Cybersecurity

Research Open Access

VAECGAN: a generating framework for long-term prediction in multivariate time series

Xiang Yin[1], Yanni Han[2*], Zhen Xu[2] and Jie Liu[3]

Abstract

Long-term prediction is still a difficult problem in data mining. People usually use various kinds of methods of Recurrent Neural Network to predict. However, with the increase of the prediction step, the accuracy of prediction decreases rapidly. In order to improve the accuracy of long-term prediction,we propose a framework Variational Auto-Encoder Conditional Generative Adversarial Network(VAECGAN). Our model is divided into three parts. The first part is the encoder net, which can encode the exogenous sequence into latent space vectors and fully save the information carried by the exogenous sequence. The second part is the generator net which is responsible for generating prediction data. In the third part, the discriminator net is used to classify and feedback, adjust data generation and improve prediction accuracy. Finally, extensive empirical studies tested with five real-world datasets (NASDAQ, SML, Energy, EEG,KDDCUP)demonstrate the effectiveness and robustness of our proposed approach.

Keywords: Long-term prediction, Multivariate time series, Attention mechanism, Generating framework

Introduction

As countries around the world strengthen the construction of modern information infrastructure and promote the development of big data and the Internet of things, more and more information are collected by us through sensor devices.The security of network data has gradually become an important problem. Network managers deploy a large number of security equipments in the network to prevent various attacks. In order to enhance the security of the network, more and more researchers are also involved in the network security situation analysis technology. Among them, the prediction technology of time series data can effectively evaluate and measure the potential threats in the network. Through this technology, the system can be used for analysis, prediction, decision-making and control, such as automatic allocation of resources in the network, network attack early warning (Qu et al. 2005), security situation prediction (Liu et al. 2021), anomaly detection (Li et al. 2019) and so on.

In recent years, ANNs have been widely used in time series prediction. Users do not need to specify the function form of independent variables and dependent variables when building Artificial Neural Networks(ANNs). It can use back propagation algorithm to estimate parameters. Theoretically, it can generate any complex continuous function. Among them, Recurrent Neural Network (RNN) and sequence-to-sequence models (Sutskever et al. 2014) have achieved great success in the field of sequence data, and also attracted the attention of researchers. RNN adopts a chain structure to simulate the dynamic behavior of time series and retains the long-term pattern of time series through gate-like structures. At present, more and more people use RNNs for time series prediction, including Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) and Gated Recurrent Unit (GRU) (Cho et al. 2014). Several studies have shown success with variants of these models (Zhu and Laptev 2017; Laptev et al. 2017; Maddix et al. 2018).

*Correspondence: hanyanni@iie.ac.cn
[2]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
Full list of author information is available at the end of the article

In software engineering projects, long-term forecasting is particularly important for system requirement management, storage maintenance and scheduling planning. Multi-step ahead prediction refers to the prediction of multiple time steps in the future for a variable based on the past and present data. Specifically, real-world applications often entail a mixture of short-term and long-term repeating patterns. The related research on the long-term prediction of time series mainly focuses on the trend prediction. A hybrid neural network is proposed to predict the trend of time series (Lin et al. 2017). In some practical applications, people try to predict the trend of stock price(Xu and Cohen 2018). However, these algorithms do not make full use of the information provided by exogenous (driving) sequences. Yao (Qin et al. 2017) and Liu (Liu et al. 2020) proposed a neural network architecture based on the encoder-decoder network to solve the problem. However, with the increasement of prediction step, the complexity of prediction is improved and the prediction accuracy decreases. Zhang et al. (2019) tried to use Generative Adversarial Network (GAN) architecture to solve the prediction problem. He proposed a GAN neural network model with Multi-Layer Perceptron (MLP) as a discriminator network and LSTM as a generator network for financial forecasting. However, these methods are based on the recursive application of single-step prediction model for multi-step prediction. If there are errors in prediction, such errors will continue to accumulate. In general, we are facing a challenge: the task of using observed time series in the past to predict the unknown time series in long-term prediction– the larger the prediction steps are, the harder the problem is.

In order to cope with the above challenge, we propose VAECGAN (Variational Auto-Encoder Conditional Generative Adversarial Network). We use the encoder of VAE to encode the data driving series into lantern space and provide it to the generator, so that the lantern space is no longer random noise, and can contain part of the data information in the driving series. In the generation stage, LSTM and attention are used to generate prediction data that has the same time trend as the past data. In the discrimination stage, convolution layers are mainly used to extract data features and discriminate between the generated data and the true data. The main contributions of this paper are as follows:

(1) A framework VAECGAN is introduced for the long-term prediction. In the model, the encoder of VAE encodes the driving series into lantern space, so that the lantern space contains part of the series data information in the driving series. The CGAN module can improve the ability of the VAE module to generate time series data.

(2) We propose a dynamic weights clipping method. The dynamic weights clipping makes the discriminator more stable. Experiments in section 5 also prove the effectiveness of the clipping.

The remainder of this paper is organized as follows: "Related work" section introduces the related background and the basic idea of our work. "Problem statement" section describes the problem statement in the paper. In "The VAECGAN model" section, we present the detail of our model, including the Encoder network, the Generator network and the Discriminator network. Experiments are given in "Experiment" section. We conclude our work and give a glimpse of the future work in "Conclusion and future work" section.

Related work

In recent years, multiple studies have straightforwardly inherited the GAN framework within the temporal setting. Mogren proposed a Recurrent Neural Network architecture, Continuous-RNN-GAN(C-RNN-GAN) (Mogren 2016), which uses confrontation training to simulate the whole joint probability of sequence and generate data sequence. This model is demonstrated by training classical music sequences in midi format. Recurrent Conditional GAN (RCGAN) (Esteban et al. 2017) model is a medical data generation framework. This model follows the architecture of the traditional GAN model, in which the generator and discriminator can be replaced by RNN. Therefore, the RCGAN model can generate real value sequence data limited by some input conditions. EEGGAN (Hartmann et al. 2018) is a framework for generating brain signals. They improve the Wasserstein-GAN model to stabilize training and investigate a range of architectural choices critical for time series generation (most notably up and down sampling). EEG-GAN opens up new possibilities for new applications, not only for data enhancement, but also for spatial or temporal oversampling (proposed by Corley and Huang in 2018) or recovery of damaged signals. Time-series Generative Adversarial Networks (Yoon et al. 2019) is also a data generation approach, which generating realistic time-series data that combines the flexibility of the unsupervised paradigm with the control afforded by supervised training. But all these works are to generate data with the same time trend, not to predict the future data.

In some studies, representation learning is commonly used to deal with the compact encodings in prediction tasks. Therefore, several works have explored the benefit of combining autoencoders with adversarial training. Larsen et al. (2015) is proposed for learning similarity measures. Makhzani et al. (2015) is proposed for improving generative capability. But all these works are applied to image generation, not data generation.

By contrast, we choose Conditional GAN (CGAN) (Mirza and Osindero 2014) as our basic framework, but compared with CGAN, we code the exogenous sequence of temporal data through the Variational Autoencoder(VAE) network instead of random noise as the input of generator. In the encoder stage, we input the exogenous sequence data, adjust the data weight through the attention mechanism, and encode it with the LSTM network. In the stage of decoder and generator, we use encode results as lantern space and target sequence data as label input to the network, and decode(generate) data through LSTM and attention function. In the discrimination stage, we get the characteristics of data through convolution layers and optimize the discriminator network in a dynamic way. The framework is shown in Fig. 1.

Problem statement

Based on the concept of adversarial training, GAN is a deep learning framework that generates data through game learning. It can learn any complex probability distribution in theory. Because GAN can produce high quality images, it has achieved great success in the field of image generation.The essence of GAN is to generate data consistent with the distribution of real data. Long term forecasting also generates a series future data with similar characteristics to the current data. This inspires us to apply GAN to time series distribution learning for generating future time distribution. But if the prediction data is generated directly from the random noise Z, the quality is not very good. We use the VAE model to encode the exogenous sequence of data from the original distribution to a normal distribution, so the latent space contains the exogenous sequence information of data. Meanwhile, the Decoder net is not only used for decoding, but also can be used as a Generator network.As we can see,$x^k = (x_1^k, x_2^k, ..., x_T^k) \in R^T$ represents the driving series of length T and the $x_t = (x_t^1, x_t^2, ..., x_t^n) \in R^n$ denotes a vector of n driving series at time t.Self-attention is used to process the driving series x_t so that the weights among the driving series can be captured at time t. Meanwhile, the input-attention mechanism is also adaptively used to select the time correlation of the driving series x^k.Then we can see from Fig. 1, LSTM function process the calculation of the self-attention and input-attention and get the result as latent space z.

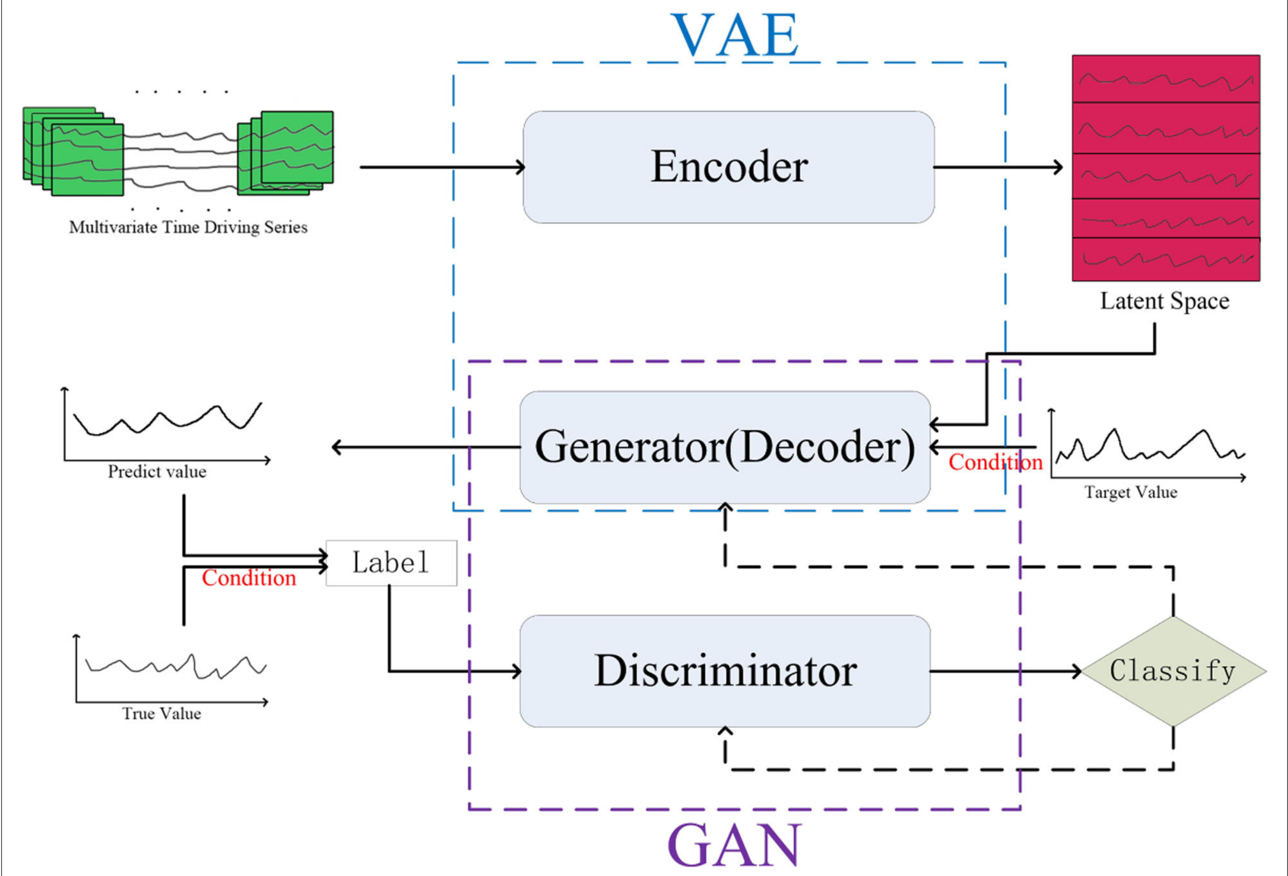

Fig. 1 The Architecture of The VAECGAN model

$$Z = E(x_1, x_2, ..., x_T) \tag{1}$$

Where$E(\cdot)$is the Encoder network. We put the target sequence and lantern space into the Generator net to generate the target sequence. Then, given T target value, i.e,$Y = (y_1, y_2, ..., y_T) \in R$, where T is the length of window size we define. Y denotes all target series during the past T time step. In the Generator (Decoder) stage, the temporal-attention mechanism is used to automatically select the time steps of the result of the encoder. Then the prediction value $\hat{Y} = (y_{\hat{T+1}}, y_{\hat{T+2}}, ..., y_{\hat{T+\epsilon}})$ will be calculated with the lantern space Z and target series Y. Given the previous reading, predict the target series $\hat{y}$.

$$\hat{y} = F(y_1, y_2, ..., y_T, Z) \tag{2}$$

Where $F(\cdot)$ is the nonlinear function.ϵ represents the prediction time steps. In order to get better prediction results, we use the real value $Y = (y_{T+1}, y_{T+2}, ..., y_{T+\epsilon})$ and prediction value $\hat{y}$ to train the Discriminator net, and add category labels $L = (L_{T+1}, L_{T+2}, ..., L_{T+\epsilon})$ as conditional variables to guide the Discriminator net. Specifically, the Discriminator is trained to minimize the average negative least square between its predictions per time-step and the labels of the sequence.

$$D_{loss} = LS(CNN(y), L) \tag{3}$$

Where $LS(\cdot)$ is the east square function. L is a vector of 1s,or 0s for sequence. The generator is trained to 'trick' the discriminator into classifying its outputs as the true data, that is, it hopes to minimize the least square between the discriminator's predictions on generated sequences and the 'true' label, the vector of 1s (we write as 1).

$$G_{loss} = D_{loss}(CNN(Z), 1) \tag{4}$$

The VAECGAN model

The VAECGAN model is composed of three networks, the encoder network, the generator network and the discriminator network. Figure 2 shows the architecture of the three parts. The encoder network processes the driving series and generates the lantern space which can keep the relationship information. The generator network use lantern space and target series to generate prediction series. The discriminator network classifies data into real and fake.

The Enocoder network

The encoder network is composed of input attention, self-attention and LSTM network. Figure 2a shows the encoder architecture. In time series prediction, long sequence input is not friendly to the Encoder-Decoder model, so we can better predict the target value by extracting important information of driving series through input-attention. Given T input series $x^k = (x_1^k, x_2^k, ..., x_T^k) \in R^T$, T is the time window. We can compute the attention weight by the following formula(5) and formula (6).

$$e_t^k = v_e^T tanh(W_e[h_{t-1}; s_{t-1}] + U_e x^k) \tag{5}$$

$$\alpha_t^k = \frac{exp(e_t^k)}{\sum_{i=1}^n exp(e_t^i)} \tag{6}$$

Where $v_e \in R^T$, $W_e \in R^{T\times 2m}$ and $U_e \in R^{T\times T}$ are parameters to learn.α_t^k is the attention weigh at time t. Then a SoftMax function ensures the attention weights sum to 1. In order to extract the series adaptively, we multiply the attention weight with the temporal series by the following formula(7).

$$\tilde{x}^1 = \left(\alpha_1^k x_1^k, \alpha_2^k x_2^k, ..., \alpha_T^k x_T^k\right) \tag{7}$$

Self-attention has been used to study textual representation and achieved great success (Vaswani et al. 2017; Yin et al. 2020). In this paper, self-attention dynamically adjusts the importance of the driving series, which makes unique adjustment coefficients for each driving series. Introduce an attention layer with an attention matrix capture the similarity of any token with respect to all neighboring tokens in an input sequence. Given the input driving series $x_t = (x_t^1, x_t^2, ..., x_t^k)$, the attention mechanism is implemented as follows:

$$g_t = tanh(W_g x_t + b_g) \tag{8}$$

$$\tilde{\alpha}_t^k = Sigmoid(W_\alpha g_t + b_a) \tag{9}$$

Where $W_g \in R^m$ and $W_\alpha \in R^{T\times m}$ are parameters to learn.b_g and b_α are the bias vectors. Then we multiply the attention weight coefficients with the attribute of driving series for showing the different importance of the different attributes of the driving series.

$$\tilde{x}^2 = \left(\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, ..., \alpha_t^k x_t^k\right)^T \tag{10}$$

Since $\tilde{x}^2$ will concatenate on $\tilde{x}^1$, we take the transpose of $\tilde{x}^2$ to make them have the same shape.

After calculating self-attention and input-attention, we feed the result as lantern space by using f_1 function which is an LSTM unit.

$$\tilde{h}_t^1 = f_1(\tilde{h}_{t-1}^1, \tilde{x}_t^1) \tag{11}$$

$$\tilde{h}_t^2 = f_1(\tilde{h}_{t-1}^2, \tilde{x}_t^2) \tag{12}$$

$$Z = [\tilde{h}_t^1; \tilde{h}_t^2] \tag{13}$$

Where $[\tilde{h}_t^1; \tilde{h}_t^2]$ is the concatenate of the two hidden states. And Z is the input for the generator net.

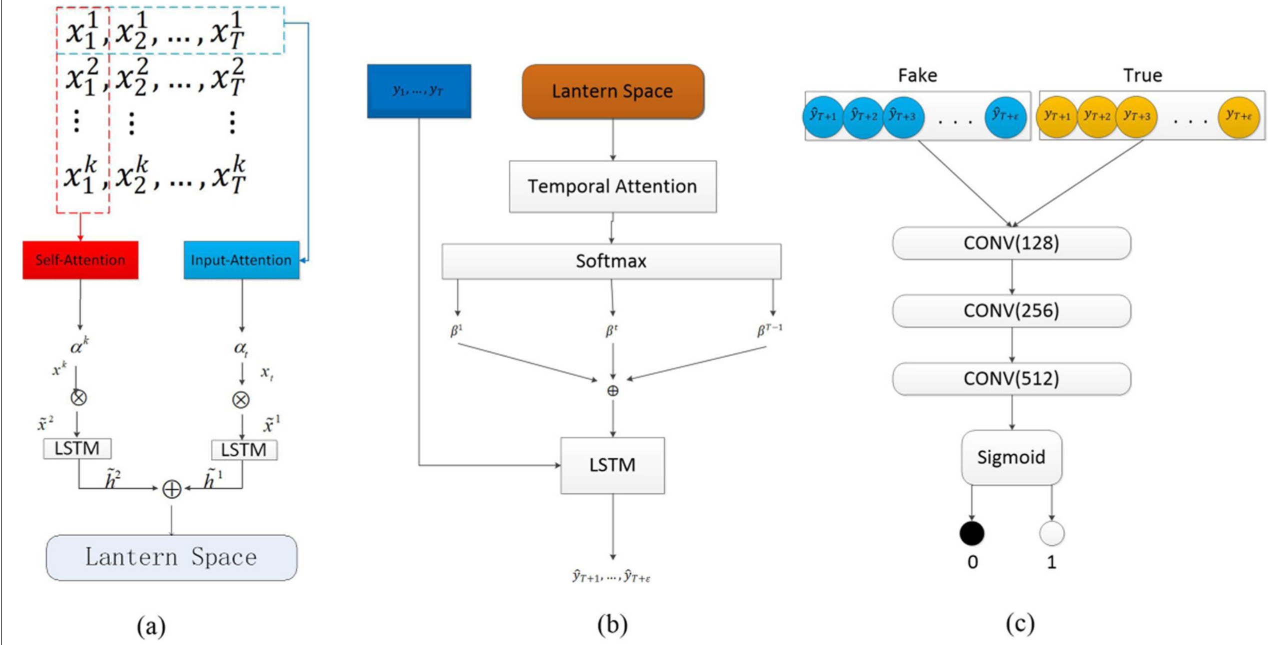

Fig. 2 The Architecture of The Encoder Network(**a**), Generator Network(**b**) and Discriminator Network(**c**)

The generator network

The generator network is composed of a sequential attention mechanism and LSTM network. Figure 2b shows the network framework. Then temporal attention is employed to adaptively select the hidden state of all time steps related encoder. The attention weight β_t^i of each time step t is calculated by the previous hidden state d_{t-1} and the cell state of the LSTM unit s'_{t-1} .

$$l_t^i = v_d^{T-1} tanh(W_d[d_{t-1}; s'_{t-1}] + U_d Z) \tag{14}$$

$$\beta_t^i = \frac{exp(l_t^i)}{\sum_{j=1}^{T-1} exp(l_t^j)} \tag{15}$$

Where $[d_{t-1}; s'_{t-1}] \in R^{2p}$ represents the concatenation of previous hidden state d_{t-1} and cell state of the LSTM unit s'_{t-1} .$v_d \in R^m$, $W_d \in R^{m\times m}$ and $U_d \in R^{m\times m}$ are parameters to learn. The context vector c_t is computed by the following formula(16).

$$c_t = \sum_{t=1}^{T-1} \beta_t^i Z \tag{16}$$

Then the context vectors c_{t-1} is combined with the given target series y_{t-1}.The calculation formula (17) is given below.

$$\tilde{y}_t = \tilde{W}_T[y_{t-1}; c_{t-1}] + \tilde{b} \tag{17}$$

Where $[y_{t-1}; c_{t-1}] \in R^{m+1}$ represents the concatenation of the target series y_{t-1} and the weighted sum context vectors c_{t-1} .$\tilde{W} \in R^{m+1}$ and $\tilde{b} \in R$ are the parameters. An LSTM unit can be used for updating the decoder hidden state d_t at time t. The calculation formula (18) is given below.

$$d_t = f_1(d_{t-1}, \tilde{y}_{t-1}) \tag{18}$$

The temporal dependence can be captured with the LSTM unit f_1.

The discriminator network

Figure 2c shows the Discriminator architecture. The discriminator network consists of convolutional neural network layers and a sigmoid activation function layer. The convolutional network is mainly composed of three 1-D convolution layers, which can better capture the interesting features and discriminate the real data from generated data.

In the GAN model, the discriminator network is mainly used to judge whether the input data is real data or generated data. It needs to adjust its parameters to give accurate judgment as much as possible. At the same time, the generator network is mainly used to generate data, which can simulate the real data as much as possible to confuse the discriminator network. The LSGAN model (Mao et al. 2017) proposed the least square method as the loss method, which can change the shortcomings of data quality is not high in the traditional GAN. Generally, taking cross entropy as the loss function makes the generator not optimize the data which judged as true by the discriminator network, even if the data does not fully conform to the trend of real data. Why does this phenomenon

Table 1 The MAE and RMSE indexes of time series prediciton over the five datasets(ϵ=50)

Models	SML2020	NASDAQ100	Energy	EEG	KDDCUP
LSTM	0.7924	0.4362	1.3416	0.3977	0.3556
	0.7145	0.3389	0.5968	0.4304	0.3928
Seq2Seq	0.7594	0.3491	1.2385	0.2814	0.3014
	0.6451	0.3369	0.6147	0.3816	0.3684
DARNN	0.5134	0.1554	1.1561	0.2951	0.2579
	0.3938	0.2881	0.5912	0.3817	0.3177
TCN	0.5614	0.0993	1.1154	0.2411	0.1935
	0.4989	0.2412	0.5983	0.3591	0.2881
DSTPRNN	0.3254	0.1217	1.0471	0.2589	0.1753
	0.2589	0.2621	0.5477	**0.3127**	**0.1829**
VAE	0.4388	0.1898	1.2378	0.2426	0.2253
	0.4136	0.2716	0.5471	0.3802	0.2968
VAECGAN	**0.3115**	**0.0579**	**0.8294**	**0.2193**	**0.1654**
	0.1638	**0.1911**	**0.4516**	0.3452	0.1901

happen? The main reason is that the generator network has completed its goal that confused the discriminator network as much as possible, so the cross entropy loss is very small at this time and cannot continue to optimize. The least square method is different. It is possible to further reduce the least square loss, so the generator network still generates data more like real data under the premise of confusing the discriminator network. Meanwhile, the least square method can also make the process of GAN training more stable. Therefore, we think that using the least square method as the loss function can effectively improve the quality and stability of the generated data. The expression of the least square loss function is as follows:

$$\min_D J(D) = \min_D \frac{1}{2} E_{z\sim p_r}[D(x)-a]^2 + 12E_{z\sim p_r}[D(G(Z))-b]^2 \tag{19}$$

$$\min_D J(G) = \min_D \frac{1}{2} E_{z\sim p_z}[D(G(z))-c]^2 \tag{20}$$

$D(\cdot)$ represents the discriminator network,$G(\cdot)$ represents the generator network. The input series data generate lantern space Z by Encoder network. The constants a (1) represent the real data label, and the constants b (0) represents the generated data label. The constants c (1) is the value for the discriminator to judge the generated data is the real data.

Table 2 The MAE and RMSE indexes of time series prediciton over the five datasets(ϵ=120)

Models	SML2020	NASDAQ100	Energy	EEG	KDDCUP
LSTM	0.8015	0.4751	1.3270	0.4091	0.4174
	0.7023	0.4635	0.8197	0.4933	0.5352
Seq2Seq	0.8012	0.3921	1.3653	0.3674	0.4228
	0.6974	0.4562	0.8214	0.5017	0.4811
DARNN	0.5327	0.3393	1.2671	0.3298	0.3973
	0.5526	0.4074	0.7489	0.4666	0.3596
TCN	0.5387	0.2565	1.1232	0.2713	0.2667
	0.5001	0.3672	**0.6152**	0.3938	0.3231
DSTPRNN	0.4543	0.2285	1.0910	0.2821	0.2588
	0.4109	**0.3613**	0.6457	0.3667	0.3182
VAE	0.5879	0.2915	1.2218	0.3017	0.2884
	0.5454	0.4237	0.7322	0.4511	0.3469
VAECGAN	**0.4499**	**0.2193**	**0.8411**	**0.2562**	**0.2418**
	0.3965	0.3782	0.6267	**0.3599**	**0.3118**

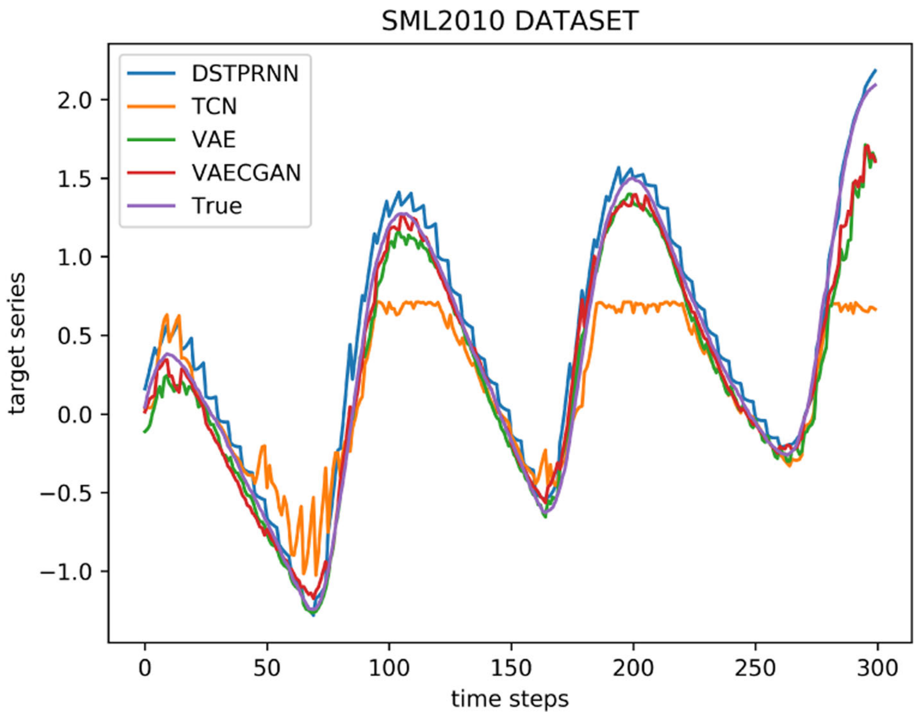

Fig. 3 The detail of time series prediction results over SML Datasets

In WGAN(Arjovsky et al. 2017),in order to satisfy Lipschitz condition, weight clipping is used to limit the weight of the whole network to a certain range(c=0.01).This method has been proved to be simple and has good performance. But this method produces some problems, that is, weight clipping also limits the performance of the network, and it is difficult to simulate complex functions. In addition, inappropriate setting of weight clipping range will also cause the gradient disappearance problem. Only when the setting of the weight clipping range is appropriate, can a suitable gradient value be returned. Therefore, this paper uses dynamic clipping strategy to solve this problem. Firstly, the weight value of the whole network is obtained. Then the weight values of the first θ precent

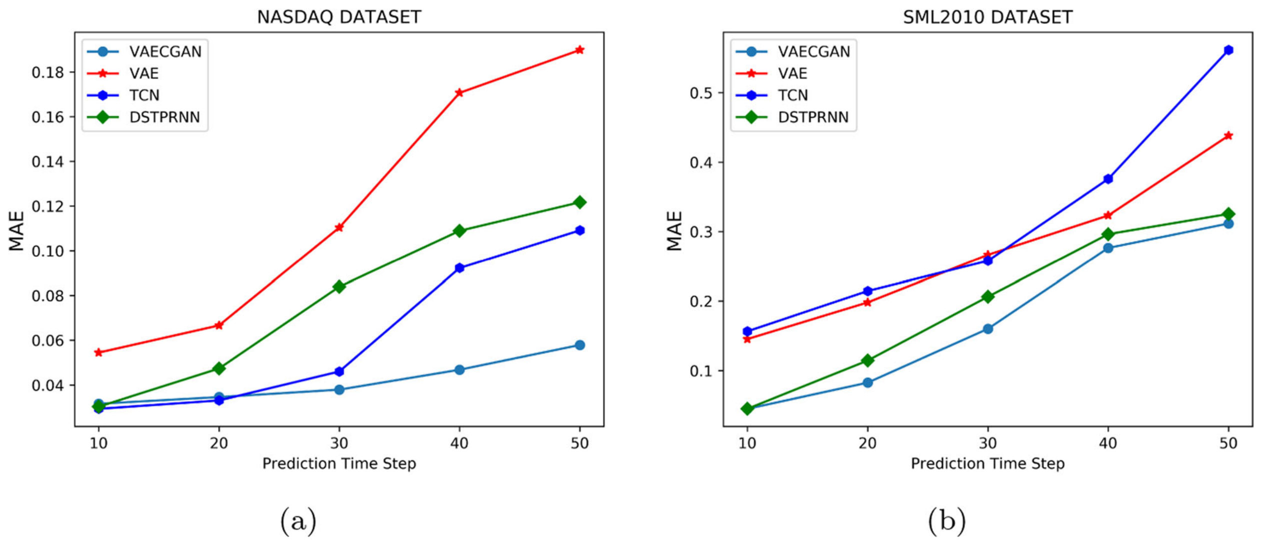

Fig. 4 MAE vs. Length of Prediction Timesteps(ϵ) over NASDAQ (a) and SML(b) Datasets

and the last θ precent are calculated, Last,the weight values are used as the dividing line for clipping. Algorithm 1 describes the training process of VAECGAN.

Algorithm 1: VAECGAN training process. The VAE model consists of the Encoder network and the Decoder(generator) network. The GAN model consists of the Generator network and the Discriminator network.

```
Require: $\theta$, the clipping threshold. TrueData, the real-world data.
$\omega$, the weight in discriminator net. Truelabel, a vector of 1s.
Fakedata, a vector of 0s.
For $i$ in ganepoch do:
  Latern$\leftarrow$VAE.Encoder
  Fakedata$\leftarrow$GAN.Genertator
  For $\tau$ in Dispoch do:
    DlossTrue$\leftarrow$GAN.Discriminator.Train(Truedata, Truelabel)
    DlossFake$\leftarrow$GAN.Discriminator.Train(Fakedata, Fakelabel)
    $\omega \leftarrow clip(\omega, \omega[\theta \times len(\omega)], \omega[(1-\theta) \times len(\omega)])$
  End for
  For $\tau$ in Genpoch do:
    Gloss$\leftarrow$ GAN.Genertator.Train(Truelabel)
  End for
  For $\tau$ in Vaepoch do:
    VAE.Train()
  End for
End for
```

Experiment

In order to evaluate the effectiveness of our model, we conduct experiments on five public datasets. The parameters setting of our proposed VAECGAN model and the evaluation metrics are introduced. Then, we adopt five different baseline models for comparison. Moreover, we show the comparison results between VAECGAN and other baselines and study the parameter sensitivity of the clipping threshold.

Data sets

SML2010 Data Set (SML)(SML2010 2014): The dataset is collected from a monitor system mounted in a domestic house. The data were sampled every minute, computing and uploading it smoothed with 15-min means. In our experience, the target value is the indoor temperature(room), and 18 other features are selected as the driving series.

NASDAQ 100 Data Set (NASDAQ)(NASDAQ100 2017): The subset of the entire nasdaq100 stock dataset includes 81 major corporations and interpolates the missing data with linear interpolation. The index value of nasdaq100 is used as the target series. These data include 105 days of inventory data from July 26 to December 22 in the 2016 year. Each day contains 390 data points except for 210 data points on November 25 and 180 data points on December 22 which is collected minute-by-minute. In our experience, the last column is the target series and the other 80 columns are driving series.

Appliances energy prediction Data Set(Energy) (Candanedo Ibarra et al. 2017): The dataset is at 10 min for about 4.5 months. In our experiment, we employ appliances energy use as the target series, delete the date attribute, and employ other attributes as driving series.

EEG Steady-State Visual Evoked Potential Signals Data Set(EEG)(EEG 2018): This dataset consists of 30 subjects performing Brain Computer Interface for Steady State Visual Evoked Potentials (BCI-SSVEP), and we only use the visual image search dataset from the first subject. In our experiment, we use O1 as the target value and the other 13 signal attributes coming from the electrodes as exogenous series.

KDDCUP: This is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining. This database contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment.

In our experiments, the last 20% points are the test data. Among the rest 80% data, the previous 80% data points are the training data and the later 20% points are the validation data. In order to make each feature make the same contribution to the results, the normalization method is used to preprocess the data.

Parameter settings and evaluation metrics

Hyper-parameters: In this experiment, we set seven parameters according to the previous work (Qin et al. 2017; Liu et al. 2020). The Adam optimizer (Kingma and Ba 2014), in which the learning rate is set as 0.0001 and the batch size is set as 128, is used to training the generator network and discriminator network. In the VAECGAN model, the length of the window size T is set as the value of 5,8,10,13,15,20. The prediction result proves that 'T equals 10' is the best choice. For simplicity, the hidden units(m) of the encoder network, the hidden units(p) of the generator network have the same size which conducts a search over 16, 32, 64, 128, 256. When m=p=64 or 128, our approach achieves the best performance over the test set. The clipping threshold θ will be proved in the next part.

Evaluation Metrics: In order to compare the effectiveness of various time series prediction algorithms, we use two common criteria to evaluate our model, namely root squared error (RMSE) (Plutowski et al. 1996) and mean absolute error (MAE) which are widely used in regression tasks. The formula of the two measurements is defined below:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_t^i - \hat{y}_t^i)^2} \tag{21}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y_t^i - \hat{y}_t^i\right| \tag{22}$$

Where y_t is the true target at time t and $\hat{y}_t$ is the predicted value at time t.

Baseline models

LSTM (Hochreiter and Schmidhuber 1997): LSTM is a variant algorithm of RNN which overcomes the limitation of vanishing gradient in RNNs. Since it can capture long-term dependence, it has achieved good results in many time series tasks.

Seq2Seq (Sutskever et al. 2014): Seq2Seq is an Encoder-Decoder model with a sequence of inputs and sequence of outputs. Encoder neuro network can turn a variable length input sequence into a fixed length vector. Then Decoder neuro network can decode the vector into a variable length output sequence. This method has good performance in machine translation, text translation or other NLP processing.

DARNN (Qin et al. 2017): This algorithm is proposed in 2017, which shows the state-of-the-art performance in single-step time series prediction. With the dual-stage attention scheme, DARNN model can not only make predictions effectively, but can also be easily interpreted.

TCN (Bai et al. 2018): The Temporal Collaborative Network integrates the modeling ability in the time domain and the feature extraction ability in the low parameter number of convolutions. It runs faster than RNN and is good at capturing timing dependency.

DSTPRNN (Liu et al. 2020): The Dual-Stage Two-Phase based RNN is inspired by the human attention mechanism. The first phase produces violent but decentralized response weight, while the second phase leads to stationary and concentrated response weight. multiple attentions are employed on target series to boost the long-term dependence.

VAE: This method uses the combination of encoder net and generator net mentioned in this paper. This method is also used to train the encoder network.

Performance comparison

In this section, our proposed model is compared with the other five baseline models on five datasets. The prediction result with fifty time steps records in Table 1. The LSTM units in the LSTM model and Seq2Seq model are 64. Other models are consistent with their papers. **The first line represents the MAE, the second line represents the RMSE, and the best result displays in boldface.**

In Tables 1 and 2, the prediction accuracy will be reduced with a long step size. The prediction results of the LSTM model and Seq2Seq model can capture the temporal dependence to a certain extent. But in the Seq2Seq model, the series data are mapped to a fixed dimension vector in the encoder stage. Therefore, some information in the series data is lost. The DCRNN model is proposed for short term prediction. It's also not good enough in the long term prediction. In contrast, although the VAE model also belongs to the encoder-decoder network, the input attention mechanism and the self-attention mechanism can retain data information to a large extent in the encoder stage. Therefore, the prediction result of the VAE model is more accurate than the Seq2Seq model. In Comparison, although the TCN model captures the temporal dependence with convolution layers, its performance is not good in the face of long-term prediction. Because the transfer learning ability of the TCN model is poor, the prediction effect of different databases is not good. DSTPRNN model adopts a two-stage attention mechanism, which can effectively capture temporal and spatial dependence. Therefore, it shows good prediction performance. The performance of the VAE model is worse than the DSTPRNN model. However, after adding the CGAN module, the prediction effect of the VAECGAN model has been greatly improved. It also proves that the training and feedback by the discriminator network can generate better prediction data.

It can be clearly seen from Fig. 3 that the red curve (VAECGAN) is more consistent with the purple curve (real values). This shows that our model is more accurate than the other three models. At the same time, it can be seen that VAECGAN maintained the same trend with the real value compared with TCN, which fluctuated a lot. Due to the poor performance of LSTM and seq2seq algorithms, we omit them.

In Fig. 4, we show the prediction effect of the various model on the SML dataset and the NASDAQ dataset. It can be seen from the observation that with the increase of the prediction step ϵ, the prediction accuracy of all the models has been reduced to varying degrees. This phenomenon also confirms the difficulty that the prediction accuracy decreases with the increase of the prediction step. But VAECGAN model is more stable than the other baseline model. And with the increase of the step size, the VAECGAN model performs better. Mean-

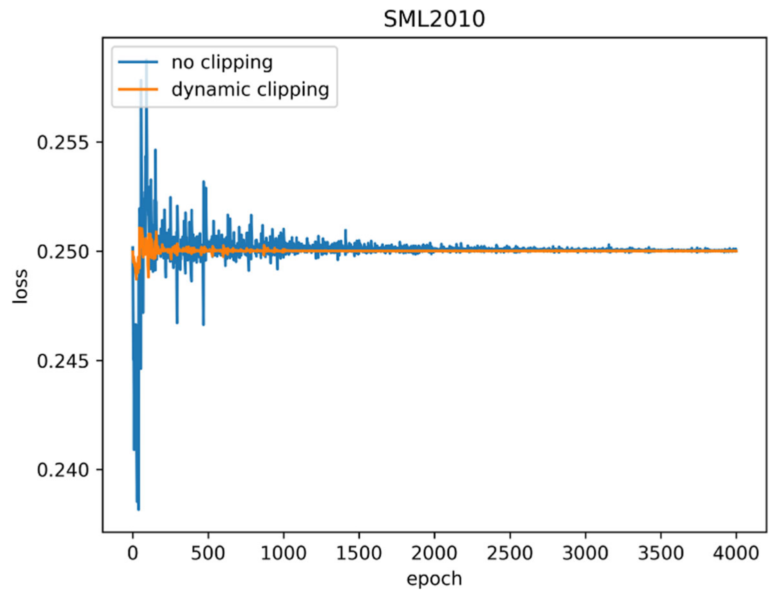

Fig. 5 The loss of the discriminator network

while, the performance of the VAECGAN model is no worse than the TCN model and DSTPRNN model in short-term perdition. The predicted value of the VAECGAN model is closer to the real value than the other model at the corner. Therefore, it can be proved that the performance of the VAECGAN model in prediction is better.

In order to verify the help of the dynamic weight clipping strategy for the stability of the discriminator, we compare the loss value of the discriminator with and without weight clipping on the SML dataset, as shown in Fig. 5. It is obvious that the yellow line is more stable and converges faster than the blue line, which indicates that the discriminator network in the model with weight clipping strategy has better stability and faster convergence effect.

In order to evaluate the sensitivity and effectiveness of the dynamic weight clipping strategy, we test the effect of different thresholds (from 0.1 to 0.3) on the prediction results. Figure 6 shows the effect of weight clipping methods. As shown in the figure, the prediction results will fluctuate with the change of clipping threshold θ. Therefore, defining an appropriate threshold can effectively

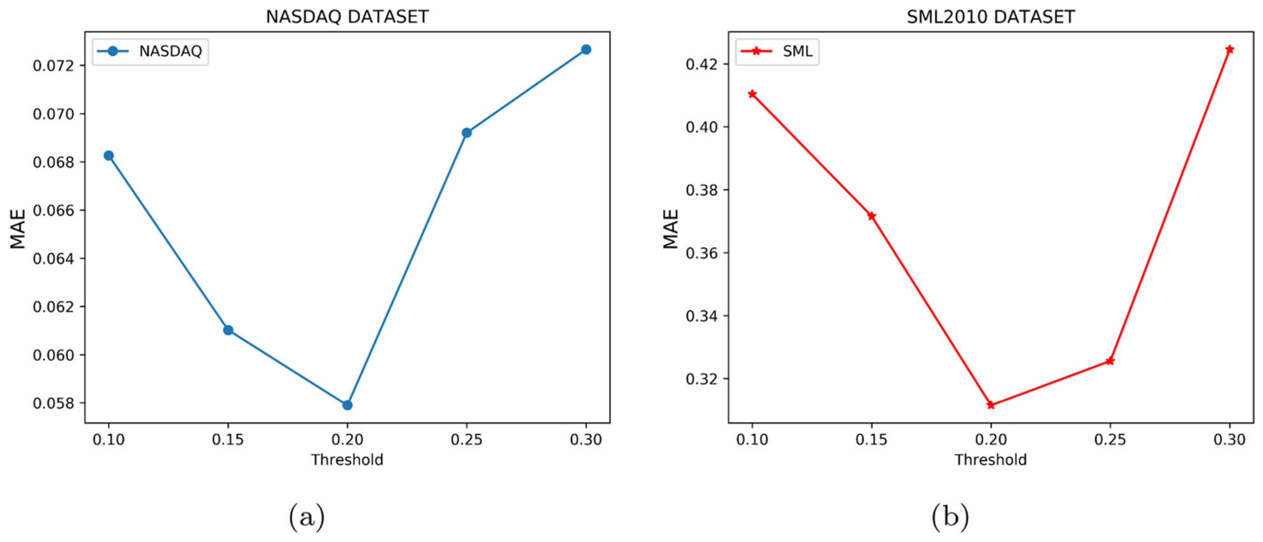

Fig. 6 MAE vs. clipping threshold over NASDAQ (a) and SML(b) Datasets

improve the prediction effect. The prediction result of the model outperforms better when the clipping threshold θ is equal to 0.2.

Conclusion and future work

In this study, a new framework VAECGAN for long-term prediction in multivariate time series has been proposed. The encoder module is used to deal with the multidimensional driving sequence data, and the results of the encoder network input into the generator as the latent space. Compared with generating data from noise, more relevant information is retained in the latent space. Meanwhile, in order to improve the accuracy of prediction, the discriminator network is used to feedback the result to the generator network. We also verify the help of dynamic threshold for data generation and the most suitable clipping threshold. Finally, we conduct the evaluation with five open real-world data sets. It is proved that the model achieved the best performance in long-term prediction on the evaluation metrics of MAE and RMSE by comparing with the five baselines.

In future work, we would continue to study the use of the GAN framework to generate long-term data to solve the problem that the algorithm in this paper sometimes generates duplicate data. We will also adjust data generation methods to improve the accuracy of short-term data prediction.

Acknowledgements
We would like to thank the anonymous reviewers for detailed comments and useful feedback.

Authors' contributions
The first author constructed the scheme and wrote the manuscript. The second author reviewed the manuscript and checked the validity of the scheme. She also proofread the manuscript and corrected the grammar mistakes. The third author and the fourth author joined the discussion of the work . All authors read and approved the final manuscript.

Funding
This work is supported by the Youth Talent Star of Institute of Information Engineering, Chinese Academy of Sciences (Y7Z0091105) and This work was supported in part by National Natural Science Foundation of China under Grant 61771469.

Availability of data and materials
Not applicable.

Declarations

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Institute of Information Engineering, Chinese Academy of Sciences, School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China. [2]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. [3]Network information department, China Mobile Communications Group Co.,Ltd, Beijing, China.

Received: 26 January 2021 Accepted: 15 April 2021
Published online: 01 July 2021

References
Arjovsky M, Chintala S, Bottou L (2017) Wasserstein gan. ArXiv abs/1701.07875
Bai S, Kolter JZ, Koltun V (2018) An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. CoRR abs/1803.01271. 1803.01271
Candanedo Ibarra L, Feldheim V, Deramaix D (2017) Data driven prediction models of energy use of appliances in a low-energy house. Energy Build 140. https://doi.org/10.1016/j.enbuild.2017.01.083
Cho K, van Merriënboer B, Gulcehre C, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation. https://doi.org/10.3115/v1/D14-1179
EEG (2018). https://archive.ics.uci.edu/ml/datasets/EEG+Steady-State+Visual+Evoked+Potential+Signals. Accessed 2018
Esteban C, Hyland SL, Rätsch G (2017) Real-valued (medical) time series generation with recurrent conditional gans
Hartmann K, Schirrmeister R, Ball T, Eeg-gan: Generative adversarial networks for electroencephalograhic (eeg) brain signals (2018)
Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–80. https://doi.org/10.1162/neco.1997.9.8.1735
Kingma D, Ba J (2014) Adam: A method for stochastic optimization. Int Conf Learn Represent 24:1–5
Laptev N, Yosinski J, Li EL, Smyl S (2017) Time-series extreme event forecasting with neural networks at uber. In: International Conference on Machine Learning. pp 1–5
Larsen ABL, Sønderby SK, Winther O (2015) Autoencoding beyond pixels using a learned similarity metric. CoRR abs/1512.09300. 1512.09300
Li D, Chen D, Jin B, Shi L, Goh J, Ng S-K (2019) Madgan: Multivariate anomaly detection for time series data with generative adversarial networks:703–716. https://doi.org/10.1007/978-3-030-30490-456
Lin T, Guo T, Aberer K (2017) Hybrid neural networks for learning the trend in time series. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17. pp 2273–2279. https://doi.org/10.24963/ijcai.2017/316
Liu D, Cheng J, Yuan Z, Wang C, Huang X, Yin H, Lin N, Niu H (2021) Prediction methods for energy internet security situation based on hybrid neural network. IOP Conf Ser Earth Environ Sci 645:012085. https://doi.org/10.1088/1755-1315/645/1/012085
Liu Y, Gong C, Yang L, Chen Y (2020) Dstp-rnn: A dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction. Expert Syst Appl 143:113082. https://doi.org/10.1016/j.eswa.2019.113082
Maddix D, Wang Y, Smola A (2018) Deep Factors with Gaussian Processes for Forecasting. CoRR abs/1812.00098. https://arxiv.org/abs/1812.00098
Makhzani A, Shlens J, Jaitly N, Goodfellow IJ (2015) Adversarial autoencoders. CoRR abs/1511.05644. 1511.05644
Mao X, Li Q, Xie H, Lau RK, Wang Z, Smolley S (2017) Least squares generative adversarial networks. In: 2017 IEEE International Conference on Computer Vision (ICCV). IEEE Computer Society, Los Alamitos. pp 2813–2821. https://doi.ieeecomputersociety.org/10.1109/ICCV.2017.304
Mirza M, Osindero S (2014) Conditional generative adversarial nets. CoRR abs/1411.1784. https://arxiv.org/abs/1411.1784
Mogren O (2016) C-rnn-gan: Continuous recurrent neural networks with adversarial training. CoRR abs/1611.09904. https://arxiv.org/abs/1611.09904
NASDAQ100 (2017). https://cseweb.ucsd.edu/~yaq007/NASDAQ100stockdata.html. Accessed 2017
Plutowski M, Cottrell GW, White H (1996) Experience with selecting exemplars from clean data. Neural Netw 9(2):273–294. https://doi.org/10.1016/0893-6080(95)00099-2
Qin Y, Song D, Chen H, Cheng W, Jiang G, Cottrell GW (2017) A dual-stage attention-based recurrent neural network for time series prediction. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17. pp 2627–2633. https://doi.org/10.24963/ijcai.2017/366
Qu G, Hariri S, Yousif M (2005) Multivariate statistical analysis for network attacks detection. p 9. https://doi.org/10.1109/AICCSA.2005.1387011
SML2010 (2014). http://archive.ics.uci.edu/ml/datasets/SML2010. Accessed 2014
Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Proceedings of the 27th International Conference on Neural Information Processing System - Volume 2. MIT, Cambridge. pp 3104–3112
Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser u., Polosukhin I (2017) Attention is all you need. In: Proceedings of the 31st

International Conference on Neural Information Processing Systems. NIPS'17. Curran Associates Inc., Red Hook. pp 6000–6010

Xu Y, Cohen SB (2018) Stock movement prediction from tweets and historical prices. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Melbourne. pp 1970–1979. https://www.aclweb.org/anthology/P18-1183

Yin X, Han Y, Sun H, Xu Z, Yu H, Duan X (2020) A multivariate time series prediction schema based on multi-attention in recurrent neural network. In: IEEE Symposium on Computers and Communications, ISCC 2020, Rennes, France, July 7-10, 2020. pp 1–7. https://doi.org/10.1109/ISCC50000.2020.9219721

Yoon J, Jarrett D, van der Schaar M (2019) Time-series generative adversarial networks. In: Wallach HM, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E, Garnett R (eds). Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. pp 5509–5519. https://proceedings.neurips.cc/paper/2019/hash/c9efe5f26cd17ba6216bbe2a7d26d490-Abstract.html

Zhang K, Zhong G, Dong J, Wang S, Wang Y (2019) Stock market prediction based on generative adversarial network. Procedia Comput Sci 147:400–406. https://doi.org/10.1016/j.procs.2019.01.256

Zhu L, Laptev N (2017) Deep and confident prediction for time series at uber. pp 103–110. https://doi.org/10.1109/ICDMW.2017.19

Publisher's Note