150	25,740,540,000	1,421,847,795
	ES0000012B88	07/03/2018	07/30/2028	1.400	23,365,049,000	1,483,202,516
	ES0000012B70	11/30/2017	11/30/2023	0.175	5,456,261,000	285,222,503
	ES00000127Z9	01/19/2016	04/30/2026	1.950	22,952,139,000	1,252,560,660
	ES00000124C5	07/16/2013	10/31/2028	5.150	18,769,067,000	1,126,497,346
	ES00000126A4	11/30/2013	11/30/2024	2.140	13,170,578,000	630,704,030
	ES00000120N0	06/20/2007	07/30/2040	4.900	20,669,793,000	1,448,690,669
	ES00000127C8	11/30/2014	11/30/2030	1.186	16,836,799,000	1,106,295,472
Japan	JP1103351E98	9/22/2014	9/20/2024	0.500	58,407,605,995	3,114,395,372
	JP1103291D68	6/20/2013	6/20/2023	0.800	56,783,915,871	2,760,761,340
	JP1103551K72	6/20/2019	6/20/2029	0.100	51,040,754,700.00	3,341,038,531.12
	JP1201551FC0	12/20/2015	12/20/2035	1.000	33,483,028,176	2,594,309,168
	JP1200881660	6/20/2006	6/20/2026	2.300	18,164,193,301	1,080,066,060
	JP1200871653	3/20/2006	3/20/2026	2.200	8,161,464,243	450,517,545
	JP1103451GC0	12/20/2016	12/20/2026	0.100	65,843,010,168	3,791,204,304
	JP1103341E67	6/20/2014	6/20/2024	0.600	53,826,367,974	2,567,397,676
	JP1103301D90	9/20/2013	9/20/2023	0.800	54,398,599,836	1,770,760,994
	JP1200631388	6/20/2003	6/20/2023	1.800	6,943,752,926	183,727,606
	JP1103391F65	6/20/2015	6/20/2025	0.400	58,932,408,728	2,240,969,006
Turkey	XS1843443356	01/31/2019	03/31/2025	4.625	1,372,925,000	49,902,475
	XS1909184753	11/14/2018	02/16/2026	5.200	1,649,992,500	63,873,111
	US900123CF53	01/29/2014	03/22/2024	5.750	2,500,000,000	80,067,646
	XS1629918415	06/17/2017	06/14/2025	3.250	1,101,885,000	36,815,007
	US900123CP36	01/17/2018	02/17/2028	5.125	2,000,000,000	87,130,939
	US900123CW86	11/14/2019	11/14/2024	5.600	2,500,000,000	56,788,080
	US900123AW05	01/24/2005	05/02/2025	7.325	3,250,000,000	86,709,767
	US900123AY60	01/17/2006	03/17/2036	6.875	2,750,000,000	130,954,080
	US900123CB40	04/16/2013	04/16/2043	4.875	3,000,000,000	174,902,998
	US900123CM05	05/11/2017	05/11/2047	5.750	3,500,000,000	206,898,779
Mexico	US91087BAG59	7/31/2019	1/31/2050	4.500	2,903,527,000	201,705,721
	US91087BAK61	4/27/2020	4/27/2032	4.750	2,500,000,000	133,551,452
	MX0MGO000102	2/23/2017	11/07/2047	8.000	258,121,595	16,530,175
	MX0MGO000078	12/30/2004	12/05/2024	10.000	240,194,562	10,264,518
	MX0MGO0000P2	6/23/2011	5/29/2031	7.750	437,969,452	19,781,815
	US91086QBE70	01/21/2014	01/21/2045	5.550	3,000,000,000	193,969,256
	XS2135361686	09/18/2020	09/18/2027	1.350	750,000,000	41,896,431
	US91087BAJ98	04/27/2020	04/27/2025	3.900	1,000,000,000	47,779,374
	US91087BAD29	10/10/2017	02/10/2048	4.600	2,525,274,000	159,228,149
	US91086QAS75	09/27/2004	09/27/2034	6.750	4,266,566,000	254,190,320

Bond Issuer	ISIN code	Issue Date	Maturity	Coupon (%)	Volume Issued (in USD)	Volume Executed (in USD)
Indonesia	US455780CT15	04/15/2020	10/15/2050	4.200	1,650,000,000	123,440,663
	USY20721BE87	04/15/2013	04/15/2043	4.625	1,500,000,000	90,206,635
	US455780CR58	01/14/2020	02/14/2050	3.500	800,000,000	63,291,522
	XS2012546714	06/13/2019	09/18/2026	1.450	817,590,000	40,398,068
	USY20721BU20	07/18/2017	07/14/2047	4.750	1,000,000,000	77,217,563
	US455780CD62	12/11/2017	01/11/2028	3.500	1,250,000,000	77,888,231
	USY20721BT56	07/18/2017	07/18/2027	3.850	1,000,000,000	56,096,003
	US455780CY00	07/28/2021	07/28/2031	2.150	600,000,000	39,203,276
	USY20721BQ18	12/08/2016	01/08/2027	4.350	1,250,000,000	77,979,617
	US455780CQ75	01/14/2020	02/14/2030	2.850	1,200,000,000	81,257,678
Nigeria	XS1717013095	11/28/2017	11/28/2047	7.625	1,500,000,000	87,347,474
	XS1777972941	02/23/2018	02/23/2038	7.696	1,250,000,000	66,002,475
	XS1717011982	11/28/2017	11/28/2027	6.500	1,500,000,000	44,900,152
	XS0944707222	07/12/2013	07/12/2023	6.375	500,000,000	10,540,924
	XS2384698994	09/28/2021	09/28/2028	6.125	1,250,000,000	42,094,445
	XS1777972511	02/23/2018	02/23/2030	7.143	1,250,000,000	45,435,632
	XS1566179039	02/16/2017	02/16/2032	7.875	1,000,000,000	35,545,324
	XS1910826996	11/21/2018	11/21/2025	7.625	1,118,352,000	28,394,300
	XS2384704800	09/28/2021	09/28/2051	8.250	1,250,000,000	75,140,475
	XS2384701020	09/28/2021	09/28/2033	7.375	1,250,000,000	62,662,368
Poland	XS2114767457	02/10/2020	02/10/2025	0.000	1,647,510,000	34,734,468
	XS0224427160	07/20/2005	07/20/2055	4.250	519,208,000	22,189,504
	XS1288467605	09/09/2015	09/09/2025	1.500	1,091,570,000	25,934,254
	XS1015428821	01/15/2014	01/15/2024	3.000	2,196,680,000	44,408,386
	XS1508566392	10/25/2016	10/25/2028	1.000	818,677,500	21,003,581
	XS1584894650	03/23/2017	10/22/2027	1.375	1,091,570,000	25,899,231
	XS1766612672	02/07/2018	08/07/2028	1.125	1,090,420,000	25,933,439
	US857524AC63	01/22/2014	01/22/2024	4.000	2,000,000,000	41,690,796
	XS1346201889	01/18/2016	01/18/2036	2.375	2,183,140,000	82,782,859
	XS1960361720	03/07/2019	03/08/2043	2.000	545,785,000	27,703,475
Walmart	US931142CH46	04/05/2007	04/05/2027	5.875	750,000,000	99,404,779
	US931142BF98	2/15/2000	2/15/2030	7.550	1,000,000,000	100,362,707
	US931142CB75	8/31/2005	09/01/2035	5.250	2,500,000,000	277,573,653
	US931142CK74	8/24/2007	8/15/2037	6.500	2,250,000,000	268,070,264
	US931142CM31	4/15/2008	4/15/2038	6.200	1,500,000,000	178,712,981
	US931142AU74	10/14/1993	10/15/2023	6.750	250,000,000	7,729,944
Johnson & Johnson	US478160CM48	11/10/2017	1/15/2048	3.500	750,000,000	94,421,361
	US478160AL82	5/22/2003	5/15/2033	4.950	500,000,000	31,020,881
	US478160AN49	8/16/2007	8/15/2037	5.950	1,000,000,000	82,177,051
	US478160AT19	6/23/2008	7/15/2038	5.850	700,000,000	58,192,503
	US478160AV64	8/17/2010	09/01/2040	4.500	550,000,000	50,015,272
	US478160BA19	5/20/2011	5/15/2041	4.850	300,000,000	29,396,442

Bond Issuer	ISIN code	Issue Date	Maturity	Coupon (%)	Volume Issued (in USD)	Volume Executed (in USD)
Verizon	XS1405769727	11/02/2016	11/02/2035	3.125	604,125,000	60,766,843
	US92343VFE92	3/20/2020	3/22/2030	3.150	1,500,000,000	127,097,596
	US92343VFU35	11/20/2020	11/20/2050	2.875	2,750,000,000	269,487,826
	AU3CB0246221	8/17/2017	2/17/2025	4.050	301,603,525	24,454,081
	US92343VDU52	3/16/2017	3/16/2037	5.250	3,000,000,000	420,910,769
	AU3CB0268142	11/06/2019	05/06/2026	2.100	304,182,753	28,351,338
Unilever PLC	XS2008925344	06/11/2019	06/11/2039	1.500	709,520,500	88,993,363
	XS1684780031	9/15/2017	9/15/2024	1.375	335,625,000	23,428,648
	XS2008921277	06/11/2019	7/22/2026	1.500	671,250,000	42,497,268
	XS1684780205	9/15/2017	9/15/2029	1.875	335,625,000	26,178,357
	XS2008925344	06/11/2019	06/11/2039	1.500	697,398,000	78,171,947
Rio Tinto PLC	US767201AT32	11/02/2021	11/02/2051	2.750	1,250,000,000	155,997,716
	US767201AD89	6/27/2008	7/15/2028	7.125	750,000,000	60,751,202
	US767201AL06	11/02/2010	11/02/2040	5.200	500,000,000	52,837,019
	XS0863127279	12/11/2012	12/11/2024	2.875	457,678,278	22,966,598
	XS0863076930	12/11/2012	12/11/2029	4.000	671,250,000	42,099,681
	US013716AW59	31/05/2005	01/06/2035	5.750	300,000,000	19,457,646
Air Liquide	FR0011951771	06/05/2014	06/05/2024	1.875	545,785,000	37,902,233
	FR0011439835	03/06/2013	09/06/2023	2.375	327,471,000	18,981,136
	FR0014005HY8	9/20/2021	9/20/2033	0.375	538,457,000	48,757,525
	FR0013182839	6/13/2016	6/13/2024	0.750	537,289,000	33,483,220
	FR0013428067	6/20/2019	6/20/2030	0.625	654,942,000	57,805,408
	FR0013182847	6/13/2016	6/13/2028	1.250	1,091,570,000	82,578,453
	FR0013241346	03/08/2017	03/08/2027	1.000	661,824,000	46,333,309
	FR0012766889	06/03/2015	06/03/2025	1.250	542,643,000	33,326,475
	FR0013505559	04/02/2020	04/02/2025	1.000	539,271,000	24,767,777
Ambev	LU0000870137	4/21/2023	3/31/2027	10.625	41,352,000	4,388,593
	US20441XAB82	10/30/2002	12/15/2011	10.500	497,432,000	10,637,773
	USP30580AA55	12/15/2011	12/15/2019	13.200	209,796,000	6,547,211
	LU1234567890	2/15/2016	2/15/2021	9.750	59,793,000	2,661,324
Cemex	US151290BZ57	01/11/2021	01/11/2031	3.875	1,107,769,000	123,496,287
	US151290BX00	9/17/2020	9/17/2030	5.200	717,384,000	60,778,504
	US151290BW27	06/05/2020	06/05/2027	7.375	935,879,000	65,878,868
	US151290BV44	11/19/2019	11/19/2029	5.450	753,053,000	55,026,216
	US766879AA85	04/01/2003	7/21/2025	7.700	149,897,000	7,485,139
Turkish Airlines	US10010YAA01	03/15/2015	03/15/2027	4.200	328,274,000	31,979,536
	USU0567PAA40	03/04/2013	03/04/2025	7.500	750,000,000	57,689,650
KCE Electronics	US48245B1098	8/21/2017	8/21/2022	8.250	800,000,000	40,990,255
	XS2180488111	06/04/2020	12/31/2041	9.683	304,556,000	34,738,793
	XS1265917481	08/03/2015	12/31/2030	8.575	290,000,000	24,240,151

Bond Issuer	ISIN code	Issue Date	Maturity	Coupon (%)	Volume Issued (in USD)	Volume Executed (in USD)
Telekomunikacja Polska	XS1585599667	01/08/2016	01/08/2024	1.875	754,117,000	69,232,737
	XS1763026934	05/19/2017	05/19/2024	1.250	532,928,000	42,621,102
Caesars Resort Collection LLC	USU1230PAB77	02/06/2023	2/15/2030	7.000	2,000,000,000	206,375,311
	USU1230PAA94	10/15/2022	10/15/2029	4.625	1,200,000,000	95,097,735
	USU2829LAD74	07/01/2020	07/01/2027	8.125	1,800,000,000	119,510,985
	US28470RAJ14	07/01/2018	07/01/2025	6.250	3,400,000,000	130,617,993
Asurion LLC	US04649VAX82	12/23/2018	12/23/2026	8.750	3,510,356	258,496
	US04649VAW00	11/03/2016	11/03/2024	7.500	150,000,000	8,370,686
	US04649VAQ32	07/08/2014	07/08/2020	6.975	100,000,000	3,347,416
	US04649VAG59	05/04/2012	05/04/2019	7.250	120,000,000	2,795,915
Intelsat Jackson Holdings	US45824TAR68	03/29/2016	02/15/2024	8.000	1,349,678,000	119,432,338
	US45824TAM71	09/15/2018	09/15/2022	6.625	1,275,000,000	41,601,937
	US45824TAS42	09/30/2014	09/30/2022	9.500	490,000,000	20,561,418
	US45824TAP03	06/05/2013	08/01/2023	5.500	2,000,000,000	89,294,027
	USL5137XAT65	9/19/2018	10/15/2024	8.500	2,950,000,000	170,262,952
	USL5137XAJ83	3/29/2016	2/15/2024	8.000	1,250,000,000	686,303,258
Athenahealth Group	US60337JAA43	02/15/2022	02/15/2030	6.500	2,350,000,000	196,740,827
	US04686RAB96	02/15/2021	02/15/2029	6.500	5,900,000,000	481,150,076
	US04686RAC79	09/15/2021	09/15/2029	6.500	1,000,000,000	69,922,650
Great Outdoors Group llc	US07014QAN16	03/06/2018	03/06/2028	8.380	200,000,000	18,961,194
	US07014QAM33	03/05/2022	03/05/2028	7.250	250,000,000	20,901,121
	US07014QAJ04	03/25/2014	03/25/2019	7.500	150,000,000	7,220,059
	US07014QAG64	06/05/2015	06/05/2020	6.950	300,000,000	17,237,389
	US07014QAK76	09/25/2019	09/25/2024	8.450	200,000,000	16,713,977
Petróleos Mexicanos	US706451BD26	9/15/2004	9/15/2027	9.500	162,425,000	11,355,157
	US706451BR12	06/04/2008	6/15/2038	6.625	491,175,000	40,759,059
	US706451BG56	12/15/2005	6/15/2035	6.625	2,749,000,000	202,429,964
	XS0213101073	2/24/2005	2/24/2025	5.500	1,098,340,000	70,566,416
	US71654QCB68	02/04/2016	08/04/2026	6.875	2,969,774,000	179,974,227
	US71654QDN97	02/08/2023	02/08/2033	10.000	2,000,000,000	177,155,911
	US71643VAB18	12/16/2021	2/16/2032	6.700	6,779,842,000	502,568,278
	USP78625DC49	9/26/2014	09/12/2024	7.190	2,612,203,780	133,945,121

Bond Issuer	ISIN code	Issue Date	Maturity	Coupon (%)	Volume Issued (in USD)	Volume Executed (in USD)
Petrobras	XS0718502007	12/12/2011	12/14/2026	6.250	658,493,565	49,036,883
	US71645WAS08	1/27/2011	1/27/2041	6.750	2,250,000,000	241,054,565
	US71645WAQ42	10/30/2009	1/20/2040	6.875	1,500,000,000	139,721,044
	US71647NAM11	03/17/2014	03/17/2024	6.250	559,315,000	33,895,621
	XS0982711714	01/14/2014	01/14/2025	4.750	3,096,714,713	17,648,749
	US71647NAT63	09/27/2017	01/27/2025	5.299	633,300,000	34,370,495
	US71647NAQ25	05/23/2016	05/23/2026	8.750	386,934,000	16,232,224
	US71647NAS80	01/17/2017	01/17/2027	7.375	710,066,000	47,443,714
	US71645WAQ42	10/30/2009	01/20/2040	6.875	728,963,000	62,530,497
Sands China Ltd	US80007RAE53	08/09/2018	08/08/2028	5.900	1,892,760,000	136,711,791
	US80007RAL96	06/04/2020	6/18/2030	4.375	697,375,000	55,222,959
	USG7801RAD10	06/04/2020	6/18/2030	4.375	700,000,000	52,121,268
	US80007RAK14	06/04/2020	01/08/2026	4.300	796,938,000	34,972,824
	US80007RAF29	08/09/2018	08/08/2025	5.625	1,786,475,000	76,954,319
	USG7801RAB53	08/09/2018	08/08/2025	5.125	1,800,000,000	35,750,190
	USG7801RAE92	01/08/2019	01/08/2026	3.800	800,000,000	23,243,316
	US80007RAQ83	03/08/2021	03/08/2029	2.850	649,621,000	27,246,119
Indonesia Asahan Alumini	USY7140WAE85	05/15/2020	05/15/2025	4.750	1,000,000,000	60,601,528
	USY7140WAA63	11/15/2018	11/15/2021	5.230	498,671,000	16,447,360
	USY7140WAF50	05/15/2020	05/15/2030	5.450	1,000,000,000	61,079,590.44
	USY7140WAB47	11/15/2019	11/15/2023	5.710	310,939,000	13,176,920
	USY7140WAC20	11/15/2018	11/15/2028	6.530	598,460,000	36,264,242
Longfor Properties	XS1743535491	01/16/2018	01/16/2028	4.500	500,000,000	34,963,499
	XS1633950453	07/13/2017	07/13/2022	3.875	450,000,000	13,282,281
	XS2033262895	9/16/2019	9/16/2029	3.950	850,000,000	44,377,944
	XS2098650414	1/13/2020	1/13/2032	3.850	400,000,000	31,701,054
	XS2098539815	1/13/2020	4/13/2027	3.375	250,000,000	16,287,838
	XS0877742105	1/29/2013	1/29/2023	6.750	500,000,000	15,988,419
	XS1743535228	1/16/2018	4/16/2023	3.900	300,000,000	9,683,816
	XS0844323930	10/18/2012	10/18/2019	6.875	400,000,000	12,740,378

The column "Volume Issued (in USD)" is referred to the volume issued by the bond and the column "Volume Executed (in USD" is referred to the volume we have actually used through the observations extracted from the database in the sample.

Appendix 4: Cumulative Net Profits for Each Bond Market (Sovereign, Corporate and High-Yield) and According to Each Price Window (10-min, 30-min, 60-min and 1-min, 5-min)

See Figs. 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15.

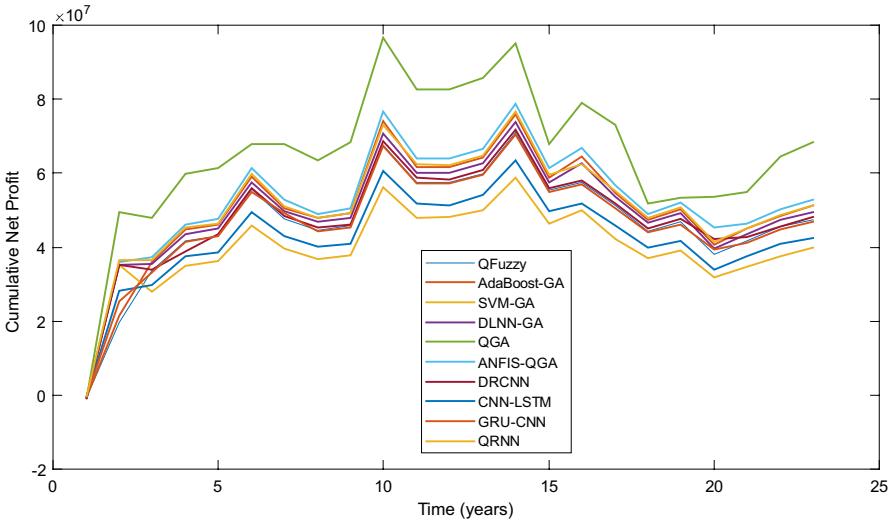

Fig. 5 Cumulative net profits for sovereign bonds (10-min)

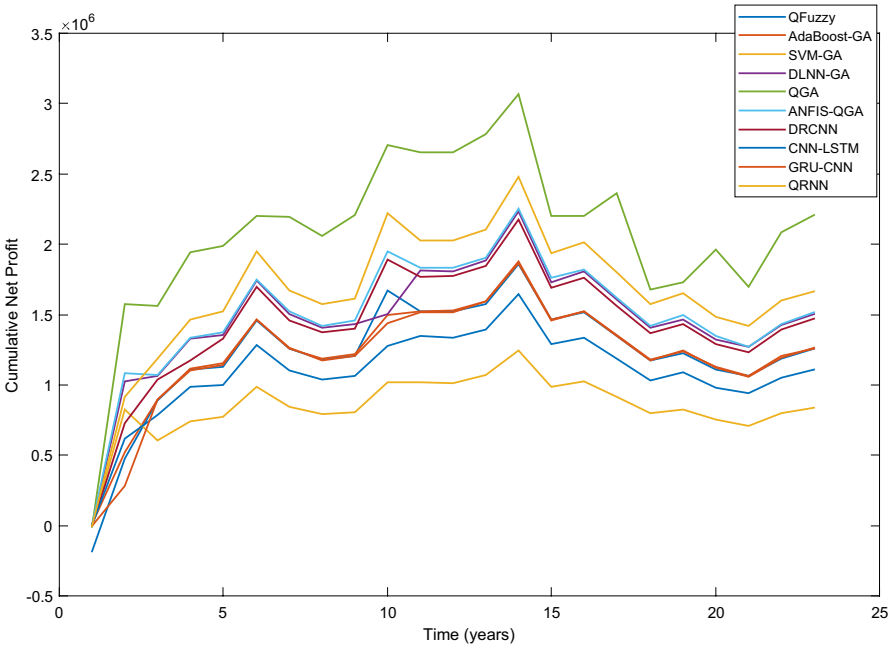

Fig. 6 Cumulative net profits for corporate bonds (10-min)

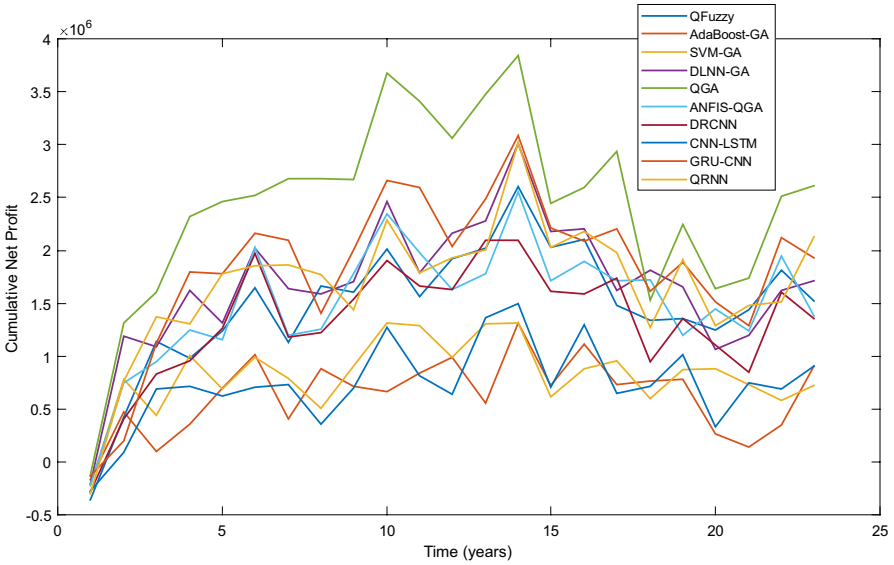

Fig. 7 Cumulative net profits for high-yield bonds (10-min)

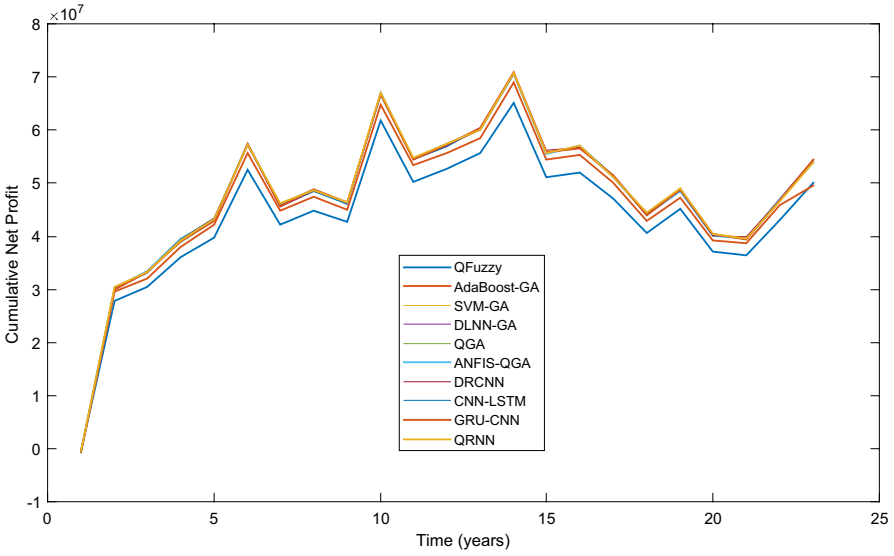

Fig. 8 Cumulative net profits for sovereign bonds (30-min)

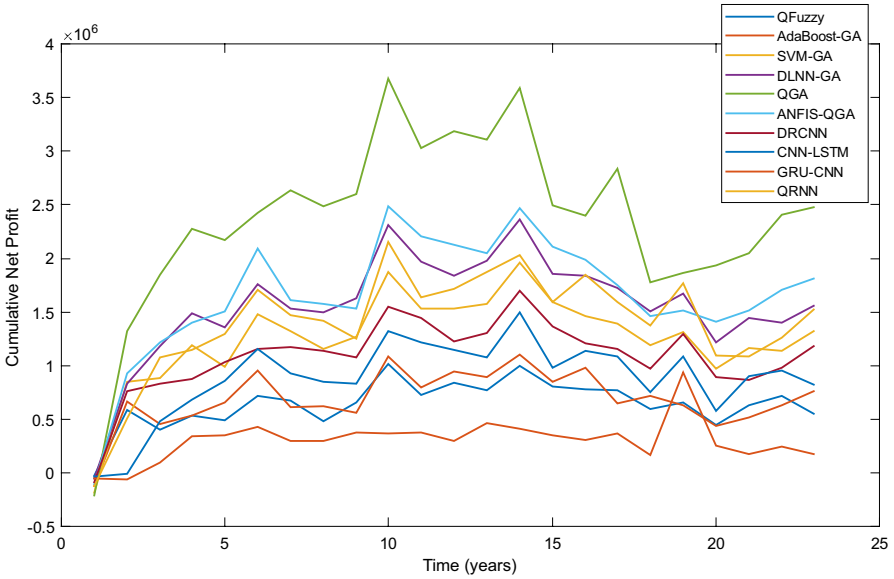

Fig. 9 Cumulative net profits for corporate bonds (30-min)

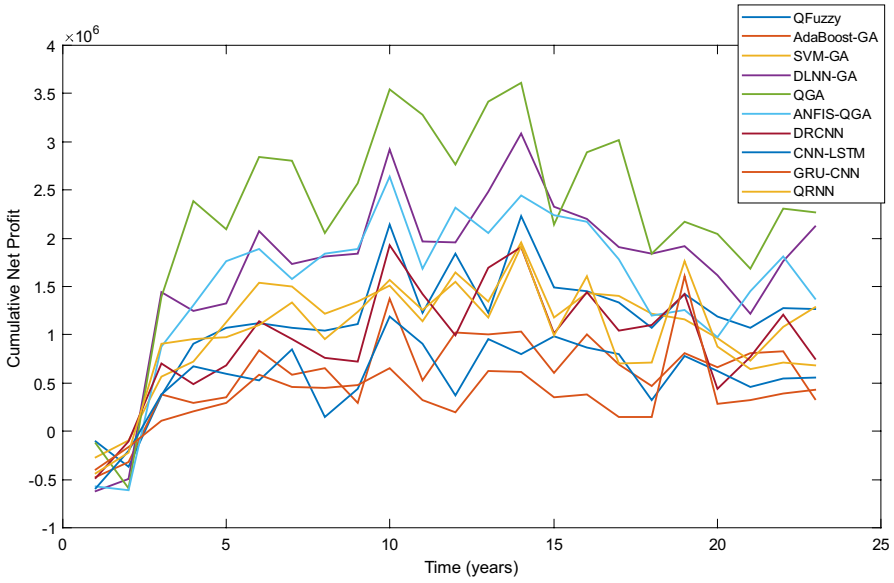

Fig. 10 Cumulative net profits for high-yield bonds (30-min)

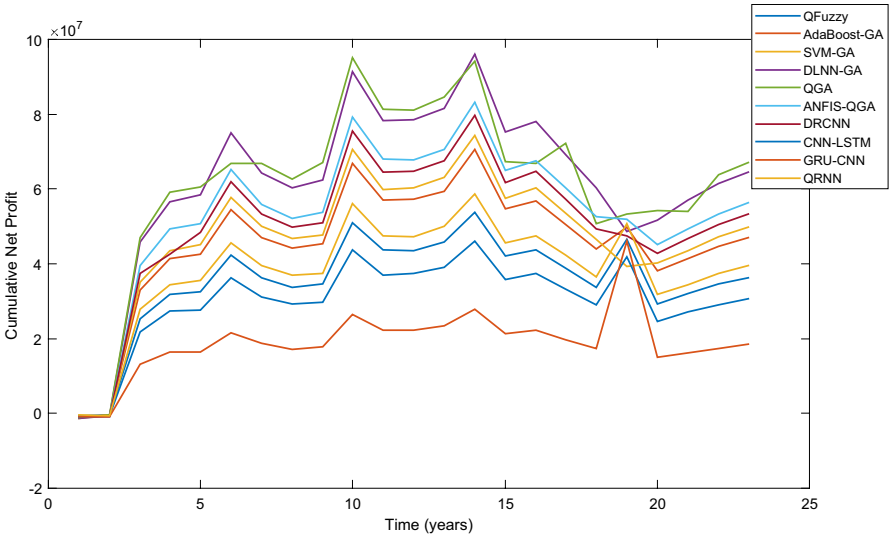

Fig. 11 Cumulative net profits for sovereign bonds (60-min)

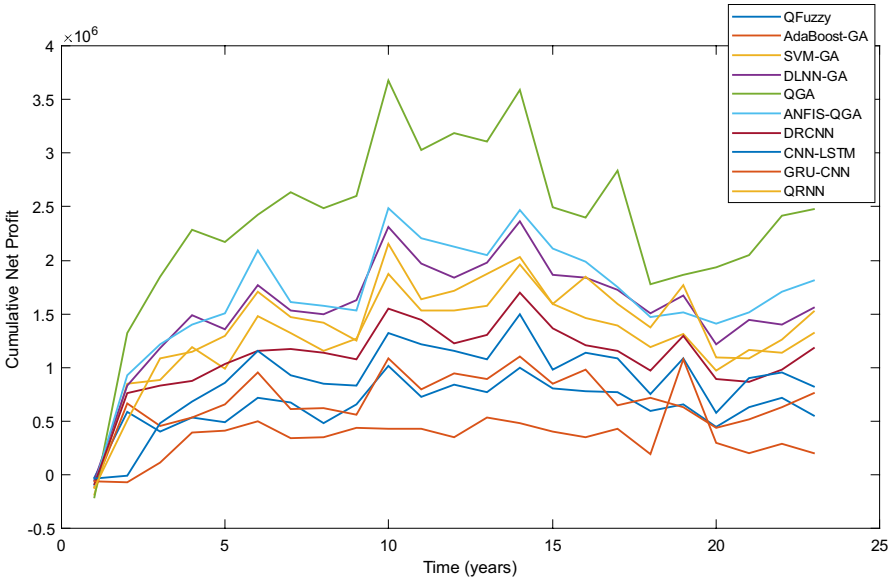

Fig. 12 Cumulative net profits for corporate bonds (60-min)

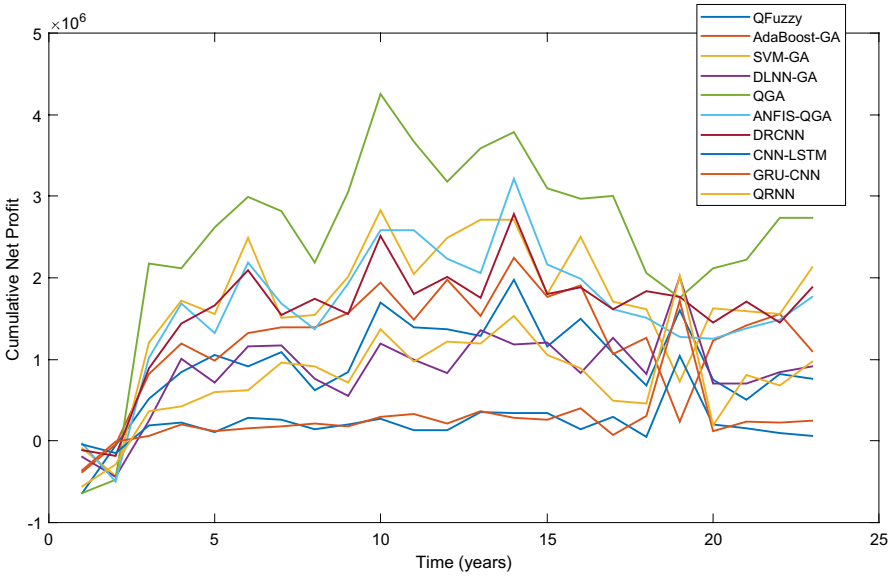

Fig. 13 Cumulative net profits for high-yield bonds (60-min)

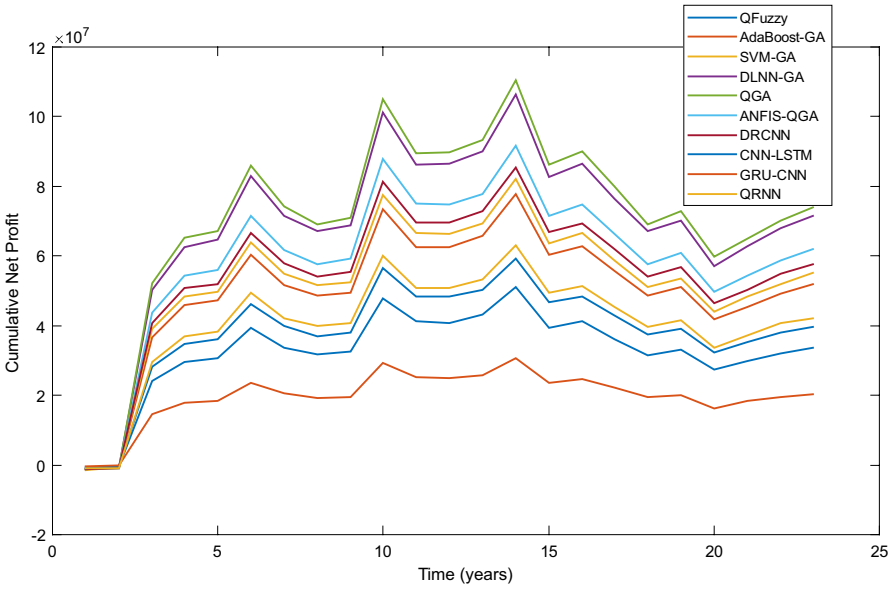

Fig. 14 Cumulative net profits for sovereign bonds (1-min)

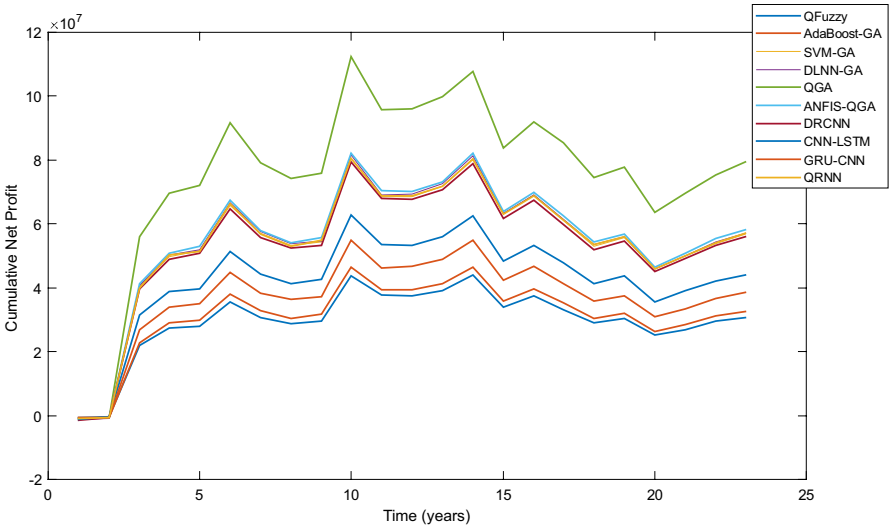

Fig. 15 Cumulative net profits for sovereign bonds (5-min)

Acknowledgements Not applicable.

Author Contributions All the authors contributed an equal effort to complete the manuscript.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This research was funded by the Universitat de Barcelona, under the grant UB-AE-AS017634.

Availability of Data and Material The datasets used and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no competing interests.

References

Albagli, E., Ceballos, L., Claro, S., & Romero, D. (2019). Channels of US monetary policy spillovers to international bond markets. *Journal of Financial Economics, 134*(2), 447–473.

Akyildirim, E., Bariviera, A. F., Nguyen, D. K., & Sensoy, A. (2022). Forecasting high-frequency stock returns: A comparison of alternative methods. *Annals of Operations Research, 313*(2), 639–690.

Alaminos, D., Salas, M.B., & Fernández-Gámez, M.A. (2022a). Deep Neural Networks Methods for Estimating Market Microstructure and Speculative Attacks Models: The case of Government Bond Market. *The Singapore Economic Review*, Online Ready.

Alaminos, D., Salas, M. B., & Fernández-Gámez, M. A. (2022b). Forecasting stock market crashes via real-time recession probabilities: a quantum computing approach. *Fractals-Complex Geometry, Patterns, and Scaling in Nature and Society, 30*(5), 1–16.

Aldana, I.B. (2017). Investing in Distressed Debt in Europe. Globe Law and Business Limited. London, United Kingdom.

Aldrich, E. (2013). *High-frequency trading: A practical guide to algorithmic strategies and trading systems*. John Wiley & Sons.

Aloud, M. E., & Alkhamees, N. (2021). Intelligent algorithmic trading strategy using reinforcement learning and directional change. *IEEE Access, 9*, 114659–114671.

Bai, J., Fleming, M.J., & Horan, C. (2013). The microstructure of China's government bond market. *FRB of New York Staff Report*, (622).

Beirne, J., & Sugandi, E. (2023). Central bank asset purchase programs in emerging market economies. *Finance Research Letters, 54*(C), 103769.

Benedetti, M., Lloyd, E., Sack, S., & Fiorentini, M. (2019). Parameterized quantum circuits as machine learning models. *Quantum Science and Technology, 4*(4), 043001.

Berezin, F. A., & Shubin, M. (2012). *The Schrödinger Equation* (Vol. 66). Springer Science & Business Media.

Bessembinder, H., Spatt, C., & Venkataraman, K. (2020). A survey of the microstructure of fixed-income markets. *Journal of Financial and Quantitative Analysis, 55*(1), 1–45.

Biais, B., & Green, R. (2019). The microestructure of the bond market in the 20th century. *Review of Economic Dynamics, 33*, 250–271. https://doi.org/10.1016/j.red.2019.01.003

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford University Press.

Blanchard, O. (2023). *Fiscal policy under low interest rates*. The MIT Press.

Bodenhofer, U. (2003). Genetic algorithms: theory and applications. *Lecture notes, Fuzzy Logic Laboratorium Linz-Hagenberg, Winter*, *2004*.

Bossy, E., & Gigan, S. (2016). Photoacoustics with coherent light. *Photoacoustics, 4*(1), 22–35.

Cao, M., & Shang, F. (2010, March). Double chains quantum genetic algorithm with application in training of process neural networks. In *2010 Second International Workshop on Education Technology and Computer Science* (Vol. 1, pp. 19–22). IEEE.

Cao, Y., Guerreschi, G. G., & Aspuru-Guzik, A. (2017). Quantum neuron: an elementary building block for machine learning on quantum computers. arXiv preprint arXiv:1711.11240.

Campbell, R., Huisman, R., & Koedijk, K. (2001). Optimal portfolio selection in a value-at-risk framework. *Journal of Banking & Finance, 25*(9), 1789–1804.

Caponetto, R., Fortuna, L., Fazzino, S., & Xibilia, M. (2003). Chaotic sequences to improve the performance of evolutionary algorithms. *IEEE Transactions on Evolutionary Computation, 7*, 289–304.

Chakravarty, S., & Sarkar, A. (2003). Trading costs in three U.S. bond markets. *Journal of Fixed Income, 13*, 39–48.

Cheng, Y., Zheng, Z., Wang, J., Yang, L., & Wan, S. (2019). Attribute reduction based on genetic algorithm for the coevolution of meteorological data in the industrial internet of things. *Wireless communications and mobile computing*, *2019*.

Chih-Hung, W., Gwo-Hshiung, T., & Rong-Ho, L. (2009). A Novel hybrid genetic algorithm for kernel function and parameter optimization in support vector regression. *Expert Systems with Applications, 36*, 4725–4735.

Chong, T., & Ng, W.-K. (2008). Technical analysis and the London stock exchange: testing the MACD and RSI rules using the FT30. *Applied Economics Letters, 15*(14), 1111–1114.

Christiansen, C., & Ranaldo, A. (2007). Realized bond—stock correlation: Macroeconomic announcement effects. *Journal of Futures Markets: Futures, Options, and Other Derivative Products, 27*(5), 439–469.

Clapham, B., Haferkorn, M., & Zimmermann, K. (2023). The impact of high-frequency trading on modern securities markets. *Business & Information Systems Engineering, 65*, 7–24.

Conkey, D. B., Brown, A. N., Caravaca-Aguirre, A. M., & Piestun, R. (2012). Genetic algorithm optimization for focusing through turbid media in noisy environments. *Optics Express, 20*(5), 4840–4849.

Darbha, M., & Dufour, A. (2013). Microstructure of the Euro-area government bond market. In H. K. Baker & H. Kiymaz (Eds.), *Market microstructure in emerging and developed markets. Robert W. Kolb series in finance.* John Wiley.

De Almeida, R. L., & Neves, R. F. (2022). Stock market prediction and portfolio composition using a hybrid approach combined with self-adaptive evolutionary algorithm. *Expert Systems with Applications, 204*, 117478.

Deng, S., Huang, X., Qin, Z., Fu, Z., & Yang, T. (2021). A novel hybrid method for direction forecasting and trading of Apple Futures. *Applied Soft Computing, 110*, 107734.

Deng, S., Zhu, Y., Huang, X., Duan, S., & Fu, Z. (2022). High-Frequency Direction Forecasting of the Futures Market Using a Machine-Learning-Based Method. *Future Internet, 14*(6), 180.

Dixon, M. F., Polson, N. G., & Sokolov, V. O. (2018). Deep learning for spatio-temporal modeling: Dynamic traffic flows and high frequency trading. *Applied Stochastic Models in Business and Industry, 35*(3), 788–807.

Dodd, O., Frijns, B., Idriawan, I., & Pascual, R. (2023). US cross-listing and domestic high-frequency trading: Evidence from Canadian stocks. *Journal of Empirical Finance, 72*, 301–320.

Dougherty, J., Kohavi, R., & Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In *Machine learning proceedings 1995* (pp. 194–202). Morgan Kaufmann.

Drezner, Z., & Misevičius, A. (2013). Enhancing the performance of hybrid genetic algorithms by differential improvement. *Computers & Operations Research, 40*(4), 1038–1046.

Elton, E. J., Martin, J. G., & Christopher, R. B. (1995). Fundamental economic variables, expected returns and bond fund performance. *Journal of Finance, 40*, 1229–1256.

Fabozzi, F. J., & Fabozzi, F. A. (2021). *Bond markets, analysis, and strategies* (10th ed.). The MIT Press.

Friewald, N., & Nagler, F. (2019). Over-the-counter market frictions and yield spread changes. *Journal of Finance, 74*(6), 3217–3257. https://doi.org/10.1111/jofi.12827

Frino, A., Garcia, M., & Zhou, Z. (2020). Impact of algorithmic trading on speed of adjustment to new information: Evidence from interest rate derivatives. *Journal of Futures Markets, 40*(5), 749–760.

Frino, A., Mollica, V., & Webb, R. I. (2014). The impact of co-location of securities exchanges'and traders' computer servers on market liquidity. *Journal of Futures Markets, 34*, 20–33.

Gao, X., Li, X., Zhao, B., Ji, W., Jing, X., & He, Y. (2019). Short-term electricity load forecasting model based on EMD-GRU with feature selection. *Energies, 12*(6), 1140.

Ghoddusi, H., Creamer, G. G., & Rafizadeh, N. (2019). Machine learning in energy economics and finance: A review. *Energy Economics, 81*, 709–727. https://doi.org/10.1016/j.eneco.2019.05.006

Gilchrist, S., Yue, V., & Zakrajšek, E. (2019). US monetary policy and international bond markets. *Journal of Money, Credit and Banking, 51*(51), 127–161.

Glode, V., & Opp, C. (2020). Over-the-counter vs. limit-order markets: The role of traders' expertise. *Review of Financial Studies, 33*(2), 866–915. https://doi.org/10.1093/rfs/hhz061

Goldberg, D. E. (1990). A note on Boltzmann tournament selection for genetic algorithms and populationoriented simulated annealing. *Complex Systems, 4*, 44.

Goldblum, M., Schwarzschild, A., Patel, A., & Goldstein, T. (2021). Adversarial attacks on machine learning systems for high-frequency trading. In *Proceedings of the Second ACM International Conference on AI in Finance* (pp. 1–9).

Gomber, P., & Haferkorn. (2015). *High frequency trading*. *Encyclopedia of information science and technology* (3rd ed.). IGI Global.

Goodell, J. W., Kumar, S., Lim, W. M., & Pattnaik, D. (2021). Artificial intelligence and machine learning in finance: Identifying foundations, themes, and research clusters from bibliometric analysis. *Journal of Behavioral and Experimental Finance, 32*, 100577.

Goudarzi, M., & Bazzana, F. (2023). Identification of high-frequency trading: A machine learning approach. *Research in International Business and Finance*, *66*, 102078.

Grinold, R. C., & Ronald, N. K. (1999). *Active portfolio management: A quantitative approach for producing superior returns and controlling risk*. McGraw-Hill.

Grover, L. K. (2005). Fixed-point quantum search. *Physical Review Letters, 95*(15), 150501.

Guerreschi, G. G. (2019). Repeat-until-success circuits with fixed-point oblivious amplitude amplification. *Physical Review A, 99*(2), 022306.

Gündüz, Y., Pelizzon, L., Schneider, M., and Subrahmanyam, M.G. (2023). Lighting Up the Dark: Liquidity in the German Corporate Bond Market. *The Journal of Fixed Income* 32(4).

Gupta, N., & Jalal, A. S. (2020). Integration of textual cues for fine-grained image captioning using deep CNN and LSTM. *Neural Computing and Applications, 32*(24), 17899–17908.

Hansen, K. B., & Borch, C. (2022). Alternative data and sentiment analysis: Prospecting non-standard data in machine learning-driven finance. *Big Data & Society, 9*(1), 20539517211070700.

He, K., Chen, Y., & Tso, G. K. (2017). Price forecasting in the precious metal market: A multivariate EMD denoising approach. *Resources Policy, 54*, 9–24.

Hendershott, T., Jones, C. M., & Menkveld, A. J. (2011). Does algorithmic trading improve liquidity? *The Journal of Finance, 66*(1), 1–33.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*(8), 832–844.

Holden, C. W., Liu, X., Mao, Y, & Nam, J. (2018) Price Discovery in the Stock, OTC Corporate Bond, and NYSE Corporate Bond Markets (June 9, 2018). Kelley School of Business Research Paper No. 18–53.

Houg, G., & Warga, A. (2000). An empirical study of bond market transactions. *Financial Analyst Journal, 56*(2), 32–46.

Huang, C. W., & Narayanan, S. S. (2017, July). Deep convolutional recurrent neural network with attention mechanism for robust speech emotion recognition. In *2017 IEEE international conference on multimedia and expo (ICME)* (pp. 583–588). IEEE.

Ibbotson, R. G., & Kaplan, P. D. (2000). Does asset allocation policy explain 40%, 90% or 100% of performance? *Financial Analysts Journal, 56*, 26–33.

International Monetary Fund. (2021). Developing Government Bond Markets. The World Bank, ISBN: 9780821349557. ISSN: 2663-3744, 21 Sep 2001. DOI: Doi: https://doi.org/10.5089/9780821349557.069

Jarrow, R. (2019). *Modeling fixed income securities and interest rate options* (3rd ed.). Stanford University Press.

Jeddi, S., & Sharifian, S. (2020). A hybrid wavelet decomposer and GMDH-ELM ensemble model for Network function virtualization workload forecasting in cloud computing. *Applied Soft Computing, 88*, 105940.

Kara, Y., Boyacioglu, M. A., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications, 38*(5), 5311–5319.

Keller, J. M., Gray, M. R., & Givens, J. A. (1985). A fuzzy K-nearest neighbor algorithm. *IEEE Trans. Systems Man Cybernet., 15*(4), 580–585.

Kim, B. S., & Kim, T. G. (2019). Cooperation of simulation and data model for performance analysis of complex systems. *International Journal of Simulation Modelling, 18*(4), 608–619.

Kirkpatrick, S., Gelatt, C. D., Jr., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*, 671–680.

Knowles, J. D., & Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation, 8*(2), 149–172.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95)*, vol. 2, pp. 1137–1143.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Pdf ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*, 84–90.

Lahmiri, S., & Bekiros, S. (2021). Deep Learning Forecasting in Cryptocurrency High-Frequency Trading. *Cognitive Computation, 13*, 485–487.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278–2324.

Lewis, J., Hart, E., & Ritchie, G. (1998). A comparison of dominance mechanisms and simple mutation on non-stationary problems. *Parallel Problem Solving from Nature (PPSN v), 1498*, 139–148.

Li, S., Deng, M., Lee, J., Sinha, A., & Barbastathis, G. (2018). Imaging through glass diffusers using densely connected convolutional networks. *Optica, 5*(7), 803–813.

Li, J., Hao, J., Sun, X., & Feng, Q. (2021). Forecasting China's sovereign CDS with a decomposition reconstruction strategy. *Applied Soft Computing, 105*, 107291.

Lin, Z., Chen, Y., Zhao, X., & Wang, G. (2013). Spectral-spatial classification of hyperspectral image using autoencoders. In *2013 9th International Conference on Information, Communications & Signal Processing* (pp. 1–5).

Lucas, A., Iliadis, M., Molina, R., & Katsaggelos, A. K. (2018). Using deep neural networks for inverse problems in imaging: Beyond analytical methods. *IEEE Signal Processing Magazine, 35*(1), 20–36.

Ma, M., & Mao, Z. (2019, June). Deep recurrent convolutional neural network for remaining useful life prediction. In *2019 IEEE international conference on prognostics and health management (ICPHM)* (pp. 1–4). IEEE.

Mahanti, S., Nashikkar, A., Subramanyam, M. G., Chacko, G., & Mallik, G. (2008). Latent liquidity: A new measure of liquidity with an application to corporate bonds. *Journal of Financial Economics, 88*, 272–298.

McCann, M. T., Jin, K. H., & Unser, M. (2017). Convolutional neural networks for inverse problems in imaging: A review. *IEEE Signal Processing Magazine, 34*(6), 85–95.

Milana, C., & Ashta, A. (2021). Artificial intelligence techniques in finance and financial markets: A survey of the literature. *Strategic Change, 30*(3), 189–209.

Nauck, D., Klawonn, F., & Kruse, R. (1997). *Foundations of neuro-fuzzy systems*. Wiley.

Nauck, D., Kruse, R. (1997). New learning strategies for NEFCLASS. In Proc.Seventh International Fuzzy Systems Association World Congress IFSA´97, volume IV, 50–55, Prague.

Neklyudov, A. (2019). Bid–Ask Spreads and the Over-the-Counter Interdealer Markets: Core and Peripheral Dealers. *Review of Economic Dynamics, 33*, 57–84. https://doi.org/10.1016/j.red.2019.04.005

Nelson, C., & Siegel, A. F. (1987). Parsimonious modeling of yield curves. *Journal of Business, 60*, 473–489.

Nielsen, M. A., & Chuang, I. L. (2001). Quantum computation and quantum information. *Physics Today, 54*(2), 60.

Norouzi, M., Collins, M., Johnson, M. A., Fleet, D. J., & Kohli, P. (2015). Efficient non-greedy optimization of decision trees. *Advances in Neural Information Processing Systems*, *28*.

Nowak, S., Andritzky, J. R., Jobst, A., & N. T. Tamirisa. (2009). Macroeconomic fundamentals, price discovery and volatility dynamics in emerging markets. International Monetary Fund Working Paper No.09/147.

Nunes, M., Gerding, E., McGroarty, F., & Niranjan, M. (2018). Artificial neural networks in fixed income markets for yield curve forecasting. Available at SSRN 3144622.

Nunes, M., Gerding, E., McGroarty, F., & Niranjan, M. (2019). A comparison of multitask and single task learning with artificial neural networks for yield curve forecasting. *Expert Systems with Applications, 119*, 362–375.

Papadamou, S., Fassas, A. P., Kenourgios, D., & Dimitriou, D. (2021). Flight-to-quality between global stock and bond markets in the COVID era. *Finance Research Letters, 38*(C), 101852.

Ping-Feng, P., Chih-Shen, L., Wei-Chiang, H., & Chen-Tung, C. (2006). A hybrid support vector machine regression for exchange rate prediction. *International Journal of Information and Management Sciences, 17*, 19–32.

Qin, L., Yu, N., & Zhao, D. (2018). Applying the convolutional neural network deep learning technology to behavioural recognition in intelligent video. *Tehnički Vjesnik, 25*(2), 528–535.

Quinlan, J. R. (1993). *C4,5: programs for machine learning* (p. 1993). Morgan Kaufmann Publishers Inc.

Ramlall, I. (2016). *Applied technical analysis for advanced learners and practitioners*. Emerald Publishing.

Ranaldo, A., Schaffner, P., & Vasios, M. (2019). Regulatory effects on short-term interest rates. *Journal of Financial Economics*, *141*(2), 750–770.

Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-validation. *Encyclopedia of Database Systems*. https://doi.org/10.1007/978-0-387-39940-9_565

Rundo, F. (2019). Deep LSTM with reinforcement learning layer for financial trend prediction in FX high frequency trading systems. *Applied Sciences, 9*(20), 4460.

Schestag, R., Schuster, P., & Uhrig-Homburg, M. (2016). Measuring liquidity in bond markets. *Review of Financial Studies, 29*, 1170–1219.

Schilling, F. (2016). The effect of batch normalization on deep convolutional neural networks.

Schneider, M. (2018). Market Microstructure, Price Impact and Liquidity in Fixed Income Markets. PhD Thesis, Scuola Normale Superiore Pisa, Italy.

Schrödinger, E. (1935). The present status of quantum mechanics. *Die Naturwissenschaften, 23*(48), 1–26.

Securities and Exchange Commission. (2014). Equity market structure literature review—Part II: High frequency trading. Retrieved from https://www.sec.gov/marketstructure/research/hft_lit_review_march_2014.pdf.

Sezer, O. B., & Ozbayoglu, A. M. (2018). Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Applied Soft Computing, 70*, 525–538.

Singh, P., Dhiman, G., & Kaur, A. (2018). A quantum approach for time series data based on graph and Schrödinger equations methods. *Modern Physics Letters A, 33*(35), 1850208.

Singh, P., & Huang, Y. P. (2019). A new hybrid time series forecasting model based on the neutrosophic set and quantum optimization algorithm. *Computers in Industry, 111*, 121–139.

Sivaraj, R., & Ravichandran, T. (2011). A review of selection methods in genetic algorithm. *International Journal of Engineering Science and Technology, 3*, 3792–3797.

Slade, S. (2017). *Artificial intelligence applications on wall street*. Taylor & Francis.

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 31, No. 1).

Ta, V. D., Liu, C. M., & Tadesse, D. A. (2020). Portfolio optimization-based stock prediction using long-short term memory network in quantitative trading. *Applied Sciences, 10*(2), 437.

Tacchino, F., Macchiavello, C., Gerace, D., & Bajoni, D. (2019). An artificial neuron implemented on an actual quantum processor. *Npj Quantum Information, 5*(1), 1–8.

Tang, Y. (2013). Deep learning using linear support vector machines. arXiv preprint arXiv:1306.0239.

Vellekoop, I. M., & Mosk, A. P. (2008). Phase control algorithms for focusing light through turbid media. *Optics Communications, 281*(11), 3071–3080.

Vukovic, D., Vyklyuk, Y., Matsiuk, N., & Maiti, M. (2020). Neural network forecasting in prediction Sharpe ratio: Evidence from EU debt market. *Physica a: Statistical Mechanics and Its Applications, 542*, 123331.

Wang, J., Gao, L., Zhang, H., & Xu, J. (2011, July). Adaboost with SVM-based classifier for the classification of brain motor imagery tasks. In *International Conference on Universal Access in Human-Computer Interaction* (pp. 629–634). Springer, Berlin, Heidelberg.

Wang, Y., Ma, F., Wei, Y., & Wu, C. (2016). Forecasting realized volatility in a changing world: A dynamic model averaging approach. *Journal of Banking & Finance, 64*, 136–149.

Wan, K. H., Dahlsten, O., Kristjánsson, H., Gardner, R., & Kim, M. S. (2017). Quantum generalisation of feedforward neural networks. *Npj Quantum Information, 3*(1), 1–8.

Wei, Z., & Chen, X. (2018). Deep-learning schemes for full-wave nonlinear inverse scattering problems. *IEEE Transactions on Geoscience and Remote Sensing, 57*(4), 1849–1860.

Zhao, Y., Li, J., & Yu, L. (2017). A deep learning ensemble approach for crude oil price forecasting. *Energy Economics, 66*, 9–16.