Predicting extreme events in the stock market using generative adversarial networks

Badre Labiad [a,1,*], Abdelaziz Berrado [a,2], Loubna Benabbou [b,3]

[a] Equipe AMIPS, Ecole Mohammadia d'Ingénieurs, Mohammed V University in Rabat, Morocco
[b] Department of Management Science, Université du Québec à Rimouski (UQAR), Campus de Lévis Québec Canada
[1] labiad.badre@gmail.com ; [2] berrado@emi.ac.ma; [3] loubna_benabbou@uqar.ca
* corresponding author

ARTICLE INFO

Article history
Received August 26, 2022
Revised April 10, 2023
Accepted April 24, 2023
Available online May 30, 2023

Keywords
Extreme events prediction
Time series generation
Stock markets simulation
Generative adversarial networks
Long short-term memory

ABSTRACT

Accurately predicting extreme stock market fluctuations at the right time will allow traders and investors to make better-informed investment decisions and practice more efficient financial risk management. However, extreme stock market events are particularly hard to model because of their scarce and erratic nature. Moreover, strong trading strategies, market stress tests, and portfolio optimization largely rely on sound data. While the application of generative adversarial networks (GANs) for stock forecasting has been an active area of research, there is still a gap in the literature on using GANs for extreme market movement prediction and simulation. In this study, we proposed a framework based on GANs to efficiently model stock prices' extreme movements. By creating synthetic real-looking data, the framework simulated multiple possible market-evolution scenarios, which can be used to improve the forecasting quality of future market variations. The fidelity and predictive power of the generated data were tested by quantitative and qualitative metrics. Our experimental results on S&P 500 and five emerging market stock data show that the proposed framework is capable of producing a realistic time series by recovering important properties from real data. The results presented in this work suggest that the underlying dynamics of extreme stock market variations can be captured efficiently by some state-of-the-art GAN architectures. This conclusion has great practical implications for investors, traders, and corporations willing to anticipate the future trends of their financial assets. The proposed framework can be used as a simulation tool to mimic stock market behaviors.

1. Introduction

Stock markets are dynamic and complex environments whose behavior is largely influenced by exogenous factors (such as political and geopolitical, economic, and natural). The influence of these factors can be so extreme that it can trigger certain crises such as bubbles, panics, booms, meltdowns, or crashes [1], [2]. For example, stock markets across the world reacted with extreme variations to the spread of the COVID-19 pandemic [3]–[6]. In such contexts, the challenge traders and investors face is proactively anticipating and integrating the impacts of such events into their trading strategies Fig. 1 shows the drop in the S&P 500 and five major emerging stock market indices during the early days of the COVID-19 pandemic (1st semester 2020).

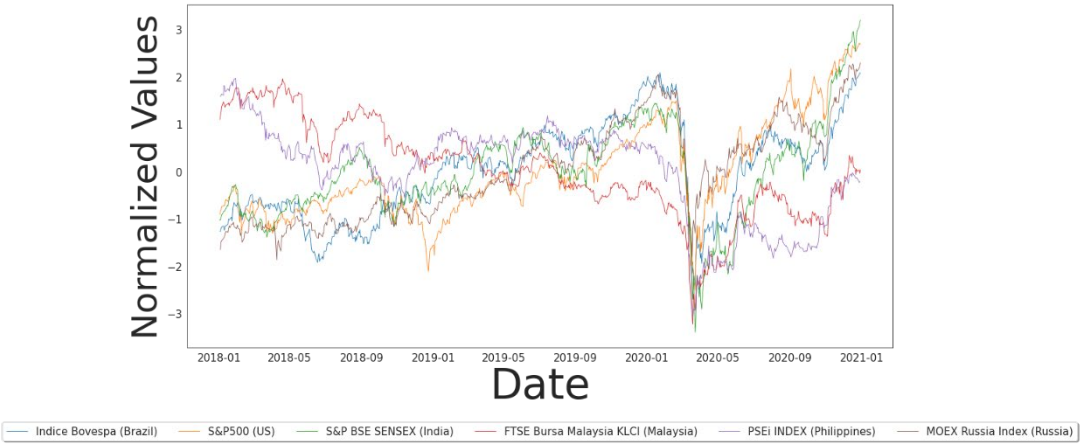


Fig. 1. Crash of S&P 500 and five major emerging stock market indices during the first days of the COVID-19 pandemic (1st semester 2020)

Efficiently simulating stock prices' extreme fluctuations would allow decision-makers and investors to make better-informed investment decisions and practice more efficient financial risk management. The classical procedure of evaluating trading strategies involves testing them on real, observed market data through a process called backtesting [7]. However, a major shortcoming of this procedure is its reliance on historical data, which offers only one view of market behavior. Efficient trading strategy calibration would require multiple market scenarios and a wide range of realistic price variations.

In this study, we designed a procedure to simulate stock-market conditions, even the extreme ones, by generating realistic synthetic market data and reproducing a wide range of price variations. This framework can be used as a practical tool to test and improve forecasting models, trading strategy calibration, portfolio management, market manipulation detection, and risk management.

The difficulties encountered while dealing with stock market data are multiple and varied (such as the nature of the factors influencing the variations; the non-linear, non-parametric, nonstationary, and arbitrary nature of the data; and extreme movements) [8]. Moreover, access to granular microstructure market data is limited and expensive. Access to a richer data set is of great importance for the research community. Therefore, the procedure proposed in this study also helps overcome data-access limitations for researchers, traders, and investors by creating synthetic datasets with the same statistical properties as the original data.

Understanding and modeling the temporal dynamics of stock markets to simulate their behaviors and predict their future movements, especially during periods of crisis and high volatility, is an objective that has caught the interest of researchers, traders, and investors for several years [9]–[11]. Deep learning [12] is one of the most promising techniques tested for stock-market prediction. This technique is inspired by the brain structure consisting of multi-layer neural networks [13] and overcomes the limitations of traditional machine-learning methods that cannot efficiently capture the complexity of financial data [14]. Motivated by the successes of deep recurrent learning models [12], especially in their ability to learn long-term dependency information, several studies have attempted to apply them to financial data, particularly to stock markets [15]–[18]. Deploying effective deep-learning models requires large and balanced datasets to get convincing results because small and unbalanced training datasets lead to overfitting and poor generalization [14]. Extreme variations in the stock market are rare events [15]. This is an additional difficulty from the modeling perspective. Furthermore, stock price data with extreme variations are unbalanced time-series datasets. To mitigate this challenge, algorithm solutions and data-oriented methods are used. The former consists of adapting algorithms to better handle minority class errors (extreme instances). The second, a more flexible approach, consists of data augmentation and oversampling (undersampling) of the minority (majority) class [19]. Generally, this technique is used to fix imbalanced data sets and missing data or create new synthetic data points when access to real data is limited [20], [21]. The synthetic data must be as realistic and flexible as the real one. Generative adversarial networks (GANs) are among the data augmentation techniques tested for stock markets [22]–[25]. However, the majority of the above-mentioned studies assessed the quality of their results under normal market conditions. Extreme variations in the stock markets are extraordinary and rare circumstances. Generating data similar to extreme market situations requires additional conditioning and tuning of the generating process to produce realistic synthetic data.

Although GAN application for stock-market prediction is an active area of research, to the best of our knowledge, there is still a gap in the literature on the use of GANs for market simulation and extreme-event prediction. Considering the above observations, we designed a framework based on the GAN procedure using three state-of-the-art algorithms, which offer interesting features regarding the possibility of conditioning the extremeness of generated data, to simulate market behavior and produce new synthetic examples that are used to efficiently predict financial stock prices during extreme variations. We tested the framework on stock data from the S&P 500 and five emerging markets. In this study, GANs were employed as a data augmentation technique to create new realistic synthetic examples to tackle the scarcity of extreme observations in the training dataset and to simulate multiple scenarios of market evolution. After enriching the training dataset, a long short-term memory model (LSTM) was used for prediction tasks.

Our study makes the following key contributions:

- We propose a framework capable of producing synthetic data that simulates market behaviors: this work contributes to the new area of research in GAN usage for stock market simulation. This application of GANs yields the potential of overcoming issues related to limited data access.
- We illustrate that synthetic stock market data can be used to enhance training datasets and improve the forecasting of extreme events: by proposing a clear methodology to assess the quality of the generated data, we show that the synthetic real-looking examples can efficiently be used to perform some practical tasks such as the forecasting of stock prices. We trained an LSTM model on both real and synthetic data to forecast stock price variations and evaluated the improvement in the forecasting error metrics after using the new synthetic examples.
- We present a comparison between three different GAN architectures to perform extreme events prediction—which, according to our knowledge, is still an understudied application of GANs: the usage of GANs for extreme event modeling is a promising area of research, and this work illustrates the practical steps to undertake to get robust results.

The rest of the paper is organized as follows: The proposed framework is detailed in section 2. The experiment results are discussed in section 3. Finally, conclusions and possible extensions are discussed in the last section.

2. Method

2.1. Problem formulation

The underlying process of stock-price evolution is hard to model. Trying to capture the stock-market dynamics by explicitly developing handcrafted assumptions and rules is intractable and inefficient. However, the GAN framework offers the possibility of reproducing features of the underlying process of the stock market movements with high fidelity. We assumed that the generated data will enrich the training examples, particularly more examples from the tail of the underlying distribution, as our focus was on the prediction of extreme events.

$Let\ X_{1:T} = (X_1, X_2, .., X_T)$ be a sample of our dataset, where X_i represents the features of the i^{th} time-step of the sample. T represents the time window size. The objective was to allow the generator framework to reproduce high-quality data by learning a density $\hat{p}\ (X1{:}\,T)$ that approximated the best original density $p\ (X1{:}\,T)$, where $\hat{p}\ (X1{:}\,T)$ is the learned density that minimizes the following distance:

$$Min\ \hat{p}\ Dist(\hat{p}\ (X1{:}\,T)||\ p\ (X1{:}\,T)) \tag{1}$$

The metric (Dist) is the Jensen–Shannon Divergence [26].

In addition, the generative process described above was needed to efficiently capture the temporal transitions. More formally, we needed to accurately capture the conditional distribution $p(XT|X1{:}T-1)$ that characterized temporal dependencies. The goal was to accurately predict $\hat{X}T+1, \hat{X}T+2, \ldots, \hat{X}T+k$ by training a model using real and synthetic data. The synthetic data itself was generated using the sample $X1{:}T = (X1, X2, \ldots, XT)$.

2.2. The proposed framework

In this section, we present the proposed framework for extreme events forecasting. As mentioned before, synthetic data were used in our study to overcome the rarity of extreme events in the stock market. Hence, by generating new synthetic data, we improved the richness of the initial dataset and gave the learning model new data to train with. To assess the fidelity and predictive power of the generated data, we used it to train a recurrent algorithm—namely, LSTM—and compared the quality of the predictions on the synthetic examples against the predictions on the real ones. Furthermore, quantitative tests were conducted to evaluate the fidelity of the synthetic data and its predictive power. The framework was composed of three modules: (1) the data generation module, (2) the forecasting module, and (3) the testing module Fig. 2 provides an overview of the proposed framework. More details about the components of the framework are presented in the following subsections.

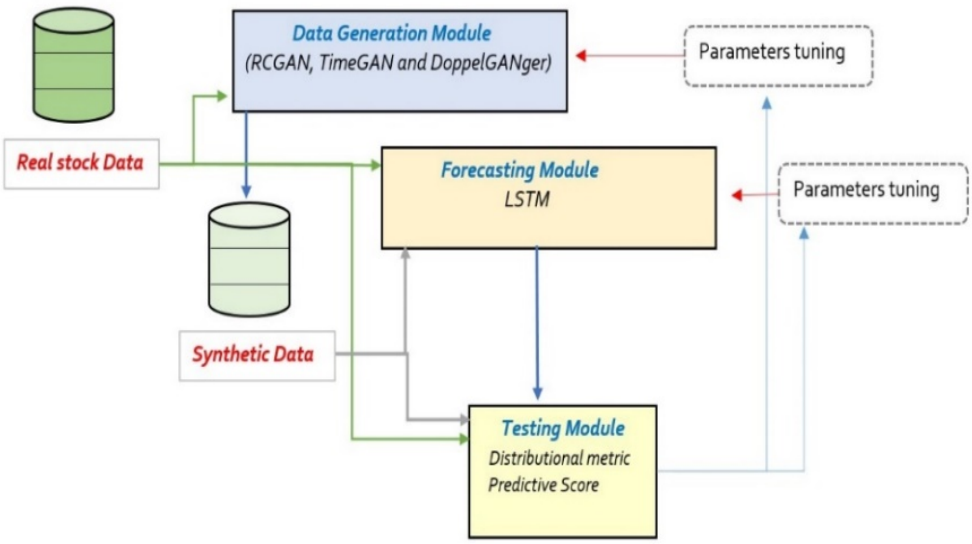


Fig. 2. Proposed framework

2.2.1 Data generation module

GANs [26] are a class of unsupervised learning algorithms composed of two deep adversarial systems: a generator G that tries to map a random noise *z*, drawn from a Gaussian, to a sample of real data *x* and a discriminator D that tries to distinguish between the real data and the generated one. The two systems are simultaneously trained in a two-player zero-sum game with a value function *V (G, D):*

$$minG\ maxD\ V\,(G, D) = Ex \sim Pdata(x)\,[logD(x)] + Ez \sim Pz(z)[log(1 - D(G(Z))] \quad (2)$$

Here, *D(x)* is the probability attributed by the discriminator that the real examples *x* are real. *D(G(Z))* is the probability attributed by the discriminator that a fake example is real. *E* is the expected value.

It is worth mentioning that training a GAN is a tricky task. A GAN can easily fail to converge because of many reasons such as vanishing gradients or mode collapse problems. To overcome GAN training issues, recent works have proposed modifications to the objective function to ensure smooth training [27].

We used GANs as data generation techniques to create synthetic training instances. We tested and compared three state-of-the-art generative algorithms: DoppelGANger (DG) https://github.com/fjxmlzn/DoppelGANger, TimeGAN https://github.com/jsyoon0823/TimeGAN, and Recurrent conditional GAN (RCGAN) https://github.com/ratschlab/RGAN. RCGAN [28] is a GAN variant that produces time series data. In RCGAN, both the generator (Fig. 3) and the discriminator are LSTM conditioned with auxiliary information.

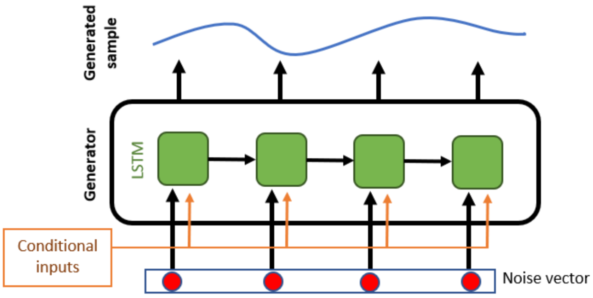


Fig. 3. Architecture of the RCGAN's generator

The conditional vector gave more context to the generator to produce data with properties similar to the real ones. The setting of RCGAN hyperparameters is shown in Table 1.

Table 1. Key hyperparameters of the RCGAN model

Hyperparameters	Values
RNN	LSTM
The number of layers of the generator/discriminator	3/3
The number of units in each layer of the generator/discriminator	20/20
Conditional inputs size/values	3 / $X_{t,High}$, $X_{t,Low}$, $X_{t,Open}$

TimeGAN [29] is basically made of four functions: recovery function, embedding function, sequence discriminator, and sequence generator. The training steps were performed in a way to make TimeGAN simultaneously learn the temporal dependencies, encode features, and generate representations Fig. 4 shows the main components of the TimeGAN architecture. The Key hyperparameters of the TimeGAN model could be seen in Table 2.

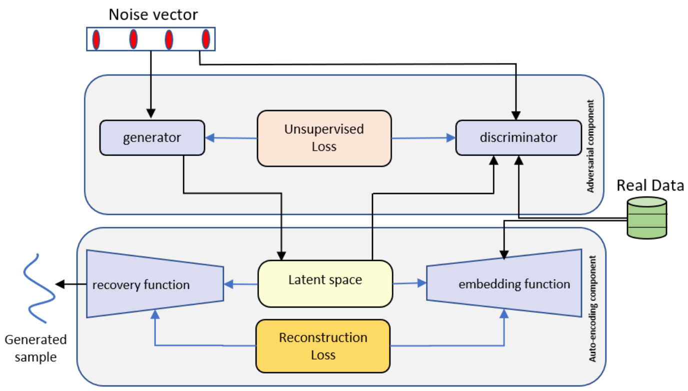


Fig. 4. Architecture of TimeGAN

Table 2. Key hyperparameters of the TimeGAN model

Hyperparameters	Values
Embedding network, generator, and discriminator	LSTM
The number of layers of the generator/discriminator/embedding	3/3/4
The number of units in each layer of the generator/discriminator/embedding	20/20/30
Latent space dimension	2

DG (Fig. 5) used recurrent neural networks (LSTM) as generators to capture long-term dependency [30]. It also used some innovative ideas to tackle the above-mentioned shortcoming of classical GAN architecture: (1) Bach-generation: To make the LSTM more efficient in generating sequential samples, instead of generating one record, DG produced S records at each pass. (2) Auto-normalization: To prevent mode collapse, the minimum/maximum of each time series was learned and generated by an independent generator and used as conditional input. (3) Attributes generation: Correlations between the time series and their attributes were captured by another generator that learned and generated the time series attributes used as conditioning inputs. This procedure differed from classical GAN approaches, where attributes and features are generated in the same step.

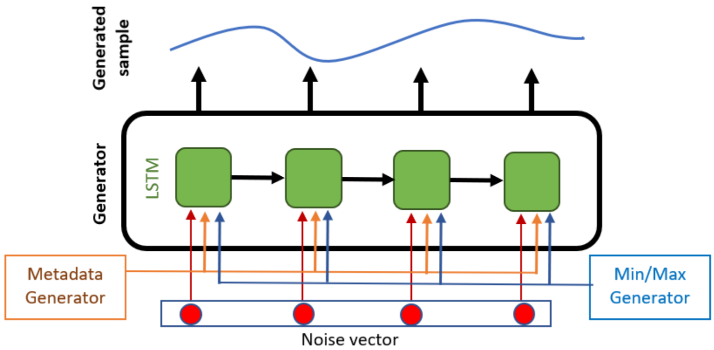


Fig. 5. Architecture of the DG's generator

To improve the fidelity of the generated distribution, DG also used an auxiliary discriminator that discriminated only on metadata. DG model hyperparameters setting are shown in Table 3.

Table 3. Key hyperparameters of the DG model

Hyperparameters	Values
S records generated by the bach-generation	4
Minimum/maximum generator: Multilayer perceptron (MLP) (input units:hidden units:output units) input units: minimum/maximum	2:4:2
Metadata generator: MLP (input units:hidden units:output units) input units: $X_{t,High}$, $X_{t,Low}$, $X_{t,Open}$,	3:6:4
Number of layers in the time series generator (LSTM)	3
Number of units in each layer of the time series generator (LSTM)	20

The noise feed to the generators was a vector of random values drawn from a Gaussian distribution. The dimension of this vector was a hyperparameter set to 10 in our experiments.

2.2.2 Forecasting module

For prediction purposes, the forecasting module used an LSTM. The LSTM cell (Fig. 6) had three gates: an input gate, a forget gate, and an output gate to add or remove information.

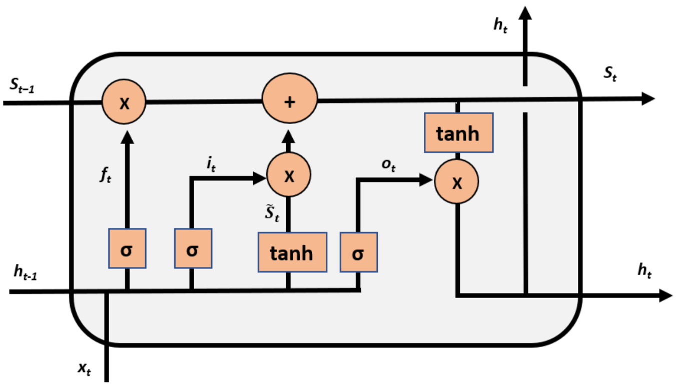


Fig. 6. Architecture of the LSTM's Cell

The cell state, indicated by « S_t », represented the internal final memory that stored short-term and long-term dependencies, and $\tilde{S}_t$ was the new intermediary state of the memory cell. To minimize the prediction error, the weights between the hidden layers were adjusted during the learning step of the LSTM. The outputs h_t were new features that captured the essential information from input features. Here, i_t, o_t, and f_t are the outputs of the cell gates. The LSTM Model hyperparameters are shown in Table 4. All the experimental computations were coded using Python within the TensorFlow framework.

Table 4. Key hyperparameters of the LSTM model

Hyperparameters	Values
Number of layers	3
Number of hidden units per layer	50/100/150
Number of training epochs	300
Training algorithm	Adam optimizer
Learning rate	0.001
Dropout layers	2
Dropout probability	50%
Mini-batch size	50

2.2.3 Testing module

To assess the quality of the generated data and the forecasting results, the testing module used two main criteria: (1) distributional diversity generated samples should be distributed like the real data (even the extreme events) and (2) predictive power generated data should be as informative as the real one and useful in training models for predictive tasks. To assess these criteria, we considered the following metrics:

- Distributional metric [31]

To assess criteria (1), we compared the distributions of synthetic and real data. The two distributions had to be as close as possible. We assumed a binning $S_h = (S_1, \ldots, S_K)$ such that about n records of the observed data $X_{1:T}$ were in each bin. The empirical probability density function of the observed series $\hat{f}_h : S_h \rightarrow R \geq 0$ and the synthetic $\hat{f}_g : S_h \rightarrow R \geq 0$ could then be defined. The absolute difference between these functions was

$$\Sigma B \in Bh|\, fh(S) - fg(S)| \tag{3}$$

This quantity measured how the two distributions were close to each other. The smaller this quantity, the closer the distributions were.

- Predictive Score [29]

By using additional synthetic samples, we enhanced the predictive power of the real data. So, we first compared the accuracy of the prediction of the same model (LSTM used in the forecasting module) on the real data, then on the synthetic data. To do so, two settings were tested: train on real and test on real (TRTR) and train on synthetic and test on real (TSTR). The TSTR procedure consisted of the addition of the generated synthetic examples to the training dataset already containing real examples.

This procedure enriched the training datasets with more examples. The tests were always performed on real data [28]. Since we were comparing continuous real-valued times series, we needed to use metrics that captured the values of differences between real (y_i) and generated data $(\hat{y}_i)$. Hence, errors were measured by using mean absolute error (MAE), mean absolute percentage error (MAPE), and mean squared error (MSE). MAE was used to measure, on average, how close the predictions were to the outcomes. MAPE was the average percentage of errors. Finally, MSE measured the average squares of the error. It gave more importance to the big errors.

$$MAE = \frac{1}{n}\sum_{i=0}^{n} |\, y_i - \hat{y}_i \,| \tag{4}$$

$$MAPE = \frac{100}{n}\sum_{i=0}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (5)$$

$$MSE = \frac{1}{n}\sum_{i}^{n}(y_i - \hat{y})^2 \quad (6)$$

2.3. Experimental Data

Generally, stock market analysts use a wide range of technical indicators as handcrafted features to predict future evolutions of the stock price. For this framework, we chose the following attributes as input features for the data generation and forecasting modules: $X_t = (X_{t,High}, X_{t,Low}, X_{t,Open}, Y_{t,Close})$, referring to the highest, lowest, opening, and closing prices, respectively, recorded on trading day t. Table 5 shows examples of these data. The forecasting module aimed to accurately predict $\hat{Y}_{T+1}, \hat{Y}_{T+2}, \ldots, \hat{Y}_{T+k}$, by training an LSTM using real and generated (synthetic) data, where k is the forecasting horizon. In our study, for the sake of simplification, this parameter was set to 1.

Table 5. Examples of the studied data (Brazil stock index)

Date	Open	High	Low	Close
02-01-2018	76403	77909	76403	77891
03-01-2018	77889	78414	77602	77995
04-01-2018	77998	79135	77998	78647
05-01-2018	78644	79071	78218	79071

2.3.1. Data description

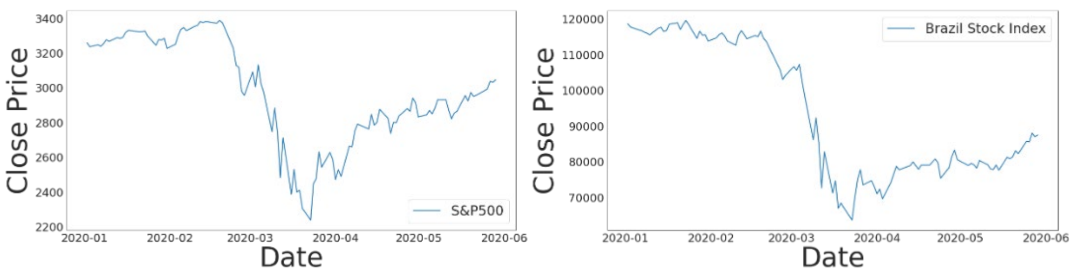
To demonstrate the utility of using synthetic data for stock market forecasting and, more specifically, for extreme variations, we applied our framework to real data from the S&P 500 and five emerging markets: Brazil, India, Malaysia, the Philippines, and Russia. The choice of these data allowed a comparison based on a wide range of market profiles.

Our dataset covered 4 years from 2018 to 2021, corresponding with more than 730 data points for each stock. The historical data were obtained from finance.yahoo.com.

Table 6 presents a summary of some descriptive statistics related to the studied datasets. The standard deviation (SD) measured how the data were spread around the mean Fig. 7 shows examples of the extreme behavior of the studied indices during the spread of the COVID-19 pandemic. The prices of these indices experienced a significant drop during the early days of the pandemic.

Table 6. Descriptive statistics of the studied indices (closing prices)

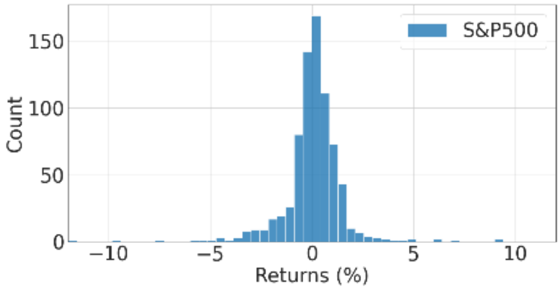
Statistics	Brazil	India	Malaysia	Philippines	Russia	S&P500
Count	741	735	732	728	740	755
Mean	93705	37213	1640	7333	2629	2958
SD	12397	3302	130	879	287	286
Min	63570	25981	1219	4623	2090	2237
Max	119528	47746	1895	9058	3289	3735



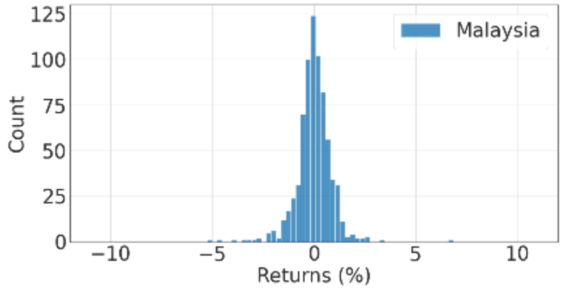
(a) S&P extreme variations (b) Brazil stock index extreme variations

Fig. 7. Examples of indices with extreme variations

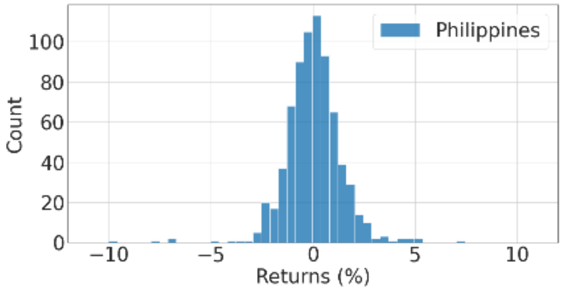
Fig. 8 illustrates how the returns are spread around the means, with extreme values being far from the center of the distributions.



(a) S&P returns



(b) Brazil stock index returns



(c) Phillippines stock index returns

Fig. 8. Examples of returns as histogrammes of the studied indices

2.3.2. Data preparation

Dataset splitting: The dataset was split into two parts: two-thirds of the records were used for the training step and the remaining observations were used for testing (robustness check) and comparing the tested models.

Data standardization: The standardization step was crucial to speed up the learning process. In our study, we standardized the data in a way that their distribution would have a mean value of 0 and an SD of 1, as illustrated in equation (5):

$$x' = x - \mu/\sigma \tag{5}$$

Where μ and σ are the mean value and the SD, respectively, of the training sample prices.

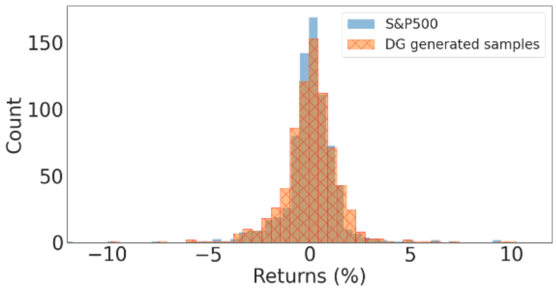
3. Results and Discussion

3.1. Experimental Results

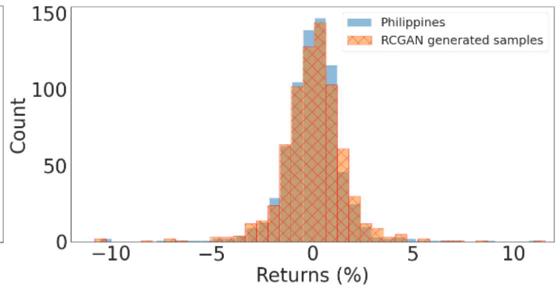
The output of the generation step was sets of synthetic data resembling the real ones with the same entries (features). The visual investigation of the histograms in Fig. 9 shows the high similarity between the distributions of the generated and real data. In addition, the synthetic data show real-looking extreme values between the ±5% and ±10% returns. Since there is a lack of space, each index Fig. 9 shows only the results for the best generation model from the three models tested.

The figure also shows that the DG algorithm produced the best real-looking data, followed by RCGAN. TimeGAN's output was similar to the original data in only the bulk of the distributions, but, in our dataset, it failed to replicate the characteristics of the tails of distributions.

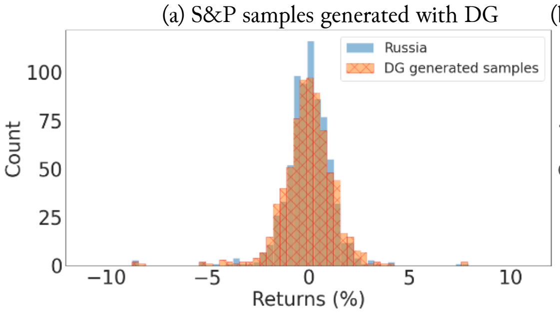
These results can be explained by the conditioning attributes offered by the DG algorithm that better captured the underlying dynamics of training data.



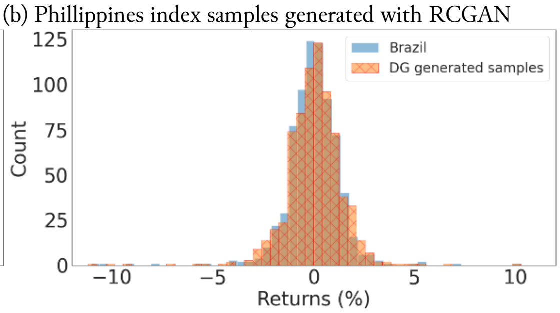
(a) S&P samples generated with DG



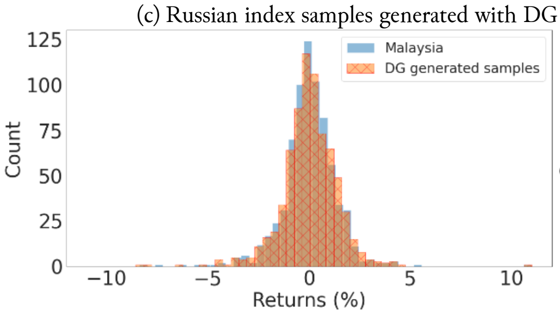
(b) Phillippines index samples generated with RCGAN



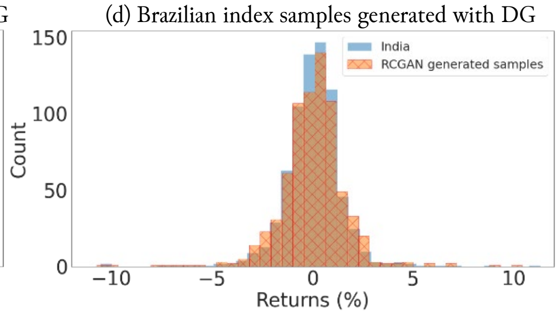
(c) Russian index samples generated with DG



(d) Brazilian index samples generated with DG



(e) Malaysian index samples generated with DG



(f) Indian index samples generated with RCGAN

Fig. 9. Histograms of the generated samplesUnits

Quantitatively, the measure of similarity between the distributions, reported in Table 7, shows that DG and RCGAN give the best distributional scores. In other words, the data generated by RCGAN and DG look more like real data in terms of the shape of their distributions. Unlike DG and RCGAN, TimeGAN did not allow any conditioning for the generation step; this can explain the relatively low quality of the produced data.

Table 7. Distributional scores

	S&P 500	Brazil	India	Malaysia	Philippines	Russia
TimeGAN	0.0472 ±0.0003	0.0642 ±0.0002	0.0763 ±0.003	0.0891 ±0.001	0.0303 ±0.003	0.0601 ±0.002
RCGAN	0.0353 ±0.0002	0.0612 ±0.0003	**0.0388** ±0.001	0.0799 ±0.002	**0.0478** ±0.002	0.0588 ±0.001
DG	**0.0287** ±0.0001	**0.0587** ±0.0001	0.0407 ±0.002	**0.0678** ±0.002	0.0509 ±0.002	**0.0418** ±0.002

The predictive power of the generated data was assessed through their usefulness for prediction purposes. LSTMs with the same architecture (see key hyperparameters in Table 4) were trained and tested via the TRTR and TSTR schemes, as described in subsection 2.2.3.

Table 8 reports how synthetic data improved the performance of the prediction algorithm (LSTM) regarding the accuracy of prediction, which was evaluated by MAE, MAPE, and MSE values between the TRTR and TSTR settings. This improvement was due to the richness (more diverse examples) and fidelity of the synthetic data produced by the framework. The outperformance of the RCGAN and DG was confirmed again by all three assessment criteria.

Table 8. Predictive scores

Index	Model	MAPE %		MAE		MSE	
		TSTR	*TRTR*	*TSTR*	*TRTR*	*TSTR*	*TRTR*
S&P 500	TimeGAN	1.84	2.07	3.05	4.85	0.0036	0.0059
	RCGAN	1.75	2.07	2.60	4.85	0.0029	0.0059
	DG	**1.39**	2.07	**1.31**	4.85	**0.0017**	0.0059
Brazil	TimeGAN	1.89	2.75	3.90	4.50	0.0026	0.0067
	RCGAN	1.74	2.75	2.99	4.50	0.0019	0.0067
	DG	**1.60**	2.75	**2.06**	4.50	**0.0012**	0.0067
India	TimeGAN	1.67	2.31	3.75	4.64	0.0032	0.0052
	RCGAN	**1.35**	2.31	**2.60**	4.64	**0.0019**	0.0052
	DG	1.59	2.31	1.43	4.64	0.0018	0.0052
Malaysia	TimeGAN	1.64	2.56	3.45	4.45	0.0025	0.0061
	RCGAN	1.51	2.56	2.47	4. 45	0.0016	0.0061
	DG	**1.39**	2.56	**1.23**	4. 45	**0.0013**	0.0061
Philippines	TimeGAN	1.86	2.87	4.32	4.66	0.0045	0.0088
	RCGAN	**1.32**	2.87	**3.31**	4.66	**0.0023**	0.0088
	DG	1.41	2.87	3.97	4.66	0.0032	0.0088
Russia	TimeGAN	1.87	2.21	3.35	4.34	0.0036	0.0064
	RCGAN	1.55	2.21	2.34	4.34	0.0025	0.0064
	DG	**1.18**	2.21	**1.84**	4.34	**0.0018**	0.0064

3.2. Robustness check

To check the robustness of our results and the generalization of our framework regarding the extreme variations modeling, we performed specific examinations on the independent test dataset (one-third of the original dataset). We considered extreme events as the movements of stock prices smaller or greater than the following quantity:

$$\mu \pm 2 * \sigma \tag{6}$$

where μ and σ are the mean value and the SD, respectively, of the training sample variations. Once these data points were identified, we specifically compared the MAE, MAPE, and MSE metrics of the predictions of these events in terms of the TSTR and TRTR procedures. Table 8 presents the mean values of the three tested algorithms.

Table 9. Predictive-score results for extreme events

Index	MAPE %		MAE		MSE	
	TSTR	*TRTR*	*TSTR*	*TRTR*	*TSTR*	*TRTR*
S&P 500	1.55	2.55	2.45	4.45	0.0026	0.0053
Brazil	1.17	2.86	2.95	4.67	0.0028	0.0061
India	1.72	2.16	2.31	5.05	0.0024	0.0057
Malaysia	1.97	2.37	2.74	5.34	0.0021	0.0067
Philippines	1.18	2.77	3.33	5.15	0.0025	0.0082
Russia	1.36	2.48	2.18	4.88	0.0020	0.0073

The results in Table 8 indicate a clear improvement in the forecasting of extreme events by using synthetic data. The quality of extreme event forecasting is almost as good as the prediction of the points from the bulk of the distribution (Table 8).

4. Conclusion

In this work, we developed a framework based on GAN algorithms to simulate extreme stock-market variations by generating synthetic data. This framework can efficiently capture the underlying dynamic of stock data and help tackle the problem of extreme event–data scarcity. Experiments performed on the data from the S&P 500 and five emerging markets show that adding synthetic data to the training process improves the prediction accuracy of extreme events. The quality of the generated data was evaluated by quantitative and qualitative metrics. The results of this study have significant practical implications for investors, traders, and corporations willing to anticipate the future trends of their financial assets. The proposed framework fills the gap in the usage of GAN for stock market simulation and it is a tool to mimic the stock market's extreme behaviors. For future research, we suggest testing more GAN architectures and using advanced techniques for hyperparameter tuning. We also suggest diversifying the metrics used to assess the generated data. Trading strategies can also be simulated on the generated data to verify to which extent the framework can be profitable.

Declarations

Author contributions: All the authors contributed equally to the writing of this paper. All the authors have read and approve the final paper.
Funding statement: None of the authors have received any funding or grants from any institution or funding body for the research.
Conflict of interest: The authors declare no conflict of interest.
Additional information: No additional information is available for this paper.

References

[1] S. Ghosh, "COVID-19, stock market, exchange rate, oil prices, unemployment, inflation, geopolitical risk nexus, the case of the BRICS nations: Evidence quantile regression," *Impact Glob. Issues Int. Trade*, pp. 86–105, Jun. 2021, doi: 10.4018/978-1-7998-8314-2.CH005.

[2] P. F. Dai, X. Xiong, Z. Liu, T. L. D. Huynh, and J. Sun, "Preventing crash in stock market: The role of economic policy uncertainty during COVID-19," *Financ. Innov.*, vol. 7, no. 1, pp. 1–15, Dec. 2021, doi: 10.1186/S40854-021-00248-Y.

[3] H. Liu, A. Manzoor, C. Wang, L. Zhang, and Z. Manzoor, "The COVID-19 Outbreak and Affected Countries Stock Markets Response," *Int. J. Environ. Res. Public Heal. 2020,* vol. 17, no. 8, p. 2800, Apr. 2020, doi: 10.3390/IJERPH17082800.

[4] M. Mazur, M. Dang, and M. Vega, "COVID-19 and the march 2020 stock market crash. Evidence from S&P1500," *Financ. Res. Lett.*, vol. 38, p. 101690, Jan. 2021, doi: 10.1016/J.FRL.2020.101690.

[5] M. Uddin, A. Chowdhury, K. Anderson, and K. Chaudhuri, "The effect of COVID – 19 pandemic on global stock market volatility: Can economic strength help to manage the uncertainty?," *J. Bus. Res.*, vol. 128, pp. 31–44, May 2021, doi: 10.1016/J.JBUSRES.2021.01.061.

[6] I. Shaikh, "Impact of COVID-19 pandemic disease outbreak on the global equity markets," Econ. Res. Istraživanja, vol. 34, no. 1, pp. 2317–2336, 2021, doi: 10.1080/1331677X.2020.1863245.

[7] A. Coletta *et al.*, "Towards Realistic Market Simulations: A Generative Adversarial Networks Approach," *ICAIF 2021 - 2nd ACM Int. Conf. AI Financ.*, pp. 1-9, Nov. 2021, doi: 10.1145/3490354.3494411.

[8] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques – Part II: Soft computing methods," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5932–5941, Apr. 2009, doi: 10.1016/J.ESWA.2008.07.006.

[9] C. L. Chang, M. McAleer, and Y. A. Wang, "Herding behaviour in energy stock markets during the Global Financial Crisis, SARS, and ongoing COVID-19*," *Renew. Sustain. Energy Rev.*, vol. 134, p. 110349, Dec. 2020, doi: 10.1016/J.RSER.2020.110349.

[10] N. Engelhardt, M. Krause, D. Neukirchen, and P. N. Posch, "Trust and stock market volatility during the COVID-19 crisis," *Financ. Res. Lett.*, vol. 38, p. 101873, Jan. 2021, doi: 10.1016/J.FRL.2020.101873.

[11] M. Ambros, M. Frenkel, T. L. D. Huynh, and M. Kilinc, "COVID-19 pandemic news and stock market reaction during the onset of the crisis: evidence from high-frequency data,", vol. 28, no. 19, pp. 1686–1689, 2020, doi: 10.1080/13504851.2020.1851643.

[12] W. Jiang, "Applications of deep learning in stock market prediction: Recent progress," *Expert Syst. Appl.*, vol. 184, p. 1-97, Dec. 2021, doi: 10.1016/J.ESWA.2021.115537.

[13] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nat. 2015 5217553*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[14] S. C. Huang, C. F. Wu, C. C. Chiou, and M. C. Lin, "Intelligent FinTech Data Mining by Advanced Deep Learning Approaches," *Comput. Econ.*, vol. 59, no. 4, pp. 1407–1422, Apr. 2022, doi: 10.1007/S10614-021-10118-5.

[15] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, Oct. 2018, doi: 10.1016/J.EJOR.2017.11.054.

[16] D. Murekachiro, T. Mokoteli, and H. Vadapalli, "Predicting emerging and frontier stock markets using deep neural networks," *Adv. Intell. Syst. Comput.*, vol. 1037, pp. 899–918, 2020, doi: 10.1007/978-3-030-29516-5_68.

[17] A. E. de Oliveira Carosia, G. P. Coelho, and A. E. A. da Silva, "Investment strategies applied to the Brazilian stock market: A methodology based on Sentiment Analysis with deep learning," *Expert Syst. Appl.*, vol. 184, pp. 1-12, Dec. 2021, doi: 10.1016/J.ESWA.2021.115470.

[18] W. Lu, J. Li, J. Wang, and L. Qin, "A CNN-BiLSTM-AM method for stock price prediction," *Neural Comput. Appl.*, vol. 33, no. 10, pp. 4741–4753, May 2021, doi: 10.1007/S00521-020-05532-Z.

[19] J. Engelmann and S. Lessmann, "Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning," *Expert Syst. Appl.*, vol. 174, p. 114582, Jul. 2021, doi: 10.1016/J.ESWA.2021.114582.

[20] K. Armanious *et al.*, "MedGAN: Medical image translation using GANs," *Comput. Med. Imaging Graph.*, vol. 79, p. 101684, Jan. 2020, doi: 10.1016/J.COMPMEDIMAG.2019.101684.

[21] M. K. Baowaly, C. C. Lin, C. L. Liu, and K. T. Chen, "Synthesizing electronic health records using improved generative adversarial networks," *J. Am. Med. Informatics Assoc.*, vol. 26, no. 3, pp. 228–241, Mar. 2019, doi: 10.1093/JAMIA/OCY142.

[22] G. Ramponi, P. Protopapas, M. Brambilla, and R. Janssen, "T-CGAN: Conditional Generative Adversarial Network for Data Augmentation in Noisy Time Series with Irregular Sampling," pp.1-12 Nov. 2018,. doi: 10.48550/arXiv.1811.08295.

[23] S. Takahashi, Y. Chen, and K. Tanaka-Ishii, "Modeling financial time-series with generative adversarial networks," *Phys. A Stat. Mech. its Appl.*, vol. 527, pp. 1-12, Aug. 2019, doi: 10.1016/J.PHYSA.2019.121261.

[24] W. Tovar, "Deep Learning Based on Generative Adversarial and Convolutional Neural Networks for Financial Time Series Predictions," Aug. 2020, Accessed: May 02, 2023. doi: 10.48550/arXiv.2008.08041.

[25] K. Zhang, G. Zhong, J. Dong, S. Wang, and Y. Wang, "Stock Market Prediction Based on Generative Adversarial Network," *Procedia Comput. Sci.*, vol. 147, pp. 400–406, Jan. 2019, doi: 10.1016/J.PROCS.2019.01.256.

[26] I. Goodfellow *et al.*, "Generative Adversarial Networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Jun. 2014, doi: 10.1145/3422622.

[27] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved Training of Wasserstein GANs," *Adv. Neural Inf. Process. Syst.*, vol. 2017-December, pp. 5768–5778, Mar. 2017, Accessed: May 02, 2023. doi: 10.48550/arXiv.1704.00028.

[28] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs," pp. 1-13, Jun. 2017, Accessed: May 02, 2023. doi: 10.48550/arXiv.1706.02633.

[29] J. Yoon, D. Jarrett, and M. van der Schaar, "Time-series Generative Adversarial Networks," *Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 1-11, 2019. Available at: https://papers.nips.cc/paper/8789-time-series-generative-adversarial-networks.

[30] Z. Lin, A. Jain, C. Wang, G. Fanti, and V. Sekar, "Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions," *Proc. ACM SIGCOMM Internet Meas. Conf. IMC*, vol. 2020-Oct, pp. 464–483, doi: 10.1145/3419394.3423643.

[31] M. Wiese, L. Bai, B. Wood, and H. Buehler, "Deep Hedging: Learning to Simulate Equity Option Markets," *SSRN Electron. J.*, Oct. 2019, pp.1-13, doi: 10.2139/SSRN.3470756.