mathematics

MDPI

Article

Probabilistic Forecasting of Crude Oil Prices Using Conditional Generative Adversarial Network Model with Lévy Process

Mohammed Alruqimi * and Luca Di Persio

Department of Computer Science, College of Mathematics, University of Verona, 37129 Verona, Italy; luca.dipersio@univr.it

* Correspondence: mohammed.alruqimi@univr.it

Abstract: Accurate crude oil price forecasting is essential, considering oil's critical role in the global economy. However, the crude oil market is significantly influenced by external, transient events, posing challenges in capturing price fluctuations' complex dynamics and uncertainties. Traditional time series forecasting models, such as ARIMA and LSTM, often rely on assumptions regarding data structure, limiting their flexibility to estimate volatility or account for external shocks effectively. Recent research highlights Generative Adversarial Networks (GANs) as a promising alternative approach for capturing intricate patterns in time series data, leveraging the adversarial learning framework. This paper introduces a Crude Oil-Driven Conditional GAN (CO-CGAN), a hybrid model for enhancing crude oil price forecasting by combining advanced AI frameworks (GANs), oil market sentiment analysis, and stochastic jump-diffusion models. By employing conditional supervised training, the inherent structure of the data distribution is preserved, thereby enabling more accurate and reliable probabilistic price forecasts. Additionally, the CO-CGAN integrates a Lévy process and sentiment features to better account for uncertainties and price shocks in the crude oil market. Experimental evaluations on two real-world oil price datasets demonstrate the superior performance of the proposed model, achieving a Mean Squared Error (MSE) of 0.000054 and outperforming benchmark models.

Keywords: crude oil-driven conditional generative adversarial network; Lévy–Merton jump-diffusion model; oil price forecasting; long short-term memory

MSC: 68T07

check for updates

Academic Editor: Snezhana Gocheva-Ilieva

Received: 10 December 2024
Revised: 13 January 2025
Accepted: 16 January 2025
Published: 18 January 2025

Citation: Alruqimi, M.; Di Persio, L. Probabilistic Forecasting of Crude Oil Prices Using Conditional Generative Adversarial Network Model with Lévy Process. *Mathematics* **2025**, *13*, 307. https://doi.org/10.3390/math13020307

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Despite the significant success achieved by deep learning models in various fields, particularly in computer vision and natural language processing, their success in time series forecasting remains limited. Time series forecasting presents unique challenges primarily due to the high degree of uncertainty and the intricate dynamics influenced by external factors. These challenges are particularly amplified in the case of crude oil price forecasting, where the inherent volatility and sensitivity of oil prices to global events make the uncertainty and fluctuations even more pronounced. The fluctuations in oil prices directly impact the cost of production, transportation, and goods, influencing inflation and overall economic stability. Therefore, accurate forecasting of oil prices has been a critical concern over recent decades. Statistical methods such as autoregressive integrated moving average (ARIMA) models were commonly used in the early stages of oil price forecasting. These methods are built on mathematical assumptions and rely on linear relationships within the data, making them effective for simple and stable patterns. However, as the need to capture

the intricate and non-linear dynamics of oil price movements, the advent of deep learning (DL) models emerged as a more powerful alternative [1]. In the past decade, numerous machine learning (ML) models have been employed to enhance forecasting accuracy [2]. Recurrent Neural Networks (RNNs), particularly Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) networks, have gained popularity for predicting oil prices due to their ability to capture temporal dependencies [3]. However, several limitations arise due to the complex and volatile nature of the energy markets. These models may struggle to optimally integrate multiple exogenous data and capture sudden jumps in the data caused by exogenous factors. This can lead to challenges such as the model overfitting to noise rather than accurately capturing underlying patterns, especially when working with small or noisy financial datasets. Transformer-based models, which excel at capturing long-term dependencies, have gradually become popular in time series forecasting as well [4]. However, Transformers typically require larger datasets to be trained efficiently, and their performance in time series data is questionable [5]. The attention mechanism of the Transformer is somewhat against order, which is critical for time series forecasting. GANs [6] have gained increasing attention in the literature. GANs offer a novel approach with the potential for superior performance, mainly due to their advanced pattern recognition capabilities. These cutting-edge neural network architectures mitigate overfitting by introducing Gaussian-mixture noise into the generator network during training. In the past few years, many GAN models have been adapted for time series applications such as TimeGAN [7], COT-GAN [8], and Time-Variant GAN [9]. These models have been primarily designed for general time series data augmentation rather than forecasting. On the other hand, there are a few models that have been designed explicitly for forecasting purposes and applied in the context of financial time series data (i.e., ForGAN [10] and FinGAN [11]). In this paper, we introduce a crude oil-driven Conditional Generative Adversarial Network (CO-CGAN) to generate probabilistic forecasting of crude oil prices. Our model leverages the ForGAN architecture [10], which combines GANs with recurrent neural networks. This architecture renders GANs more suitable for forecasting tasks and places them into a supervised learning setting whilst producing conditional probability distributions of oil prices given previous historical observations. Our model (CO-CGAN) allows uncertainty estimates beyond the traditional use of point estimates in the forecasting literature. We address the challenge of insufficient data representing rare events affecting price fluctuations by integrating a Lévy–Merton jump process and a sentiment score feature derived from crude oil market sentiment analysis. These components enrich the CGAN model's context, allowing it to better capture the uncertainty and volatility dynamics of oil prices driven by external factors. Given the critical role of hyperparameter tuning in achieving optimal training, we employed the Grey Wolf Optimizer (GWO) to optimize the model's hyperparameters efficiently.

Extensive experiments were conducted on two real-world datasets, Brent and WTI. To evaluate the proposed CO-CGAN model performance, we compared our model to ForGAN, TimeGAN, and two other benchmark models specifically designed for this purpose based on Conditional GAN (CGAN) and Wasserstein-GAN (WGAN).

Standard point-wise evaluation metrics have been used, including MSE, RMSE, MAE, MAPE, and MSPE, as well as the statistical measure (R^2) and the Kullback–Leibler divergence (KL). The experiments demonstrate that our model achieves high forecasting accuracy and robustness compared to the benchmark models, achieving an MSE of (0.000054).

2. Generative Adversarial Networks (GANs)

GANs have rapidly gained attention since their introduction in 2014 [6], becoming a prominent area of interest within the deep learning research community. Their success

can be attributed to their ability to generate and manipulate high-quality data across various domains. While GANs have significantly impacted computer vision, particularly in tasks like image and video generation, style transfer, and image completion, their application has extended to other areas, including natural language processing (NLP) and time series analysis. However, ongoing research explores using GANs for time series generation and forecasting [12]. A GAN consists of two primary components: a Generator G and a Discriminator D, typically implemented as neural networks (Figure 1). These two neural networks are trained through a unique adversarial process. The generator G takes in random noise $z \in \mathbb{R}^r$ and creates synthetic data similar to the distribution of the actual training data. At the same time, the discriminator D evaluates the authenticity of the data, distinguishing between actual and generated samples. During this training process, the generator strives to produce increasingly realistic samples capable of deceiving the discriminator. In contrast, the discriminator simultaneously improves its ability to differentiate between real and fake data. This adversarial process continues until the generator produces data that the discriminator can no longer reliably distinguish from the original. As a result, GANs can generate synthetic data that maintain the original dataset's statistical properties and distributional characteristics, making them highly effective for tasks requiring realistic data synthesis. Mathematically, a discriminator D and generator G play the following two-player minimax game with value function $V(G, D)$ [6]:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \quad (1)$$

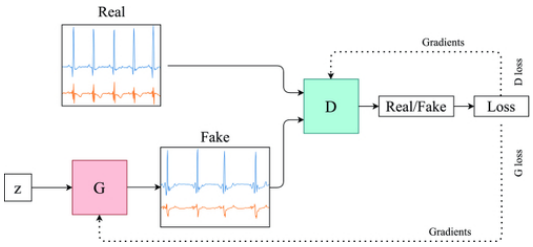


Figure 1. Generative adversarial network [12].

Since their introduction in 2014, numerous GAN variants have been developed to address specific challenges or expand the original framework's capabilities. Notable examples include Conditional GANs (CGANs) [13], which generate data conditioned on auxiliary information such as labels or classes, enabling controlled and targeted data generation. Bidirectional GAN (BiGAN) [14] further extends GANs by simultaneously learning generative and inference models, facilitating the mapping of data points back into the latent space. Other popular variants include Wasserstein GAN (WGAN) [15], which improves training stability by minimizing the Wasserstein distance.

3. Literature Review

Over recent decades, various methodologies have been developed for forecasting oil prices. Initially, statistical and econometric approaches, such as ARIMA (AutoRegressive Integrated Moving Average) and GARCH (Generalized AutoRegressive Conditional Heteroskedasticity), were widely employed. These models often rely on mathematical assumptions and have been effective in capturing linear patterns in time series data. Recently, deep learning models have gained popularity in the literature due to their capabilities in capturing more complex and non-linear patterns in datasets. These models also enable a better incorporation of external factors [1]. Ref. [2] comprehensively compares and analyses a wide range of machine learning-based approaches for energy and financial forecasting. In our previous work [3], we empirically compared deep learning models for crude oil price

forecasting, focusing on approaches based on LSTM, GRU, and Transformer networks, which are widely applied in the literature. LSTM and GRU models have been commonly applied in the literature [16] due to their attribution in maintaining temporal dependencies, even when working with small datasets. Additionally, researchers have investigated combining LSTM and GRU models with other techniques such as Convolutional Neural Networks (CNNs) [17], time series decomposition [16], and attention layers to enhance their performance. Other researchers have explored Transformer-based models, such as Autoformer [18], for general time series forecasting. While these models achieve promising accuracy with high-frequency time series datasets, their performance with oil price datasets is not as impressive. This can be attributed to the fact that the attention mechanism in Transformers does not maintain order, which is critical for this task. Additionally, Transformer training typically requires a larger dataset [3,5]. Generative Adversarial Network (GAN) models have emerged as robust tools for generating synthetic data distributed similarly to the original data. This advancement opens new doors for investigating the use of these models in time series forecasting. In the past few years, several GAN-based models have been introduced to generate synthetic time series data, for instance: TimGAN [7], COT-GAN [8] and Time-Variant GAN [9]. While these models are primarily designed for time series data augmentation, a few other GAN-based models have been proposed for time series forecasting tasks, such as TSF-CGANs [19], ForGAN [10], and Fin-GAN [11]. TSF-CGANs [19] is a time series forecasting model based on CGAN combined with convolutional neural networks (CNNs) and bi-directional long short-term memory (Bi-LSTM). ForGAN is a conditional GAN-based model that was presented in 2019 [10] for probabilistic time series forecasting. This model combines a gated recurrent unit (GRU) or a long short-term memory (LSTM) cell, depending on the data. Both the generator and the discriminator first pass their inputs through a recurrent neural network layer aiming at exploiting the time series structure. The Fin-GAN [11] model employs the ForGAN architecture by introducing an economic-driven loss function for the generator. Examining these models with crude oil price datasets has not yet been investigated well.

4. Proposed Model

The proposed model, built on the ForGAN architecture [10], is a one-step-ahead probabilistic forecasting model applying a conditional generative adversarial network to model the probability distribution of forthcoming crude oil prices based on past observations. We further integrate a Lévy–Merton jump process into the generator and crude oil market sentiment analysis as an additional, conditional input to improve the modelling of price shocks and volatility.

The leading architecture of the model is illustrated in Figure 2. In other words, the objective is to model the probability distribution of one-step-ahead crude oil price x_{t+1} given the historical data $c = x_0, \ldots, x_t$, i.e., $\rho_{(xt+1|c)}$. The historical data include crude oil prices and sentimental score observations.

Generator G and discriminator D are trained simultaneously in an adversarial network. The generator takes a noise vector Z, which is sampled from a Gaussian distribution (ρ_Z), and generates the value of the next step (x_{t+1}) with the condition data ρ_data. Meanwhile, the discriminator receives (x_{t+1}) and determines if (x_{t+1}) is real or generated by the generator. Hence, the model function is expressed as follows:

$$\begin{aligned} &\min_G \max_D V(D,G) = \mathbb{E}_{x_{t+1}\sim\rho_{\text{data}}\,(x_{t+1})}[\log D(x_{t+1} \mid c)] \\ &+ \mathbf{E}_{z\sim\rho_x(z)}[\log(1 - D(G(z \mid c)))] \end{aligned} \tag{2}$$

In other words, the generator G learns to transform a known probability distribution ρ_z into the generator distribution ρ_G, which resembles ρ_{data}.

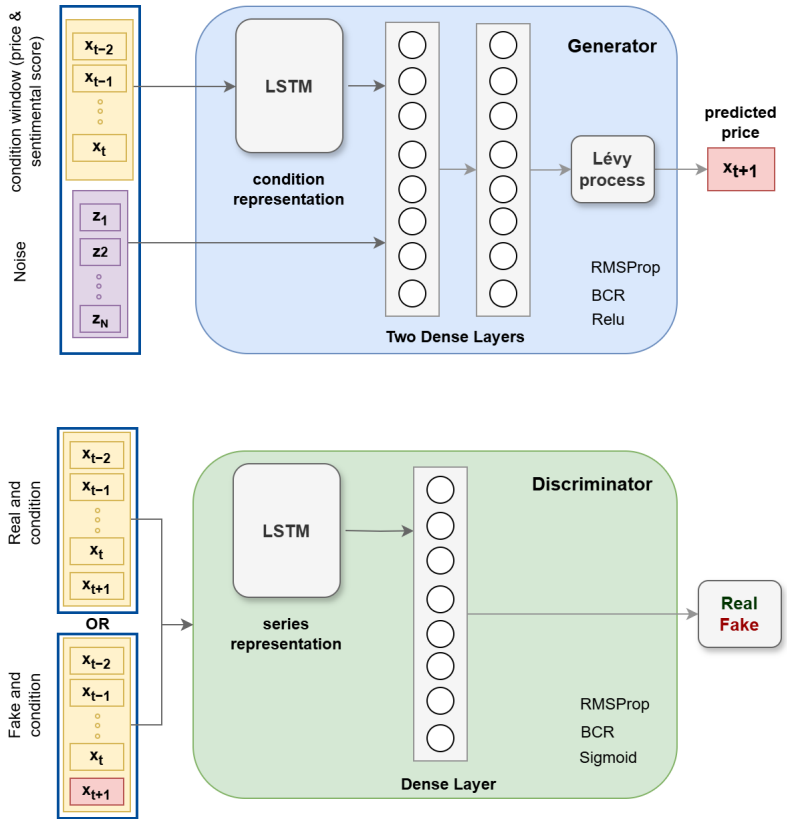


Figure 2. Overview of the proposed model.

4.1. Conditional GAN (CGAN)

Conditional GANs (CGANs) [13], an extension of traditional GANs, provide additional control over the generation process by conditioning the model on supplementary information, denoted as y. This information can be any data, such as labels or data from different sources. By incorporating this conditioning information, CGANs can generate samples that adhere to predefined characteristics, enhancing their ability to generate data from conditional distributions. This results in more targeted and context-specific data generation, which means placing the GAN into a supervised forecasting setting. In other words, rather than having all data samples originate from the same underlying distribution, we assume that the data $\mathbf{x}$ are drawn from a conditional distribution $p_{\text{data}}(\mathbf{x} \mid \text{condition(y)})$, where the distribution depends on a specific deterministic condition—see Figure 2. Both the generator G and the discriminator D are provided with additional deterministic information (condition) alongside the standard inputs. In our case, the sub-sequences (time lags) of oil prices and the sentiment scores represent the labelled data condition. The model will generate (predict) future oil prices based on past observations (time lag). The modified value function $V(G, D)$ for this configuration is the following:

$$\begin{aligned} \min_G \max_D V(D,G) &= \mathbb{E}_{x\sim\rho_{\text{data}}(x)}[\log D(x \mid y)] \\ &+ \mathbb{E}_{z\sim\rho_x(z)}[\log(1 - D(G(z \mid y)))] \end{aligned} \tag{3}$$

4.2. Lévy–Merton Jump-Diffusion Model

To capture the abrupt price changes and high volatility often observed in crude oil prices, we incorporate the Lévy–Merton jump-diffusion model [20] into our model. Hence,

extending the classical Geometric Brownian Motion (GBM) by adding a jump component to capture price volatility, thereby enhancing the accuracy of our forecasts. In particular, let S_t represent the crude oil price at time t, then its evolution under the Merton jump-diffusion model is governed by the following SDE:

$$dS_t = S_t((\mu - \lambda k)dt + \sigma dW_t + dQ_t) \tag{4}$$

where we can see the action of two primary components (see Figure 3):

- Diffusion component: σdW_t, where W_t represents a standard Wiener process, with μ as the drift (expected return) and σ as the volatility. This part captures the *continuous* price changes following a standard GBM.
- Jump component: dQ_t, modelled as a compound Poisson process to account for discontinuous jumps:

$$dQ_t = \sum_{i=1}^{N_t}(Y_i - 1) \tag{5}$$

where N_t is a Poisson process with intensity λ, representing the number of jumps in the interval, and Y_i are i.i.d. random variables representing the jump sizes, which follow a log-normal distribution: $\ln(Y_i) \sim \mathcal{N}(m, \delta^2)$.
Accordingly, the term $k = \mathbb{E}(Y_i - 1) = e^{m+\frac{1}{2}\delta^2} - 1$ represents the expected relative jump size, and the parameter λ is the frequency of jumps. Let us underline that since the expectation of Q_t is

$$\mathbb{E}(Q_t) = \lambda k t, \tag{6}$$

$Q_t - \lambda k t$ is a martingale, preserving the no-arbitrage condition of the model.

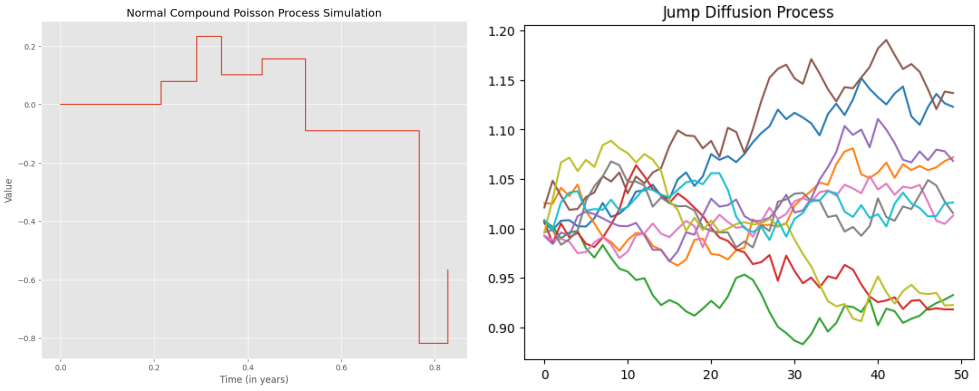


Figure 3. (**Left**): Normal compound Poisson process simulation. λ = 10; T = 1y; jumps $\sim$ N(0, 0.22). Ten jumps per year on average, and magnitude of jumps follows normal distribution N(0, 0.22). (**Right**): Jump paths generated using Merton model (10 paths, 50 steps).

The Lévy–Merton jump-diffusion model is embedded within the CGAN generator to enhance the forecast distribution by incorporating the jump dynamics of crude oil prices. In particular, during training, the jump-diffusion parameters $(\mu, \sigma, \lambda, m, \delta)$ are optimized through the CGAN's neural network gradients, allowing the model to adjust to the complex volatility structure in real time, which implies more accurate probabilistic forecasts and better identification of the risk of extreme price movements.

Essentially, our objective is to generate one-step-ahead forecasts, x_{t+1}, conditioned on historical observations $c = (x_0, x_1, \ldots, x_t)$. Accordingly, we integrate the Lévy–Merton model within the CGAN framework to handle both continuous market fluctuations and discrete jumps, Hence, this ensures a more realistic representation of the crude oil price dynamics.

In particular, let $G(z|c)$ be the generator of the CGAN, where $z \sim \mathcal{N}(0,1)$ represents Gaussian noise that is injected into the network. We modify the generator's output to incorporate the dynamics of the Lévy–Merton model. Indeed, the generator outputs the predicted value $\hat{x}_{t+1}$, where

$$\hat{x}_{t+1} = G(z|c) + J_L(\lambda, m, \delta) \tag{7}$$

with $J_L(\lambda, m, \delta)$ being the Lévy jump term modelling the sudden price shifts, defined as follows:

$$J_L(\lambda, m, \delta) = \sum_{i=1}^{N_t} (Y_i - 1) \tag{8}$$

where $N_t \sim \text{Poisson}(\lambda)$ is the number of jumps in the time interval, and $Y_i \sim \log \mathcal{N}(m, \delta^2)$ are the jump magnitudes. Then, such jumps are added to the generated value to account for extreme events in crude oil prices, while the generator learns both the continuous and jump dynamics during training by optimizing the parameters of the Lévy–Merton model, including λ (jump intensity), m (mean jump size), and δ (jump volatility).

More specifically, since the core of the CGAN training process is the adversarial min–max game between the generator G and the discriminator D with the standard CGAN objective defined as

$$\min_G \max_D V(D,G) = \mathbb{E}_{x_{t+1} \sim \rho_{\text{data}}(x_{t+1})}[\log D(x_{t+1}|c)] + \mathbb{E}_{z \sim \rho_z(z)}[\log(1 - D(G(z|c)))], \tag{9}$$

to integrate the Lévy–Merton jump-diffusion model into this objective, we update the generator to output $\hat{x}_{t+1}$ as follows:

$$\hat{x}_{t+1} = G(z|c) + J_L(\lambda, m, \delta). \tag{10}$$

Thus, the value function becomes

$$\min_G \max_D V(D,G) = \mathbb{E}_{x_{t+1} \sim \rho_{\text{data}}(x_{t+1})}[\log D(x_{t+1}|c)] + \mathbb{E}_{z \sim \rho_z(z)}[\log(1 - D(G(z|c) + J_L(\lambda, m, \delta)))] \tag{11}$$

The discriminator $D(x_{t+1}|c)$ evaluates the authenticity of the predicted value x_{t+1}, determining whether it comes from the true distribution ρ_{data} or is generated by the model $G(z|c) + J_L(\lambda, m, \delta)$. Let us underline that the latter ensures that the generated values include both the continuous price changes (modelled by the CGAN) and the sudden jumps (modelled by the Lévy process).

Moreover, during training, the CGAN learns the optimal parameters of the generator and discriminator. Simultaneously, the parameters of the Lévy–Merton jump-diffusion model—namely, λ, m, and δ—are optimized as part of the generator's output to ensure that the jump component aligns with the observed data distribution. Specifically, the loss function for the generator is updated to include the jump component:

$$\mathcal{L}_G = -\mathbb{E}_{z \sim \rho_z(z)}[D(G(z|c) + J_L(\lambda, m, \delta))] \tag{12}$$

Furthermore, the Lévy process parameters are updated via backpropagation RMSProp, with the gradients derived from the adversarial loss:

$$\frac{\partial \mathcal{L}_G}{\partial \lambda}, \frac{\partial \mathcal{L}_G}{\partial m}, \frac{\partial \mathcal{L}_G}{\partial \delta} \tag{13}$$

with the latter ensuring that the jump magnitudes and frequencies are tuned dynamically during the training process. Hence, the generator can capture typical market behaviour and rare, large-scale events.

The last step consists of considering that the conditional probability distribution generated by the model, i.e., $\rho_G(x_{t+1}|c)$, is a convolution of the generator's output distribution and the Lévy process distribution. Let $\rho_L(x_{t+1})$ represent the distribution induced by the jump process; then, the overall conditional distribution of x_{t+1}, generated by our model, can be expressed as follows:

$$\rho_G(x_{t+1}|c) = \int \rho_G(\hat{x}_{t+1}|c) \cdot \rho_L(x_{t+1} - \hat{x}_{t+1}) d\hat{x}_{t+1}. \quad (14)$$

The latter convolution ensures that the model captures both the smooth price changes modelled by $\rho_G(\hat{x}_{t+1}|c)$ as well as the discontinuous price jumps modelled by $\rho_L(x_{t+1} - \hat{x}_{t+1})$.

4.3. Accumulative Sentimental Score

Accumulative Sentimental Score (SENT) is a cumulative sentimental score derived from the CrudeBERT_Plus model, as introduced in [21]. This model is a specialized variant of FinBERT, fine-tuned to evaluate the influence of market events on crude oil prices. Its development is grounded in the principles of Adam Smith's theory of supply and demand, emphasizing the interplay between market sentiment and pricing dynamics. Market sentiment plays a pivotal role in shaping oil price movements, as it reflects collective expectations and reactions to geopolitical events, economic fluctuations, and market trends. Sentiments expressed in news headlines, reports, and market commentaries can amplify or mitigate future supply and demand perceptions, influencing trading behaviours and price volatility. The dataset used for training CrudeBERT_Plus consists of headlines from 1034 unique news providers. The largest portion of this dataset originates from Dow Jones newswires (approximately 21,200 headlines), followed by Reuters (around 3000 headlines), Bloomberg (approximately 1100 headlines), and Platts (approximately 870 headlines). Details about generating this sentiment score are thoroughly discussed in [21]; the data are publicly available and can be reached through the following link: https://github.com/fhgr/crudeoil-sentiment2022-dataset (accessed on 10 October 2024).

5. Benchmark Models

To assess the predictive performance of our model, we compare it against two established models: ForGAN and TimGAN.

Furthermore, we developed benchmark models based on common GAN architectures adapted for time series forecasting tasks: Conditional GAN (CGAN) and Conditional Wasserstein GAN with Gradient Penalty (CWGAN-GP). The following list provides a comprehensive overview of the benchmark models employed in this evaluation.

1. Conditional GAN (CGAN): This model is built based on the vanilla conditional GAN model [13], with both the generator and discriminator incorporating a Gated Recurrent Unit (GRU). However, this model does not incorporate a Lévy stochastic process.
2. CWGAN-GP Model: We built this model upon the foundational CWGAN-GP (Conditional Wasserstein GAN with Gradient Penalty) architecture, incorporating enhancements to improve its performance and robustness. Specifically, we integrated a Lévy process, as described in Section 4.2, and LipSwish activation. Wasserstein GAN (WGAN) is an improvement over the traditional GAN framework, addressing issues of training stability and mode collapse. By using the Wasserstein distance (also known as Earth Mover's distance) as a measure of distribution similarity, WGAN provides smoother gradients for the generator, leading to more reliable convergence during training. The model comprises the following key components:
   LSTM units: The CWGAN-GP model integrates LSTM units in its generator and critic components. This design leverages the ability of LSTMs to capture long-term

dependencies and complex temporal patterns in sequential data.

Wasserstein Loss: The critic (discriminator) loss uses the Wasserstein loss, which is simply the difference between the real and fake sample scores:

$$d_loss = d_real_loss + d_fake_loss + \lambda_{gp} \cdot gradient_penalty \tag{15}$$

The generator loss is the negative of the discriminator's prediction of the generated data:

$$g_loss = -torch.mean(d_g_decision). \tag{16}$$

Gradient Penalty: The model includes a gradient penalty term to enforce the Lipschitz constraint, which is crucial for WGAN-GP.

No Critic Sigmoid Activation: The critic does not use a sigmoid activation in the output layer. Instead, it outputs raw scores, which are then used to compute the Wasserstein distance.

LipSwish Activation: LipSwish activations are used to constrain the Lipschitz constant of the discriminator [22]. Adding this layer to the generator and the discriminator improves the training stability.

3. ForGAN Model: ForGAN is a conditional GAN designed for probabilistic time series forecasting, introduced by [10]. It leverages a conditional GAN to forecast x_{t+1} given the previous L values of the series $x_t, x_{t-1}, \ldots, x_{t-(L-1)}$. This approach utilizes more information than standard point estimates, facilitating uncertainty estimates. The generator and discriminator use a recurrent neural network layer to exploit the time series structure. ForGAN can include either a GRU or an LSTM cell.
4. TimeGAN: TimeGAN is a generative adversarial network specifically designed for time series data. The model introduced by [7] has garnered significant attention. It utilizes an embedding network to map time series data into a high-dimensional latent space and a recovery network to map it back. This model incorporates a recurrent architecture to capture temporal dependencies, and a joint optimization approach ensures that both the adversarial and reconstruction losses are minimized, leading to more realistic and temporally consistent synthetic time series data.

6. Experiments

To evaluate the performance of our model, we tested it using two real-world datasets that exhibit high fluctuation dynamics. We compared our results with those of state-of-the-art models in the literature. We will introduce the datasets, outline the evaluation methods, describe the experimental setup, and discuss the hyperparameter fine-tuning process.

6.1. Dataset

We used two real-world datasets for the experiments: crude WTI and crude Brent prices. These datasets encompass 2386 observations spanning from 3 January 2012 to 1 April 2021, including the period during which the COVID-19 pandemic impacted the oil and stock markets. To enhance our model for estimating the volatility, we incorporated a daily cumulative sentimental score (SENT) with the two datasets as external features. The SENT feature is obtained from the CrudeBERT model described in [21]. The period from 2012 to 2021 was chosen for analysis due to the availability of sentiment scores during this time frame. Figures 4 and 5 visualize the dataset and the correlation analysis of the three time series (Brent, WTI, and the sentimental index).

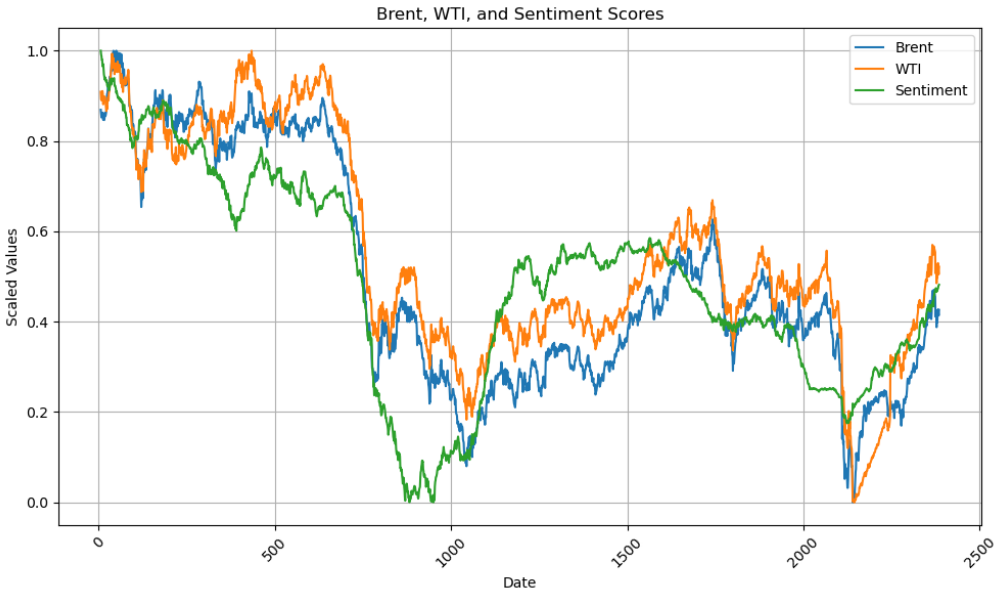


Figure 4. Datasets: WTI, Brent, and the sentimental index.

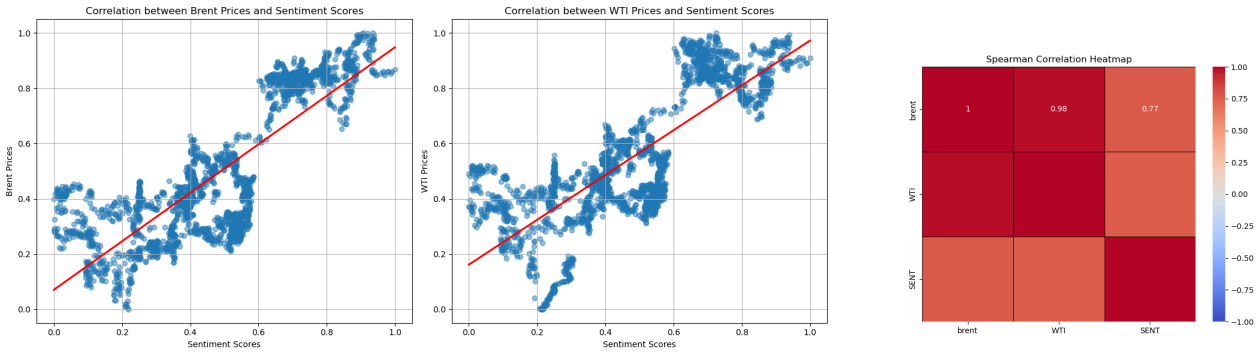


Figure 5. Correlation analysis of the dataset features.

6.2. Evaluation Metrics

Point-wise error metrics are commonly used in forecasting tasks. We report MAE, MSE, RMSE, MSPE, and MAPE to compare our model with state-of-the-art models, in addition to R^2 and KL divergence measures. Below, we define each measure along with its formula, assuming that Y represents the actual values and $\hat{Y}$ represents the predicted values.

- Mean absolute error (MAE): Represents the average of the absolute difference between the actual and predicted values. This measure is less sensitive to outliers.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right| \tag{17}$$

- Mean Squared Error (MSE): Measures the average squared difference between the predicted and actual values. This measure emphasizes larger errors.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{18}$$

- Root Mean Squared Error (RMSE): Square root of MSE.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2} \tag{19}$$

- Mean Squared Prediction Error (MSPE): Quantifies the average squared difference between predicted and actual values.

$$MSPE(Y, \hat{Y}) = \frac{1}{n}\sum_{i}^{n}(Y_i - \hat{Y}_i)^2 \tag{20}$$

- MAPE: Presents the percentage error between actual and predicted values. It is useful for forecasting and percentage-based error analysis.

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \tag{21}$$

- R^2 (Coefficient of Determination): Measures the proportion of variance between the actual and predicted values. A high R^2 (close to 1) means the model very closely predicts the actual values.

$$R^2 = 1 - \frac{SSR}{SST} \tag{22}$$

  where SSR stands for the sum of squared residuals, which measures the sum of the squared differences between actual and predicted values. SST represents the Sum of the Squared Total, representing the total variance in the data. It is the sum of the squared differences between each actual value and the mean of the actual values.
- Kullback–Leibler Divergence (KL Divergence): Measures the distribution similarity between the actual and generated data. The KL divergence between two distributions Q and P is formulated using the following notation: $KL(P||Q)$, where the $||$ operator indicates "divergence" or Ps divergence from Q.

6.3. Hyperparameters and Model Training

We conducted extensive experiments to determine the best hyperparameters for the model. To avoid excessive complexity, we trained the generator independently as a standalone model, applying the Grey Wolf Optimizer (GWO) to find near-optimal hyperparameters. These parameters include window size, optimizer type, learning rate, dropout rate, and the number of hidden units and layers. However, the Lévy parameters, σ and λ, were optimized through neural network gradients, with (0.02) as an initial value for both. The table below shows the hyperparameters and their selected values for the generator model. We used the same parameters in the ForGan model for the discriminator with the GRU cell to avoid further hyperparameter fine-tuning. Our strategy was not to use an early stopping mechanism. Instead, we fixed the number of epochs at 10,000 and saved the best-performing model at the training stage. We also tested generating the noise vector for the generator using different methods: Geometric Brownian Motion (GBM), uniform noise, Gaussian noise, and Poisson noise. All experiments were repeated three times and implemented in PyTorch v.2.4.

7. Results and Discussion

Table 1 presents the optimal configuration we found during the training or by the GWO. In particular, the Merton parameters (σ represents the volatility of the diffusion, and λ represents the jump frequency) were calibrated using a simple neural network during

the training process. The absolute values of σ and λ were used to generate the Merton path. Meanwhile, the other parameters were determined through optimization with the GWO. The numerical results achieved are summarized in Tables 2 and 3. Figures 6 and 7 plot the actual and forecasted prices by the proposed model. The first chart from the left (in Figures 6 and 7) presents a line plot comparing the predicted and actual values, while the second chart visualizes the same comparison using a scatter plot. The third chart highlights a distributional comparison between the actual and generated values. These visualizations demonstrate the predicted values' closeness to the actual data, showcasing the model's high accuracy. Tables 2 and 3 illustrate the proposed model's evaluation compared to several benchmark models (CGAN, CWGAN-GP, ForGAN, and TimeGAN) on two datasets: Brent and WTI. The tables present model performance metrics (MAE, MSE, RMSE, MSPE, MAPE, R^2, and KL) for Brent and WTI datasets, respectively. The proposed model outperforms the others across most metrics, indicating superior predictive performance and lower error rates. Table 4 presents a comparative analysis of the models' average runtime (in seconds). Among the models, CGAN has the shortest runtime, followed by the proposed model, while TimeGAN demonstrates a significantly higher runtime. It is worth mentioning that this represents the training runtime of the models using the optimal configuration assigned manually. However, it does not include the runtime spent for the GWO optimization process. The experiments were conducted on a Windows 11 environment with a Core i5 processor featuring eight cores and 16 GB of RAM; the implementation used the PyTorch framework.

Table 1. Generator hyperparameters and their assigned values.

Hyperparameter	Value
Optimiser	RMSProp
Loss Function	Binary Cross-Entropy Loss (BCE)
Activation Function	Relu
Learning Rate	0.0034
Batch Size	32/16
Window Size	10/15
Dropout	0.33
Torch Fixed Seed	4
Noise Dimension	32
Noise Type	Gaussian
LSTM Hidden Dimension	8
Lévy σ and λ (Brent)	0.0891 and 0.0302
Lévy σ and λ (WTI)	0.3181 and -0.1201

Table 2. Model evaluation metrics—Brent dataset.

Model	MAE	MSE	RMSE	MSPE	MAPE	R^2	KL
Our Model	0.004800	0.000054	0.007348	0.000455	0.013696	0.989210	0.000139
CGAN	0.00700	0.000103	0.010198	0.000991	0.021051	0.979220	0.000325
CWGAN-GP	0.008050	0.000118	0.010886	0.001990	0.031507	0.981623	0.000691
ForGAN	0.011500	0.000207	0.014387	0.002198	0.035774	0.958641	0.000460
TimeGAN	0.273225	0.114837	0.338876	21.167172	1.142248	-0.909833	0.296663

Table 3. Model evaluation metrics—WTI dataset.

Model	MAE	MSE	RMSE	MSPE	MAPE	R^2	KL
Our Model	0.005600	0.000062	0.007874	0.000293	0.012174	0.987919	0.000069
CGAN	0.007300	0.000109	0.010440	0.000597	0.016878	0.980206	0.000279
CWGAN-GP	0.009700	0.000176	0.013266	0.002869	0.037412	0.976150	0.001153
ForGAN	0.010600	0.000188	0.013711	0.001140	0.025341	0.963368	0.000305

Table 4. Run time analysis.

Model	Run Time Average
Our Model	196.152
CGAN	152.752
CWGAN-GP	1088.501
ForGAN	301.539
TimeGAN	4076.762

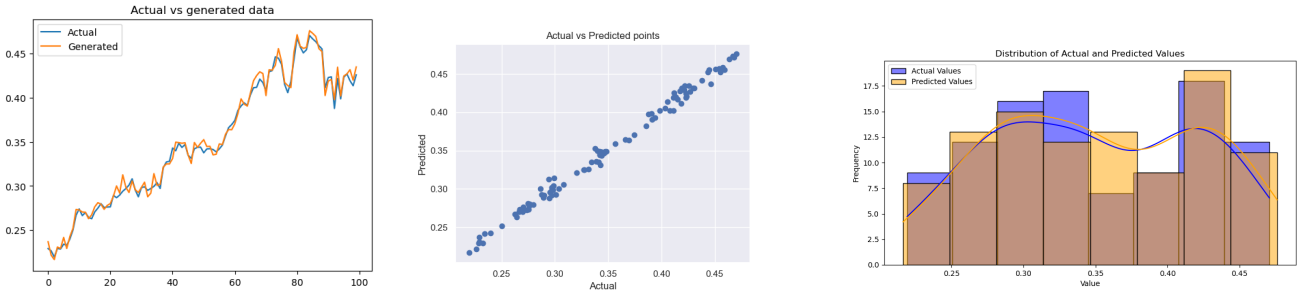


Figure 6. Evaluations of proposed model: actual vs. generated values for Brent dataset.

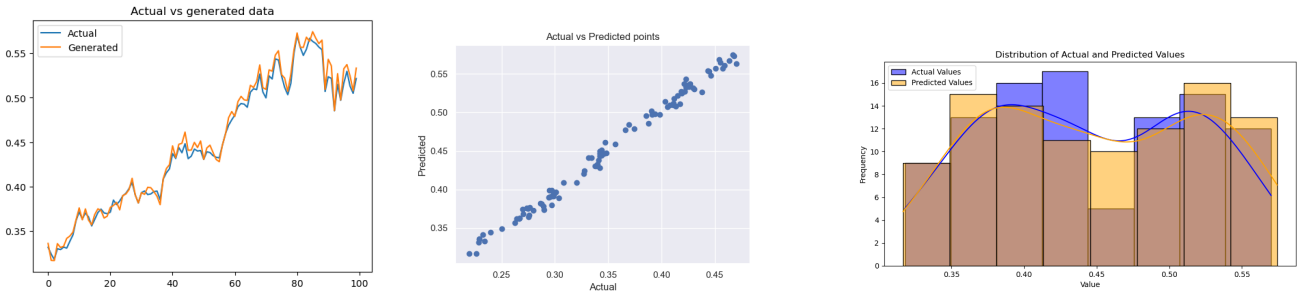


Figure 7. Evaluations of proposed model: actual vs. generated values for WTI dataset.

Figures 8 and 9 provide a comparative analysis of actual vs. forecasted values for the Brent and WTI datasets, respectively. Each figure displays the performance of the proposed model alongside baseline models (CGAN, CWGAN-GP, and ForGAN). The plots illustrate how closely the generated values from each model track the actual values over time. In both datasets, the proposed model demonstrates a high level of accuracy, with generated values closely following the actual data trends, showcasing its effectiveness in time series forecasting compared to the baseline models. Figure 10 comprehensively analyses the distribution and correlation of generated values vs. actual values for the Brent dataset. On the left, the scatter plot matrix shows the pairwise relationships between the actual and generated values across different models (proposed model, CGAN, CWGAN-GP, ForGAN). Each plot highlights the distribution alignment, indicating how closely the generated values follow the data distribution. On the right, the heatmap displays the correlation matrix, with the proposed model showing the highest correlation (0.997) with the actual values, demonstrating its effectiveness in accurately replicating the data distribution compared to other models. Additionally, incorporating the Lévy process enhanced forecasting accuracy, as indicated by a lower MSE than the base CGAN model. Our empirical findings indicate that dedicating time to tuning model parameters yields better results than focusing on constructing more complex model architectures. Therefore, we applied the Grey Wolf Optimizer (GWO) to optimize our model parameters effectively.



Figure 8. Comparison of actual values with predicted values—proposed model and baseline models—Brent dataset.



Figure 9. Comparison of actual values with forecasts from proposed model and baseline models—WTI dataset.

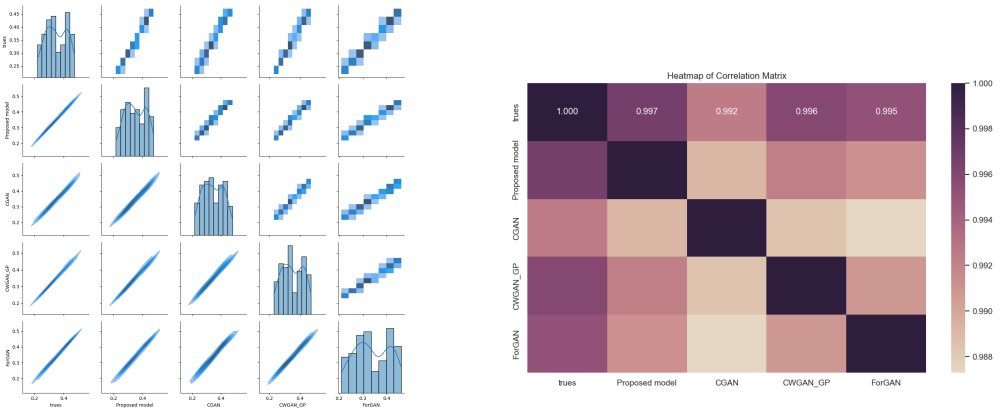


Figure 10. Brent dataset—distribution and correlation analysis of generated values.

8. Conclusions and Future Works

This work introduced a crude oil-driven conditional GAN model for oil price forecasting. Our primary contribution lies in enhancing oil price prediction performance by comprehensively addressing multiple facets of the forecasting task, leveraging diverse applications and cutting-edge techniques. The proposed model uses historical prices and sentimental scores as conditions. It integrates a Lévy–Merton jump-diffusion process with parameters learned through neural network gradients. We optimized the model parameters using the GWO. The model was evaluated on two real-world datasets (crude Brent and crude WTI), demonstrating accurate forecasts and outperforming established models across different test periods that exhibited varying movements. Additionally, incorporating the Lévy process improved forecasting accuracy, as evidenced by lower MSE than the base CGAN model. However, the lack of data related to sentiment market analysis limited our experiments to a specific period (from 2012 to 2021). We employed a simple neural network to calibrate the Merton model variables to reduce computational costs. More advanced differentiable SDE solvers, see, e.g., [22], could also be utilized for this purpose. Similarly to how GANs have been adapted for time series forecasting via the conditional GAN variant, LLMs could be explored for similar applications by reprogramming the input dataset and employing a robust prompt engineering methodology. Additionally, these models can provide a broad context through embedded knowledge and web crawling for relevant events, which can significantly enhance their generalization and predictive performance, particularly within the context of data scarcity.

Author Contributions: Conceptualization, M.A. and L.D.P.; Methodology, M.A. and L.D.P.; Software, M.A.; Investigation, M.A.; Writing—original draft, M.A.; Writing—review & editing, L.D.P.; Supervision, L.D.P.; Project administration, L.D.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original data presented in the study are openly available at https://github.com/Med-Rokaimi/GAN_synthetic_data_time_series (accessed on 10 October 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Foroutan, P.; Lahmiri, S. Deep learning systems for forecasting the prices of crude oil and precious metals. *Financ. Innov.* **2024**, *10*, 111. [CrossRef]
2. Ghoddusi, H.; Creamer, G.G.; Rafizadeh, N. Machine learning in energy economics and finance: A review. *Energy Econ.* **2019**, *81*, 709–727. [CrossRef]
3. Alruqimi, M.; Di Persio, L. Enhancing Multi-step Brent Oil Price Forecasting with Ensemble Multi-scenario Bi-GRU Networks. *Int. J. Comput. Intell. Syst.* **2024**, *17*, 225. [CrossRef]
4. Wen, Q.; Zhou, T.; Zhang, C.; Chen, W.; Ma, Z.; Yan, J.; Sun, L. Transformers in Time Series: A Survey. *arXiv* **2023**, arXiv:2202.07125.
5. Zeng, A.; Chen, M.H.; Zhang, L.; Xu, Q. Are Transformers Effective for Time Series Forecasting? In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022.
6. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*; Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K., Eds.; Curran Associates, Inc.: Newry, UK, 2014; Volume 27.
7. Yoon, J.; Jarrett, D.; van der Schaar, M. Time-series Generative Adversarial Networks. In *Advances in Neural Information Processing Systems*; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Newry, UK, 2019; Volume 32.
8. Xu, T.; Wenliang, L.K.; Munn, M.; Acciaio, B. COT-GAN: Generating Sequential Data via Causal Optimal Transport. *arXiv* **2020**, arXiv:2006.08571.
9. Asre, S.; Anwar, A. Synthetic Energy Data Generation Using Time Variant Generative Adversarial Network. *Electronics* **2022**, *11*, 355. [CrossRef]

10. Koochali, A.; Schichtel, P.; Dengel, A.; Ahmed, S. Probabilistic Forecasting of Sensory Data with Generative Adversarial Networks—ForGAN. *IEEE Access* **2019**, *7*, 63868–63880. [CrossRef]
11. Milena Vuletić, F.P.; Cucuringu, M. Fin-GAN: Forecasting and classifying financial time series via generative adversarial networks. *Quant. Financ.* **2024**, *24*, 175–199. [CrossRef]
12. Brophy, E.; Wang, Z.; She, Q.; Ward, T. Generative Adversarial Networks in Time Series: A Systematic Literature Review. *ACM Comput. Surv.* **2023**, *55*, 199. [CrossRef]
13. Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. *arXiv* **2014**, arXiv:1411.1784.
14. Jaiswal, A.; AbdAlmageed, W.; Wu, Y.; Natarajan, P. Bidirectional Conditional Generative Adversarial Networks. *arXiv* **2018**, arXiv:1711.07461.
15. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein GAN. *arXiv* **2017**, arXiv:1701.07875.
16. Busari, G.A.; Lim, D.H. Crude oil price prediction: A comparison between AdaBoost-LSTM and AdaBoost-GRU for improving forecasting performance. *Comput. Chem. Eng.* **2021**, *155*, 107513. [CrossRef]
17. Kim, G.I.; Jang, B. Petroleum Price Prediction with CNN-LSTM and CNN-GRU Using Skip-Connection. *Mathematics* **2023**, *11*, 547. [CrossRef]
18. Wu, H.; Xu, J.; Wang, J.; Long, M. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. *arXiv* **2021**, arXiv:2106.13008.
19. Huang, X.; Li, Q.; Tai, Y.; Chen, Z.; Liu, J.; Shi, J.; Liu, W. Time series forecasting for hourly photovoltaic power using conditional generative adversarial network and Bi-LSTM. *Energy* **2022**, *246*, 123403. [CrossRef]
20. Merton, R.C. Option pricing when underlying stock returns are discontinuous. *J. Financ. Econ.* **1976**, *3*, 125–144. [CrossRef]
21. Kaplan, H.; Weichselbraun, A.; Braşoveanu, A.M.P. Integrating Economic Theory, Domain Knowledge, and Social Knowledge into Hybrid Sentiment Models for Predicting Crude Oil Markets. *Cogn. Comput.* **2023**, *15*, 1355–1371. [CrossRef] [PubMed]
22. Kidger, P.; Foster, J.; Li, X.; Oberhauser, H.; Lyons, T. Neural SDEs as Infinite-Dimensional GANs. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.