CHAPTER 12

Analysis of false data detection rate in generative adversarial networks using recurrent neural network

A. Sampath Kumar[a], Leta Tesfaye Jule[b,c], Krishnaraj Ramaswamy[c,d], S. Sountharrajan[e], N. Yuuvaraj[f], and Amir H. Gandomi[g]

[a]Department of Computer Science and Engineering, Dambi Dollo University, Dambi Dollo, Ethiopia
[b]Department of Physics, College of Natural and Computational Science, Dambi Dollo University, Dambi Dollo, Ethiopia
[c]Centre for Excellence in Indigenous Knowledge, Innovative Technology Transfer and Entrepreneurship, Dambi Dollo University, Dambi Dollo, Ethiopia
[d]Department of Mechanical Engineering, Dambi Dollo University, Dambi Dollo, Ethiopia
[e]School of Computing Science and Engineering, VIT Bhopal University, Bhopal, India
[f]Research and Development, ICT Academy, Chennai, India
[g]University of Technology Sydney, Ultimo, NSW, Australia

12.1 Introduction

Recently, the generative adversarial networks (GANs) have emerged as a potential class of generative models, where it operates as a joint optimization model with two neural networks of contrasting goals. Since decades, the generative adversarial network (GAN) [1–10] is emerging as a viable solution for most application with its adversarial training ability in optimizing the generative ability. GANs are an emergent model for unsupervised and semisupervised learning. The learning is achieved with implicit modeling of high-dimensional data distribution [6]. GANs are considered interesting since it moves away from a viewpoint of likelihood maximization; however, it uses an adversarial game approach for the process of training the generative models [11].

Conventional GAN has no prior training data on the distribution of data, which provide the goal of generating the samples from the distributions [12]. Effective training on GANs is rather challenging. The generator and discriminator model capacities are balanced for the generator to learn effectively. The lack of unambiguous convergence criterion tends to complement this problem [13]. Various attempts in existing researches to scale up the GAN are unsuccessful due to its reliability of identifying fake or false data and unstable training. GAN encounters various difficulties, while scaling up its robustness and scalability. Hence, a stable training model across a limited range of datasets with deeper learning algorithm can be made significant to scale up the operation of GAN in finding the false detection rate.

Generative Adversarial Networks for Image-to-Image Translation
https://doi.org/10.1016/B978-0-12-823519-5.00012-9
Copyright © 2021 Elsevier Inc.
All rights reserved.

The problem of imbalanced learning can be defined as a problem of learning from a binary or multiclass dataset, where for one of the classes called the majority class the number of instances is significantly greater than in the remaining classes called the minority classes [14]. In unbalanced datasets, standard learning methods work poorly because they are a prejudice to the majority classes. In particular, minority classes contribute lesser to minimize the objective function during training in a standard classification method [15].

Designing the GANs is still difficult in practice, even if GANs have achieved great success in image generation. In case a GAN would be unstable, network architectures should be well designed. Various GAN methods are developed [16–18] for improving the stabilization ability of GANs learning. The instability associated with GAN learning is caused by the saturation occurring while sampling the data in the discriminator [19]. Hence to resolve such problem, this chapter uses optimal weight selection to avoid the saturation, thereby increasing the stability of operation.

In this chapter, we develop a GAN and the operation in the GAN is scaled up using a recurrent neural network (RNN). It uses its neighborhood relationship between the samples to generate the target output and error generated. The errors are then propagated in the backward direction over the GAN to update the network weights to estimate the output. The RNN generator, on the other hand, reduces the probability of the RNN discriminator in identifying the false generated samples and increasing the probability of discriminator in identifying correctly the real samples. The objective function in RNN is designed in such a way that its gradient operator for the false samples is quite far from the decision boundary of RNN discriminator, thereby producing the increasing true classification rate.

12.1.1 Contributions

The main contributions of the study are presented below:

- The author(s) uses RNN to scale up the operation of GAN for stable training.
- The discriminator uses the same RNN to classify the generated and real data samples by updating the weights.
- The aim of RNN in the GAN structure is to delimit the error rate using its time series prediction based on its past inputs. It further uses its neighborhood relationship between the samples to generate the target output and error generated.
- The errors are then propagated in the backward direction over the GAN to update the network weights to estimate the output. The RNN generator, on the other hand, reduces the probability of the RNN discriminator in identifying the false generated samples and increasing the probability of discriminator in identifying correctly the real samples.
- The objective function in RNN is designed in such a way that its gradient operator for the false samples is quite far from the decision boundary of RNN discriminator, thereby producing the increasing true classification rate.

- The experiments are carried out on the real-world time series dataset show the results of accurate classification with increased false detection rate than benchmark GAN method.

The outline of the chapter is as follows: Section 12.2 discusses the related works. Section 12.3 provides the proposed GAN classification with modifications made in the work to improve the performance of classifier. Section 12.4 evaluates the entire work and Section 12.5 concludes the work with possible directions of the future scope.

12.2 Related works

Lyu et al. [20] designed denoising-based GAN for removing the mixed noise by the combination of three different GAN elements. It includes feature extractor networks, discriminator, and a generator. The feature extractor network additionally trains the generator-discriminator network using a mutual game. The direct mapping is implemented to eliminate the noise to improve the quality of input data. Chen et al. [21] developed a denoising method with GAN (D-GAN) to remove the presence of speckle noise. The generator is trained with ground truth value for mapping the noisy regions in an image. The discriminator finds the similarity comparison using the loss function with reconstructed input data. The generator-discriminator finally eliminates the noise with its stable training to achieve denoising effectiveness. Nawi et al. [22] formed a GAN embedded with a discriminative metric for the generation of real samples using its deep metric learning. The feature extractor acts as a discriminator that identifies both the discriminative and preserving loss. It further reduces the distance of estimation between the real and generation samples that preserve the input data from losses and improves the model stability with weight adaptation strategy. Li et al. [23] used multitask learning-based GAN (MTL-GAN) to segment the grains or noises present in the images of alloy microstructure. It uses the detection of grains at edges and its segmentation using rich convolutional features. The fine-tuning of GAM is employed to find the hidden grains and extracts the quantitative indicators. The above methods achieve the accurate noise reduction but the accuracy tends to reduce on increasing data samples.

Zhong et al. [24] developed a deep-GAN model embedded with the output noises input from the decoder-encoder model. The GAN is improved with adversarial training and Bayesian inference. The entire model is pretrained for mapping the noise vector to optimal feature that acts as an input feed for the generator. The generator's learnability is trained with dataset carrying intrinsic distribution information to reduce the set of errors. Cui et al. [25] use the encoder-decoder network in GAN to remove the noise from the input signal. The network tends to classify the original input data from the adversarial loss and pixel losses and thereby the original information is preserved. Sun et al. [26] modified the GAN model and formulated it with t-distributions noise mixture using a latent generative space. The learning components

of the t-distributions are mixture the diversity of class deification even in the presence of the noise vector. The classification loss embedded with generator-discriminator losses stabilizes the entire network. The use of adversarial loss tends to reduce with the stability of noise vector with increased data samples.

Ak et al. [27] developed an enhanced attention GAN with improved stability using an integrated attention module for linear modulation. It is designed to avoid matching losses between the features that significantly degrade the classification errors. The training stability is hence improved to resolve the collapse of the system with the reduction of instance noise. Xu et al. [28] developed a multideep GAN to resolve the unstable training using its reconstructive sampling. The multigranularity GAN is then decomposed to remove the noises present in the input data to further enhance the training. Deep neural network-based melanoma detection has been introduced by Banuselvasaraswathy et al. [29]. The deep learning-based training analysis suggested could be used in proposing an improved deep-based GAN development. Zhang et al. [30] developed a deep-GAN using its explicit training data that map the noise distribution with real-time data. Further, the generation of fake samples helps in balancing the datasets for stable training. These methods tend to reduce the probability of sampling under increased data samples using the first part of the generator.

Zhang and Sheng [31] used Wasserstein GAN to improve the model's generalization ability. It detects the input seismic signal and can distinguish the noise and the original signal especially in low signal-to-noise ratio (SNR) conditions. Hence, the stability is improved on adversarial sample datasets. Yang et al. [32] developed a GAN architecture with loss function to distinguish the clean signal from a parallel noisy signal. It uses supervised loss representation, i.e., high-level loss in the hidden layer of GAN to find the losses under low SNR and in a resource-constrained environment. These two methods suffer poorly from the unstable training and the inclusion of RNN can effectively improve the stabilized training pattern. In Refs. [26, 33–36], various machine learning and deep learning algorithms for classification of diseases are proposed. The optimization techniques suggested by them for diagnosing breast cancer are an effective one. The other big data analytics techniques suggested by them are noteworthy in handling big data and this in turn reveals the possibility of reducing the traffic in the network.

Sampathkumar et al. [37–40] proposed various optimization approaches for identifying the accuracy of the accurate gene for a particular disease. Various classifiers and statistical approaches are used to predict the feature selection gene with reduced dimensionality, which prove that algorithms are effective one. These methods fail to discriminate between original and fake samples resulting in poor determination accuracy.

Congestion control in WSN could be improved by multiple routing paths and this could be achieved by means of priority-based scheduling algorithms as discussed in Ref. [41]. Various researches have been undergone in the machine learning algorithm over the accuracy prediction. In this chapter, GAN with RNN [42–45] has concentrating over the reducing

false samples, which have not been carried out in the previous works. The utilization of RNN is to effectively process the temporal data, which are unavailable in the existing methods. The utilization of recurrent layers overcomes the challenges associated with a loss function. The RNN further can enable unsupervised learning against the conventional supervised learning models [46–48].

12.3 Methods

The proposed method is designed with the integration of GAN with RNN [22,25,32]. The GAN is responsible for optimal of neural network with adversarial training. The generator network is regarded as the first network that helps in mapping the input from the source and acts as a low-dimensional in contrast with high-dimensional dataset with respect to the target domain. The adversarial or discriminator network is the second network that generates the output, which the adversarial network fails in classifying the real-time dataset. The generator-discriminator gets optimized w.r.t. the discriminator output. For easier classification by the discriminator, the outputs from the generator w.r.t real dataset samples. To optimally attain the results, the study uses RNN algorithm to generate the optimal weights to achieve accurate predictive results.

The training of regression model [49] benefits the adversarial network signal. The adversarial network process the predicted and real classification with receptive field with weight function that quantifies well the task with global real-time prediction. It reduces the errors associated with classification of task. Hence, it is very necessary to design a loss function and that should contribute well with adversarial training.

Fig. 12.1 shows various components that include a generator G operating on a noise vector z, which is sampled using an input distribution p_z. It uses recurrent layers to transform a sample x from the input noise vector z. Finally, the discriminator D classifies the input samples from the real data distribution samples p_{data}.

The algorithm or the workflow of entire system is given in following steps:

Step 1: Generate the synthetic sampling from real data distributions using the generator network.

Step 2: Transform noise vector from distribution into newer samples.

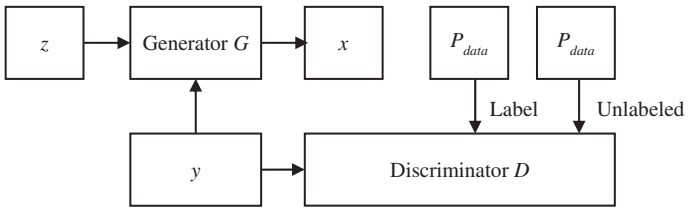


Fig. 12.1 Generative adversarial network.

Step 3: Run RNN and the weights of RNN is updated using the fitness function.
Step 4: Select the fitness value with improved probability estimator to increase classifier with probable weights updated based on the newer samples.
Step 5: Access the new samples and the samples from the generator using discriminator network.
Step 6: Find the similarity between these two data.

12.4 GAN-RNN architecture

GAN is designed with the two neural networks. First, the generator network generates the convincing synthetic sampling x drawn from real data distributions p_{data}. The generator network transforms the noise vector z from p_z distribution into newer samples x. Second, the discriminator network accesses the samples from the real data distributions p_{data} and it then accesses the samples from G and classifies between the two data. The training of GAN is carried out to solve the optimization problem, where the discriminator gets maximized and generator gets minimized.

$$\min_G \max_D f(D, G) = \mathrm{E}_{x \sim p_{data}} \left|\log D(x)\right| + \mathrm{E}_{x \sim p_z} \left\lfloor \left|\log\left(1 - D(G(z))\right)\right| \right\rfloor \tag{12.1}$$

where G is represented as the generator, D is represented as the discriminator, and $f(D, G)$ is represented as the objective function.

The final layer in the second network uses an activation function, i.e., sigmoid activation function: $D(x)$, $D(G(z)) \in [0, 1]$. The activation functions are maximized, where the error is minimized during predict w.r.t. target values on real/fake samples. On contrary, the generator reduces the discriminator to predict the fake samples. Hence, the generator loss directly depends on the discriminator performance.

12.4.1 Optimization of GAN using RNN

Consider a sample (a, b), where a is the input and b is the label with class C. The probability estimation using the RNN model is defined as $Y = \mathrm{softmax}(f(a; \theta))$. The study uses cross-entropy objective function that helps in updating the weights of RNN through backward learning, i.e., backpropagation during the process of training the classifier, which is seen in Fig. 12.2. The fitness function is selected to be robust with improved probability estimator that increases the performance of GAN-RNN classifier with probable weights update. The weights are updated optimally as in Ref. [36].

Consider $L(b, Y)$ as the loss function and RNN is associated with finding the best weights $\{w_1, w_2, \ldots, w_K\}$ that combine the results of loss function at each iteration.

The study uses a cross-entropy loss function to validate the performance of the GAN-RNN classifier (Fig. 12.3). The error produced by validating the model lies between the range $[-22]$ (Fig. 12.4). The cross-entropy increases the prediction probability of the

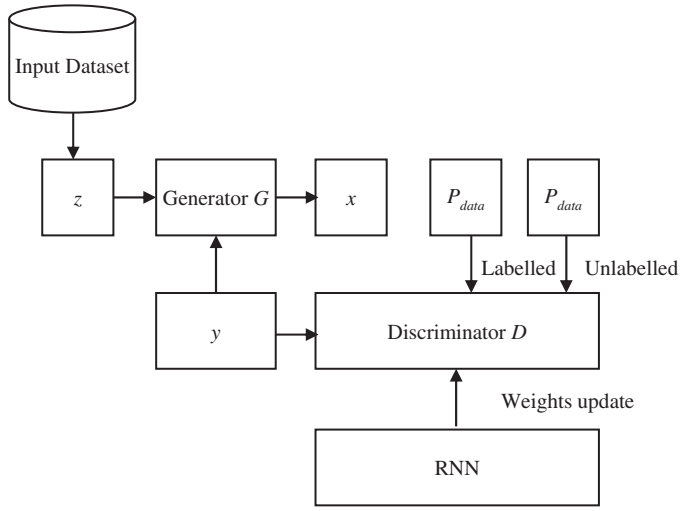


Fig. 12.2 GAN-RNN learning module.

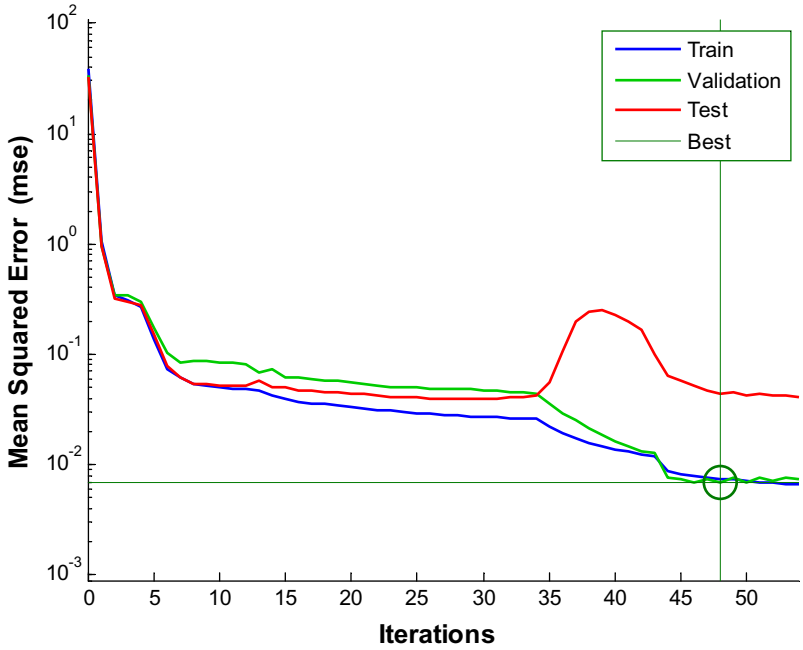


Fig. 12.3 Average performance of GAN-RNN model.

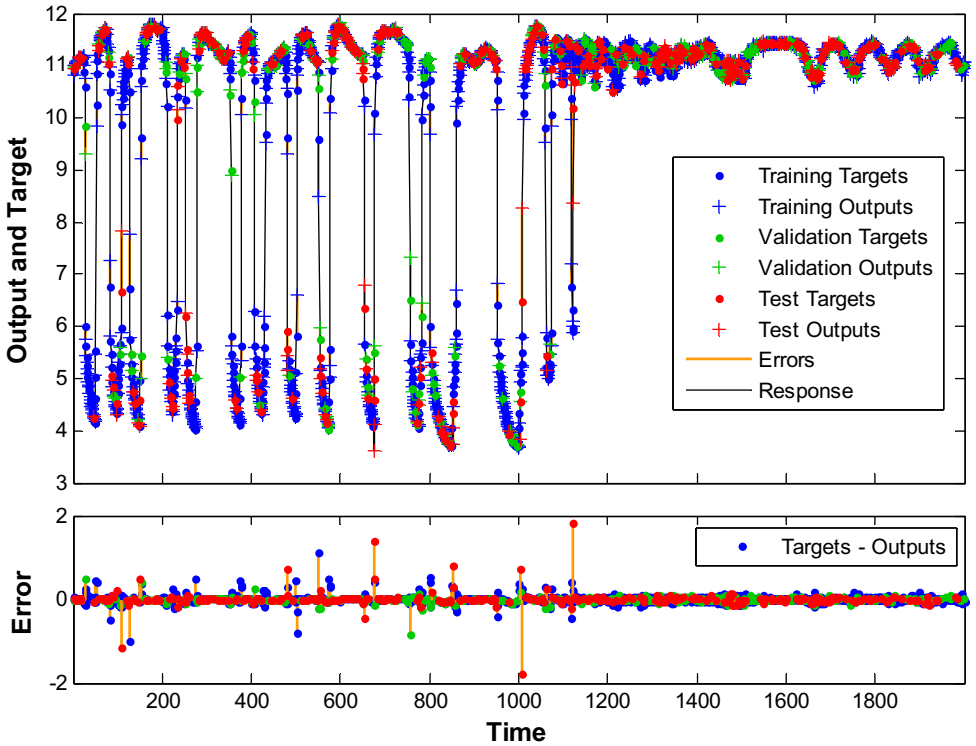


Fig. 12.4 Average error and plots of entire datasets.

input labels or the actual labels. The reduction of average values of the loss function is the maximization of log-likelihood of the input data. The cross-entropy loss function is defined using the following expression:

$$C_E = -\sum_i (b \log(w_i) + (1-b)\log(1-w_i)) \tag{12.2}$$

$$f = -\sum_{c=1}^{M} C_E(c, Y) \log C_E(c, Y) \tag{12.3}$$

where M represents the total number of class, log represents the natural logarithm, γ represents the binary indicator with c as the class label with accurate classified result over an observed data O, and p represents the predicted observation O from the class c.

With the training samples (N), the loss function is defined as

$$\min_w f = -\sum_{i=1}^{M} f(b_i, Y_i) \tag{12.4}$$

where b represents the true label and Y represents the predicted label.

12.5 Performance evaluation

This section provides the results of proposed GAN-RNN model with various other GAN models that include: GAN [26], Wasserstein GAN [30], deep-GAN [29], D-GAN [31], and MTL-GAN [23].

12.5.1 Dataset collection

The multivariate datasets are mixed of integer and real-valued attribute characteristics and the categorical attributes are removed and this is used to evaluate the GAN-RNN model including: Dataset-1: breast cancer Wisconsin (original) with 699 instances, Dataset-2: breast cancer Wisconsin (diagnostic) with 569 instances, Dataset-3: heart disease with 303 instances, Dataset-4: heart failure clinical records with 299 instances, Dataset-5: diabetes 130-US hospitals with 100,000 instances, and Dataset-6: lung cancer with 32 instances [50]. The features or classes used in this chapter are given in brief in Tables 12.1 and 12.2.

12.5.2 Performance metrics

The performance of GAN-RNN model is validated against various metrics that include accuracy, sensitivity, specificity, F-measure, geometric-mean (G-mean), percentage error, and training performance. The definitions of all the performance metrics are given below.

Table 12.1 Attributes of first three datasets for prediction.

Attributes	Dataset-1	Dataset-2	Dataset-3
Data set characteristics	Multivariate	Multivariate	Multivariate
Attribute characteristics (*categorical attributes are removed)	Integer	Real	Integer and real
Number of instances	699	569	303
Number of attributes	10	32	75
Missing values or errors present	Yes	No	Yes

Table 12.2 Attributes of last three datasets for prediction.

Attributes	Dataset-4	Dataset-5	Dataset-6
Data set characteristics	Multivariate	Multivariate	Multivariate
Attribute characteristics (*categorical attributes are removed)	Integer and real	Integer	Integer
Number of instances	299	100,000	32
Number of attributes	13	55	56
Missing values or errors present	No	Yes	Yes

Accuracy to validate the GAN-RNN is the total accurate samples predictions required to ensure that the GAN-RNN models predict the output correctly without noise. Hence, it can be defined as the ratio of total correct predictions vs total predictions.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{12.5}$$

where TP is the true positive cases, TN is the true negative cases, FP is the false positive cases, and FN is the false negative cases.

F-measure to validate the GAN-RNN is defined as the weighted harmonic mean of the sensitivity and specificity values. It tends to range between zero and one.

$$\text{F} - \text{measure} = \frac{2TP}{2TP + FP + FN} \tag{12.6}$$

It is inferred that the higher the F-measure value is, the higher the performance of GAN-RNN classifier would be.

Sensitivity is defined as the ability of the GAN-RNN model to correctly identify the true positive rate.

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{12.7}$$

Specificity is defined as the ability of the GAN-RNN model to correctly identify the true negative rate.

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{12.8}$$

Geometric-mean (G-mean) to validate the GAN-RNN is defined as the aggregation of both specificity and sensitivity measures that maintain the trade-off between the measures if the dataset is imbalanced. It is measured using the following equation:

$$\text{G} - \text{mean} = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{TN + FP}} \tag{12.9}$$

Higher the G-mean is, higher is the performance of GAN-RNN and vice versa.

Mean absolute percentage error (*MAPE*) to validate the GAN-RNN is defined as the performance measure of prediction accuracy to estimate the total eliminated losses during the prediction of accurate data instances from the preprocessed datasets. Thus, MAPE is defined as the ratio of the difference between the actual classes (A_t) and predicted classes (F_t), and the actual class. The MAPE value is then multiplied with 100 to find the percentage errors and finally, it is divided with the fit points or actual data points (n). The percentage error is hence defined as

$$\text{MAPE} = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right| \tag{12.10}$$

12.5.3 Discussions

Table 12.3 shows the results of accuracy between the proposed GAN-RNN (Method-6) with benchmark GAN (Method-1), Wasserstein GAN (Method-2), Deep-GAN (Method-3), D-GAN (Method-5), and MTL-GAN (Method-5). The results are

Table 12.3 Accuracy of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6.

(a)

Methods	Dataset-1
GAN	96.6507
W_GAN	97.3779
Deep_GAN	97.4082
D_GAN	97.489
MTL_GAN	97.4991
GAN_RNN	97.6304

(b)

Methods	Dataset-2
GAN	97.9334
W_GAN	97.9536
Deep_GAN	97.9738
D_GAN	97.9738
MTL_GAN	97.9839
GAN_RNN	97.9839

(c)

Methods	Dataset-3
GAN	96.0952
W_GAN	96.1154
Deep_GAN	96.1255
D_GAN	96.2164
MTL_GAN	96.2366
GAN_RNN	96.2972

(d)

Methods	Dataset-4
GAN	97.6395
W_GAN	97.6395
Deep_GAN	97.7203
D_GAN	97.7203
MTL_GAN	97.7405
GAN_RNN	97.791

Continued

Table 12.3 Accuracy of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6—cont'd

(e)

Methods	Dataset-5
GAN	94.1348
W_GAN	94.2863
Deep_GAN	94.3671
D_GAN	94.4176
MTL_GAN	94.5994
GAN_RNN	94.6095

(f)

Methods	Dataset-6
GAN	97.5698
W_GAN	97.5698
Deep_GAN	97.6506
D_GAN	97.6506
MTL_GAN	97.6708
GAN_RNN	97.7213

validated against various datasets as discussed in Section 12.5. The results validated against the existing model over each dataset show that the GAN-RNN obtains higher classification accuracy than other existing GAN methods. The results of simulation over the large Dataset-5 show that the GAN-RNN and other existing methods suffer from reduced accuracy and this is due to the complexity in computing the larger data features, however, the GAN-RNN has proven to be operating with reduced complexity than other methods.

Table 12.4 shows the results of sensitivity accuracy between the proposed GAN-RNN with existing GAN models over various datasets. The results validated against the existing model show that the GAN-RNN obtains higher sensitivity accuracy than other existing GAN methods. The sensitivity over the large Dataset-5 is lesser on all GAN methods. However, the GAN-RNN has proven to be operating with higher sensitivity than other methods.

Table 12.5 shows the results of specificity accuracy between the proposed GAN-RNN with existing GAN models over various datasets. The results validated against the existing model show that the GAN-RNN obtains higher specificity accuracy than other existing GAN methods. The specificity over the large Dataset-5 is lesser on all GAN methods. However, the GAN-RNN has proven to be operating with higher specificity than other methods.

Table 12.6 shows the results of F-measure accuracy between the proposed GAN-RNN with existing GAN models over various datasets. The results validated against

Table 12.4 Sensitivity of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6.

(a)

Methods	Dataset-1
GAN	65.6131
W_GAN	67.0181
Deep_GAN	69.8168
D_GAN	72.5236
MTL_GAN	82.0721
GAN_RNN	85.7091

(b)

Methods	Dataset-2
GAN	86.0626
W_GAN	92.7609
Deep_GAN	93.8729
D_GAN	94.4587
MTL_GAN	94.7415
GAN_RNN	94.7516

(c)

Methods	Dataset-3
GAN	62.5821
W_GAN	63.3194
Deep_GAN	64.2193
D_GAN	64.5829
MTL_GAN	66.4211
GAN_RNN	67.2605

(d)

Methods	Dataset-4
GAN	87.6594
W_GAN	87.6594
Deep_GAN	88.4775
D_GAN	88.6088
MTL_GAN	89.3562
GAN_RNN	89.4269

(e)

Methods	Dataset-5
GAN	61.3297
W_GAN	61.6933
Deep_GAN	62.2185

Continued

Table 12.4 Sensitivity of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6—cont'd

(e)

Methods	Dataset-5
D_GAN	63.8456
MTL_GAN	64.3506
GAN_RNN	64.6738

(f)

Methods	Dataset-6
GAN	87.7241
W_GAN	87.7241
Deep_GAN	88.5432
D_GAN	88.6745
MTL_GAN	89.4219
GAN_RNN	89.4936

Table 12.5 Specificity of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6.

(a)

Methods	Dataset-1
GAN	93.9537
W_GAN	93.9739
Deep_GAN	93.9739
D_GAN	93.9739
MTL_GAN	93.9739
GAN_RNN	94.6203

(b)

Methods	Dataset-2
GAN	93.7214
W_GAN	94.5193
Deep_GAN	94.7213
D_GAN	94.8021
MTL_GAN	94.8223
GAN_RNN	94.8829 up

(c)

Methods	Dataset-3
GAN	91.8923
W_GAN	91.9529
Deep_GAN	91.9933
D_GAN	93.2366
MTL_GAN	93.6507
GAN_RNN	94.0042

(d)

Methods	Dataset-4
GAN	94.7304
W_GAN	94.7405
Deep_GAN	94.8314
D_GAN	94.8314
MTL_GAN	94.9223
GAN_RNN	94.9223

(e)

Methods	Dataset-5
GAN	92.4276
W_GAN	92.5892
Deep_GAN	92.5993
D_GAN	92.6498
MTL_GAN	92.6902
GAN_RNN	92.7407

(f)

Methods	Dataset-6
GAN	94.6607
W_GAN	94.6708
Deep_GAN	94.7617
D_GAN	94.7617
MTL_GAN	94.8526
GAN_RNN	94.8526

Table 12.6 F-measure of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6.

(a)

Methods	Dataset-1
GAN	91.3151
W_GAN	92.7604
Deep_GAN	92.9119
D_GAN	93.427
MTL_GAN	93.6391
GAN_RNN	94.2855

Continued

Table 12.6 F-measure of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6—cont'd

(b)

Methods	Dataset-2
GAN	79.383
W_GAN	79.5143
Deep_GAN	80.0395
D_GAN	81.1313
MTL_GAN	81.8181
GAN_RNN	82.111

(c)

Methods	Dataset-3
GAN	60.0657
W_GAN	71.6534
Deep_GAN	71.9777
D_GAN	74.8875
MTL_GAN	78.1508
GAN_RNN	81.3838

(d)

Methods	Dataset-4
GAN	87.971
W_GAN	88.0922
Deep_GAN	90.0526
D_GAN	90.0829
MTL_GAN	91.4565
GAN_RNN	91.4666

(e)

Methods	Dataset-5
GAN	54.1239
W_GAN	61.8736
Deep_GAN	62.217
D_GAN	62.6816
MTL_GAN	64.2279
GAN_RNN	64.3693

(f)

Methods	Dataset-6
GAN	88.0326
W_GAN	88.1538
Deep_GAN	90.1152
D_GAN	90.1455
MTL_GAN	91.5201
GAN_RNN	91.5313

the existing model show that the GAN-RNN obtains higher F-measure accuracy than other existing GAN methods. The F-measure over the large Dataset-5 is lesser on all GAN methods. However, the GAN-RNN has proven to be operating with higher F-measure than other methods.

Table 12.7 shows the results of G-mean accuracy between the proposed GAN-RNN with existing GAN models over various datasets. The results validated against the existing model show that the GAN-RNN obtains higher G-mean accuracy than other existing GAN methods. The G-mean over the large Dataset-5 is lesser on all GAN methods. However, the GAN-RNN has proven to be operating with higher G-mean than other methods.

Table 12.8 shows the results of MAPE accuracy between the proposed GAN-RNN with existing GAN models over various datasets. The results validated against the existing

Table 12.7 G-mean of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6.

(a)

Methods	Dataset-1
GAN	95.4278
W_GAN	98.8628
Deep_GAN	99.4183
D_GAN	99.7213
MTL_GAN	99.8627
GAN_RNN	99.8627

(b)

Methods	Dataset-2
GAN	96.1954
W_GAN	96.1954
Deep_GAN	96.5691
D_GAN	96.6398
MTL_GAN	96.9731
GAN_RNN	97.0135

(c)

Methods	Dataset-3
GAN	81.1616
W_GAN	81.6767
Deep_GAN	82.212
D_GAN	82.4443
MTL_GAN	83.5664
GAN_RNN	84.031

Continued

Table 12.7 G-mean of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6—cont'd

(d)

Methods	Dataset-4
GAN	96.2621
W_GAN	96.2621
Deep_GAN	96.6368
D_GAN	96.7075
MTL_GAN	97.0408
GAN_RNN	97.0812

(e)

Methods	Dataset-5
GAN	81.4545
W_GAN	81.6969
Deep_GAN	81.9696
D_GAN	82.9796
MTL_GAN	83.3028
GAN_RNN	83.4957

(f)

Methods	Dataset-6
GAN	83.9301
W_GAN	84.7885
Deep_GAN	86.4257
D_GAN	88.0013
MTL_GAN	93.0937
GAN_RNN	94.6289

Table 12.8 MAPE of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6.

(a)

Methods	Dataset-1
GAN	28.4113
W_GAN	27.0064
Deep_GAN	24.2077
D_GAN	21.5108
MTL_GAN	11.9523
GAN_RNN	10.2453

Table 12.8 MAPE of various GAN with (a) Dataset-1, (b) Dataset-2, (c) Dataset-3, (d) Dataset-4, (e) Dataset-5, and (f) Dataset-6—cont'd

(b)

Methods	Dataset-2
GAN	80.7386
W_GAN	20.5224
Deep_GAN	13.6918
D_GAN	2.60904
MTL_GAN	48.4075
GAN_RNN	14.4503

(c)

Methods	Dataset-3
GAN	31.4423
W_GAN	30.704
Deep_GAN	29.8051
D_GAN	29.4516
MTL_GAN	27.6023
GAN_RNN	26.7741

(d)

Methods	Dataset-4
GAN	32.6947
W_GAN	32.3311
Deep_GAN	31.8059
D_GAN	30.1788
MTL_GAN	29.6738
GAN_RNN	29.3506

(e)

Methods	Dataset-5
GAN	72.7857
W_GAN	72.6039
Deep_GAN	64.6269
D_GAN	63.2826
MTL_GAN	55.851
GAN_RNN	55.1632

(f)

Methods	Dataset-6
GAN	27.7352
W_GAN	27.5534
Deep_GAN	19.5814
D_GAN	18.2381
MTL_GAN	10.8116
GAN_RNN	10.1248

model show that the GAN-RNN obtains reduced MAPE accuracy than other existing GAN methods. The MAPE over the Dataset-5 is highest of all datasets due to its increased dataset size. However, the GAN-RNN has proven to be operating with reduced MAPE than other methods.

Further, the proposed method is tested with a high-dimensional dataset to test the efficacy and robustness of the proposed model. The proposed method is tested additionally tested with dataset-5 by varying the number of training data and the results are given below.

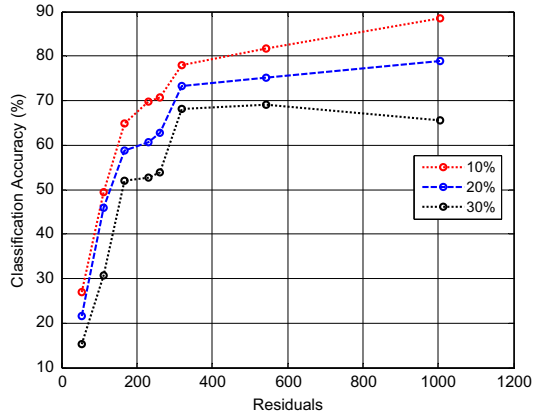
From Fig. 12.5, we found that the performance of the proposed system with 90% training data performs the best with various class labels. Hence, we conclude that with increasing training labels, the supervised method performs the best on reduced noise labels, which are evident from the figure. We hence choose the 90% training data with 10% noise labels for further evaluation.

The result shows that the proposed method on all three categories has higher performance with data stability than the other models as reported in Tables 12.9–12.11. The result shows that with 90% training data, we found that the stability of the system is higher than that of other models.

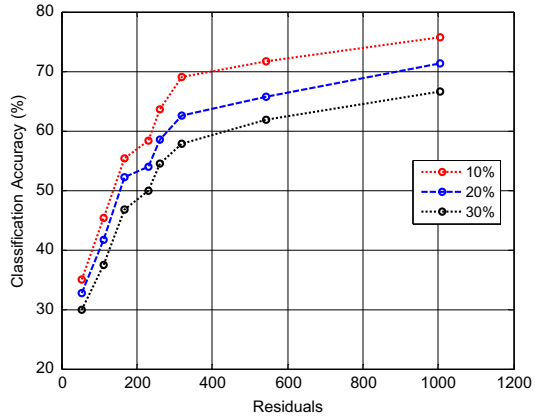
With 90% of the training data, we found the GAN with RNN generates optimal synthetic samples from the datasets. The classification of data using discriminator network on the real data distribution shows accurate transformation of noise vector into new samples. This resolves the problems of optimization and the use of a cross-entropy objective function updates the RNN weights using through backward learning that increases the robust of the classifier with minimal losses. However, with reduction of training data, the performance tends to degrade as the noise vector transformation is less accurate, which affects the stability of the system.

12.6 Conclusions

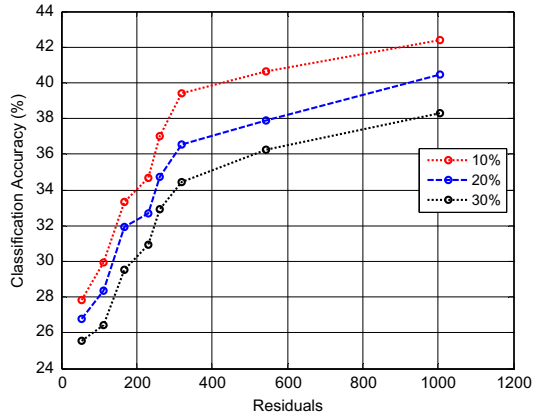
In this chapter, the scaling of GAN is improved with RNN that effectively learns the network recursively for classification. The recursive learning from the past records and neighborhood relationship reduces the error rate, while analyzing the time series data. The backward propagation of the GAN with updated network weights has explicitly enabled the output estimation. The target output is projected by the GAN-RNN model and most accurate predictions are thus made. Therefore, the probability of identifying the false samples is reduced using RNN discriminator operated with a gradient operator-based objective function. The results of the simulation show that the GAN-RNN (97.01%) model has a higher classification rate than other existing methods such as benchmark GAN (96.67%), Wasserstein GAN (96.82%), Deep-GAN (96.87%), D-GAN (96.91%), and MTL-GAN (96.96%). The training of the model using synthetic datasets has enabled higher predictive results with reduced false classification rate. In future, the system may choose an optimal loss function on contributing with adversarial training.



(A) 60% trainingdata



(B) 75% trainingdata



(C) 90% trainingdata

Fig. 12.5 Comparison of GAN-RNN accuracy with varying training data over Dataset-5 with varying training labels.

Table 12.9 Comparison of other parameters GAN-RNN with 60% training data with 10% label noise.

Metrics	GAN	W_GAN	Deep_GAN	D_GAN	MTL_GAN	GAN_RNN
F-measure	75.1937	75.4599	75.6402	75.8311	80.668	86.5453
G-mean	75.5129	75.7568	77.5809	79.7666	82.3765	85.2939
MAPE	73.5287	69.6674	62.5377	43.3257	40.2916	38.3286
Sensitivity	83.5006	76.7431	77.5066	79.3848	79.5969	86.8634
Specificity	75.9477	77.9414	81.1982	86.7468	88.1371	88.5719

Table 12.10 Comparison of GAN-RNN with 75% training data with 10% label noise.

Metrics	GAN	W_GAN	Deep_GAN	D_GAN	MTL_GAN	GAN_RNN
F-measure	42.0106	44.2377	56.1514	56.321	58.8355	90.0036
G-mean	78.2172	78.4622	80.0529	80.0953	80.5301	92.0408
MAPE	31.3378	28.2188	26.7341	23.997	23.3819	18.3954
Sensitivity	66.7712	70.4946	78.8757	92.009	92.7089	98.3610
Specificity	79.9575	80.159	83.8824	83.9036	85.3575	86.2483

Table 12.11 Comparison of GAN-RNN with 90% training data with 10% label noise.

Metrics	GAN	W_GAN	Deep_GAN	D_GAN	MTL_GAN	GAN_RNN
F-measure	72.0748	72.1278	73.0727	74.2605	79.5969	85.4954
G-mean	47.4309	61.1378	64.3522	48.7045	81.9841	92.4756
MAPE	21.8431	19.0105	18.9151	13.7695	12.3591	11.1395
Sensitivity	82.1325	84.9757	85.0712	90.2051	91.6272	92.8468
Specificity	79.1197	82.175	83.1718	86.5241	88.6461	90.8636

References

[1] M.Y. Liu, O. Tuzel, Coupled generative adversarial networks, in: Advances in Neural Information Processing Systems, 2016, pp. 469–477.
[2] H. Zhang, I. Goodfellow, D. Metaxas, A. Odena, Self-attention generative adversarial networks, in: International Conference on Machine Learning, 2019, May, pp. 7354–7363.
[3] C. Li, M. Wand, Precomputed real-time texture synthesis with Markovian generative adversarial networks, in: European Conference on Computer Vision, Springer, Cham, 2016, October, pp. 702–716.
[4] T. Karras, S. Laine, T. Aila, A style-based generator architecture for generative adversarial networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 4401–4410.
[5] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, D.N. Metaxas, Stackgan: text to photo-realistic image synthesis with stacked generative adversarial networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5907–5915.
[6] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A.A. Bharath, Generative adversarial networks: an overview, IEEE Signal Process. Mag. 35 (1) (2018) 53–65.