Limitations and Applicability of GANs in Banking Domain

Anubha Pandey [1] and **Deepak Bhatt** [2] and **Tanmoy Bhowmik** [3]

Abstract. Threats due to payment-related frauds are always a primary concern for financial institutions (FIs), often leading to huge losses and impacting consumer experience. To combat emerging frauds and improve the system's robustness, FIs need an efficient system to detect fraud while authorizing payments. The biggest challenge in developing a fraud detection system is a high degree of class imbalance between fraudulent and legitimate transactions. Recently, Generative Adversarial Networks (GANs) are employed as an oversampling technique to augment the dataset with synthetic minority samples.

In this paper, we present a systematic study to train GANs for synthetic fraud generation, demonstrating improved classifier performance detecting fraud. Training of GANs is conducted in various settings, including min-max objective and with or without auxiliary loss discriminating synthetic fraud and real fraud from non-fraud samples. Auxiliary loss is obtained using contrastive loss or triplet loss. Quality of trained GANs is estimated by evaluating the lift in classifier performance when trained with dataset augmented with synthetic fraud. Further, the effect of Discriminator Rejection Sampling (DRS) is studied in synthetic sample selection used for training data augmentation. The performance comparison of different settings proposed in this study is evaluated using a publicly available Credit-Card dataset and showed an absolute improvement of up to 6% in Recall and 3% in precision. We hope this paper will help advance the applicability of GANs with a practical insight into the research that has been done on this topic so far and open doors to interesting future research direction.

1 INTRODUCTION

Credit card has become a ubiquitous method for online payment. Consequently, the increase in more sophisticated fraudulent transactions is alarming. The fraudulent transactions affect the user level and business level, resulting in financial loss and customer trust. Banks and fintech companies need an efficient system to monitor the massive volume of transaction logs and detect the frauds[6, 27]. However, it should not decline legitimate transactions affecting consumer experience.

The commonly used pipeline for the fraud detection system employs a binary classifier to distinguish samples of fraud transactions from the legitimate ones [26, 12, 36]. The fraudulent transactions are rare; they represent a tiny fraction of activity within an organization, resulting in class imbalance. The class imbalance issue makes the binary classifiers biased towards the majority class and hence makes the fraud detection a challenging problem [23, 22, 7]. A similar high degree of class imbalance is observed in a variety of real-world applications like medical diagnosis, information retrieval system, bioinformatics [31, 1, 34, 17, 21, 38, 39].

There exist several techniques for the class imbalance learning [35, 24, 15, 36]. [30] has done a comparative study of several supervised and unsupervised machine learning algorithms to handle the class-imbalance in credit card fraud detection. One of the solutions to the class imbalance problem is to re-balance the training sets used by the binary-classifier [2, 11, 21]. There exist several oversampling techniques that have proved to be effective in handling class imbalance. The commonly used methods are variants of SMOTE(Synthetic Minority Oversampling Techniques) [9, 20, 8]. The SMOTE aims to generate samples along the line between two samples of the minority class. However, these methods generate synthetic samples based on the existing samples in the dataset and fail to capture minority class distribution. Hence can't detect new fraudulent transactions.

Recently, Generative Adversarial Networks (GAN) [16, 29] have received a lot of attention from the research community of credit card fraud detection. Several works [14, 33, 5, 40] have shown the efficacy of GAN for augmenting the dataset with synthetic minority (fraud) samples. However, mode collapse is a common phenomenon that occurs with GANs. Mode collapse happens when GAN generates limited varieties of samples and hence fails to capture the whole data distribution. To overcome the issue of mode collapse, researchers [33, 5] have used different architectures of GAN like WGAN[3], Least Square GAN[28], Relaxed WGAN[19] to augment the dataset and have shown an improvement in the classifier's performance. On the other hand, [40] has trained a GAN based architecture to generate complementary samples of the majority class(legitimate transactions). They have used a combination of two WGANs and two Autoencoders and use a three-phase training process for fraud detection.

In this paper, a comprehensive study of several existing techniques to train GANs in fraud detection scenario is conducted along with highlighting their merits and demerits. We have shown experiments on conditional WGAN-GP for the generation of fraudulent data conditioned separately on class labels for fraud samples obtained from k-means clustering or non-fraud samples from the training set. It is observed that just using GANs may lead to boundary distortion hence leading to a drop in the performance for the majority class (legitimate transactions). We have proposed an auxiliary loss using Triplet Network and Siamese Network separately on top of the WGAN-GP model to learn more discriminative fraud samples. Further, the effect in the quality of synthetic samples is studied when the WGAN-GP network is trained in an end to end fashion along with a neural network-based classifier and found to be useful for dealing with the boundary distortion problem. All the models are simple architecture with few parameters and are trained end-to-end for the generation of

[1] Mastercard, India, email: Anubha.Pandey@mastercard.com
[2] Mastecard, India, email: Deepak.Bhatt@mastercard.com
[3] Mastercard, India, email: Tanmoy.Bhowmik@mastercard.com

fraudulent data as compared to [40]. We have further shown the applicability of Discriminator Rejection Sampling [4] to improve the quality of the synthetic fraud samples used for data augmentation. In the later section, we have highlighted an open problem in data augmentation, which is how to decide on the number of synthetic fraud samples for data augmentation.

The paper is organized as follows. In section 2, we have described several configurations used to train the WGAN-GP model for improved data augmentation. The structural details of these configurations are provided in section 3 along with the dataset description, and other experimental settings followed. Section 4 compares the performance of all the models and visualizes the synthetic samples obtained for data augmentation. It also talks about the effect of increasing the number of synthetic samples in the augmented set on the classifier's performance. Finally, section 5 concludes the article and provides a possible future research direction.

2 METHODOLOGY

2.1 Fraud detection framework

Fraud Detection is formulated as a binary classification problem. For each transaction record in the dataset, we have a feature vector and corresponding class label (fraud or non-fraud).The commonly used pipeline for credit card fraud detection using generative models[13, 33, 14, 5] is described below:

1. Train a GAN to generate the fraudulent samples from the train set.
2. Augment the training set with the synthesized fraud samples.
3. Train a classifier on the original and augmented training set separately and compare the performances.

2.2 Data augmentation using different configurations of WGAN-GP

2.2.1 WGAN-GP

We use a WGAN-GP [18] architecture to oversample from the fraudulent (minority) class. It has a Generator module $G : \mathbf{Z} \rightarrow \mathbf{X}$ parameterized by $\theta_{\mathbf{G}}$ and a discriminator module $D : \mathbf{X} \rightarrow [0, 1]$ parameterized by $\theta_{\mathbf{D}}$. Where, $\mathbf{Z}$ is a set of random noise vector sampled from unit Gaussian distribution $\mathcal{N}(0, 1)$ and $\mathbf{X}$ is a set of the feature vector of the fraud samples. Below are the loss functions to train the Discriminator(D) and Generator(G) module in WGAN-GP:

$$L_D = \frac{1}{n}\sum_{i=1}^{n}(D_{\theta_{\mathbf{D}}}(\tilde{\mathbf{x}}_{\mathbf{f_i}}) - D_{\theta_{\mathbf{D}}}(\mathbf{x}_{\mathbf{f_i}}) + \lambda(||\nabla_{\tilde{\mathbf{x}}} D_{\theta_{\mathbf{D}}}(\tilde{\mathbf{x}}_{\mathbf{f_i}})||_2 - 1)^2 \tag{1}$$

where, $\tilde{\mathbf{x}}_{\mathbf{f_i}} = t\hat{\mathbf{x}}_{\mathbf{f_i}} + (1-t)\mathbf{x}_{\mathbf{f_i}}$ with $0 \leq t \leq 1$

.

$$L_G = \frac{1}{n}\sum_{i=1}^{n}(-D_{\theta_{\mathbf{D}}}(G_{\theta_{\mathbf{G}}}(\mathbf{z_i}))) \tag{2}$$

Where, $\hat{\mathbf{x}}_{\mathbf{f_i}}$ and $\mathbf{x}_{\mathbf{f_i}}$ are the generated and real fraud samples respectively and $\mathbf{z}$ is a random noise sample.

2.2.2 Conditional WGAN-GP

We add conditions to WGAN-GP [5] , as shown in Figure 1, and extend the input space of the model. $G : \mathbf{Z} \times \mathbf{Y} \rightarrow \mathbf{X}$
$D : \mathbf{X} \times \mathbf{Y} \rightarrow [0, 1]$
Where $\mathbf{Y}$ is the set of conditions corresponding to the features in $\mathbf{X}$ set. We conduct two separate experiments with different conditional variables, one with class labels of the fraud samples obtained using k-means clustering and second with the non-fraud samples in the training set. The loss functions for the Discriminator(D) and Generator(G) module in conditional WGAN-GP are described below:

$$L_D = \frac{1}{n}\sum_{i=1}^{n}(D_{\theta_{\mathbf{D}}}(\tilde{\mathbf{x}}_{\mathbf{f_i}}, \mathbf{y}_{\mathbf{f_i}}) - D_{\theta_{\mathbf{D}}}(\mathbf{x}_{\mathbf{f_i}}, \mathbf{y}_{\mathbf{f_i}}) + \lambda(||\nabla_{\tilde{\mathbf{x}}} D_{\theta_{\mathbf{D}}}(\tilde{\mathbf{x}_{\mathbf{f_i}}}, \mathbf{y}_{\mathbf{f_i}})||_2 - 1)^2 \tag{3}$$

$$L_G = \frac{1}{n}\sum_{i=1}^{n}(-D_{\theta_{\mathbf{D}}}(G_{\theta_{\mathbf{G}}}(\mathbf{z_i}, \mathbf{y}_{\mathbf{f_i}}), \mathbf{y}_{\mathbf{f_i}})) \tag{4}$$

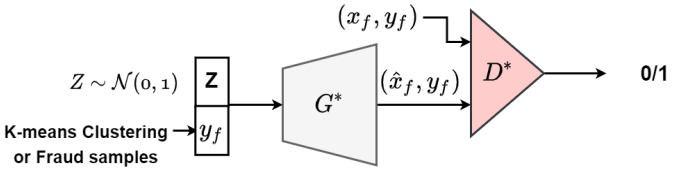

Figure 1. Conditional GAN

2.2.3 WGAN-GP with Siamese Network

Siamese Network [25] uses Contrastive divergence loss to minimize the distance between positive pairs and maximize the distance between negative pairs. We use it on top of the underlying WGAN-GP model, as shown in Figure 2 to ensure the distribution learned by the generator for the fraud samples does not overlap with the non-fraud samples. We train both the network in an end-to-end fashion.

Siamese Network has two Neural Network with the shared weights and maps the fraud (real and generated) and non-fraud samples into a shared space such that the distance between them is preserved. We pass the pairs of generated, and real fraud samples as positive pairs i.e., $(\hat{\mathbf{x}}_{\mathbf{f}}, \mathbf{x}_{\mathbf{f}}, \mathbf{l} = 1)$and generated fraud samples and real non-fraud samples as negative pairs i.e., $(\hat{\mathbf{x}}_{\mathbf{f}}, \mathbf{x}_{\mathbf{nf}}, \mathbf{l} = 0)$ to the Siamese Network S parameterized by $\theta_{\mathbf{S}}$ and train the generator and Siamese network on Contrastive divergence loss function as defined below:

$$L_S = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{l_i})\frac{1}{2}d(S_{\theta_{\mathbf{S}}}(\hat{\mathbf{x}}_{\mathbf{f_i}}), S_{\theta_{\mathbf{S}}}(\mathbf{x}_{\mathbf{f_i}}))^2 + (1-\mathbf{l_i})\frac{1}{2}\{max(0, m - d(S_{\theta_{\mathbf{S}}}(\hat{\mathbf{x}}_{\mathbf{f_i}}), S_{\theta_{\mathbf{S}}}(\mathbf{x}_{\mathbf{nf_i}})))\}^2 \tag{5}$$

where, d is the euclidean distance and m is the margin hyperparameter.

2.2.4 WGAN-GP with Triplet Network

The Triplet Network has three Neural Network with the shared weights and maps the fraud (real and generated) and non-fraud samples into a shared space such that the distance between them is preserved using triplet loss function. The objective of the triplet loss[32] is to minimize the distance between the generated fraud samples and real fraud samples and simultaneously maximize the distance between the generated fraud samples and real non-fraud samples; hence, it is a max-margin framework.

We pass the triplet generated fraud samples, real fraud samples, and real non-fraud samples, i.e., $(\hat{\mathbf{x}}_{\mathbf{f}}, {\mathbf{x}_{\mathbf{f}}}^+, {\mathbf{x}_{\mathbf{nf}}}^-)$ to the Triplet Network T parameterized by $\theta_{\mathbf{T}}$ and train the generator and Triplet Network on Triplet loss function as defined below:

$$L_T = \frac{1}{n}\sum_{i=1}^{n} max(0, m + d(T_{\theta_{\mathbf{T}}}(\hat{\mathbf{x}}_{\mathbf{f_i}}), T_{\theta_{\mathbf{T}}}(\mathbf{x}_{\mathbf{f_i}})) - d(T_{\theta_{\mathbf{T}}}(\hat{\mathbf{x}}_{\mathbf{f_i}}), T_{\theta_{\mathbf{T}}}(\mathbf{x}_{\mathbf{nf_i}}))) \tag{6}$$

where, d is the euclidean distance and m is the margin hyperparameter.

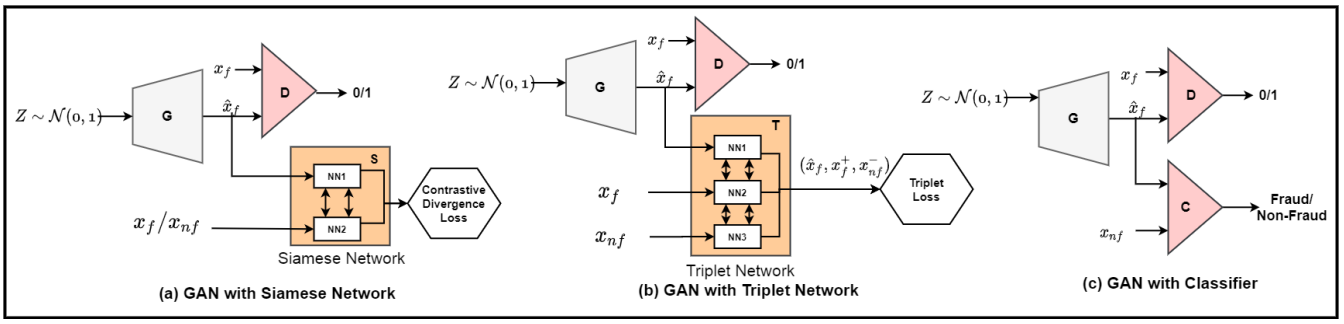

Figure 2. Different configurations used to train WGAN-GP architecture.

2.2.5 *WGAN-GP with Classifier*

We train the WGAN-GP model with a binary Classifier, as shown in Figure 2. We pass the generated fraud samples along with the real non-fraud samples into a classifier C parameterized by $\theta_{\mathbf{C}}$ and train the generator on the classification loss. In this configuration, we have two different classifiers in the network, C tries to distinguish samples from fraudulent (minority) and non-fraudulent (majority) transactions. Whereas, another classifier, discriminator(critic) D, tells how far is the learned distribution from the true distribution. This configuration of WGAN-GP and classifier ensures that the generated fraud samples do not overlap with the real non-fraud samples and, simultaneously, follow the distribution of the minority(fraud) class. We use binary cross-entropy loss to train the classifier and the generator module of the architecture, as defined below:

$$L = \sum_{i=1}^{n} -\log(C_{\theta_{\mathbf{C}}}(\hat{\mathbf{x}}_{\mathbf{f_i}})) - \log(C_{\theta_{\mathbf{C}}}(\mathbf{x}_{\mathbf{nf_i}})) \tag{7}$$

2.2.6 *WGAN-GP with Discriminator Rejection Sampling*

In the standard GAN, it is a common practice to discard the discriminator after training and generators are used for synthetic data generation. It is believed that after training a GAN, the generator perfectly captures the underlying data distribution. However, recent studies[4, 37] have shown that the GANs do not converge to the true data distribution, and the trained generator still generates samples that can be easily distinguished by the discriminator from the real sample. They have also shown that the discriminator captures the data distribution more closely than the generators. Hence, we should consider the distribution defined by both the generator and discriminator for better quality samples.

We use Discriminator Rejection Sampling(DRS) method [4], to sample from the distribution learned by the discriminator $p_d(\mathbf{x})$. We use DRS as a post-processing step, where we use the trained discriminator D^* to improve the synthetic fraud samples from the trained generator G^*.

3 EXPERIMENTAL SETUP

3.1 Dataset description

All the experiments uses publicly available Credit-Card dataset [11]. The dataset has transactions for two days done in September 2013 by the European cardholders. There are 284807 transactions in the dataset, out of which 492 are fraudulent transactions, i.e., the frauds account for 0.172% of the total transactions. 31 features represent the transactions, namely 'Amount,' 'Time,' 'Class,' and 28 other numerical features obtained from PCA (V1, V2,... V28). Feature 'Time' has time elapsed from the first transaction, and 'Class' has label 1 for fraudulent transactions and 0 otherwise. There is no missing values in the dataset. We use Log transformed 'Amount' values to give more normal distribution and normalize the features between 0 and 1. We divide the dataset into a train and test set such that the train set has 70% of the transactions in the dataset i.e., 199364 transactions, and the test set has 30% of the transactions in the dataset i.e., 85443 transactions. 344 and 148 fraud samples account for 0.173% of the total transactions in the training and testing set, respectively.

3.2 Architecture details

Lift in the performance of XGBoost classifier [10] is used as a metric to quantify the quality of generated synthetic samples when used for augmenting the training set. To evaluate various settings of WGAN-GP, we train it on different loss functions with the aim of generating more realistic fraud data. The architectural information of all the different configurations used is described below:

1. **WGAN-GP**
   The generator module of the WGAN-GP model has four fully connected layers with 30, 128, 256, 512 neurons, respectively, Relu is the activation function used after each layer except the last layer. The generator accepts the random noise of dimension 30 as input. The discriminator has a series of 5 fully connected layers with 30, 512, 256, 128, 1 neurons respectively in each layer, Relu is used as the activation function in all the layer except the last layer has Sigmoid activation. We train both the discriminator and generator module together on the loss function defined in Equation 1 and 2, respectively. We set LAMBDA = 10.0 in the Gradient-Penalty term. We iterate over the discriminator module 5 times per generator module update. We train the model using Adam Optimizer for 10000 epochs with mini-batch of size 64 and learning rate 2e-4. We observe the convergence of the model at around 4000 epoch.
2. **Conditional WGAN-GP**
   We use a k-means clustering algorithm to divide the fraud dataset into 2 clusters and label the fraud samples as 0 or 1 accordingly. We pass the labels of fraud samples as a condition to the WGAN-GP along with random noise as input and train the WGAN-GP model with the same architecture as defined above. To form pairs of fraud and non-fraud samples, we randomly pick samples from the respective classes and pair them. We trained the model on the loss function defined in Equation 3 and 4 and observed the model's convergence at 4000 epoch.
3. **WGAN-GP with Triplet Network**
   We form a triplet of the synthetic samples obtained from the generator with the real fraud samples and real non-fraud samples and pass them to the Triplet Network. The network has three neural

networks with shared weights. There are three fully connected layers with 30, 30, and 2 neurons, respectively. Each layer has a Relu activation except for the last layer. It uses Triplet loss to simultaneously ensure that the positive pair of generated and real fraud samples are close and the negative pair of generated fraud and real non-fraud samples are apart by some margin. We set the hyperparameter margin to 1 in the Triplet loss function defined in the Equation6. We trained the triplet network with the WGAN-GP model end-to-end using the Adam optimizer for 5000 epochs and observed the convergence at around 3500 epoch.

4. **WGAN-GP with Siamese Network**
   We use the same architecture of the WGAN-GP model, as mentioned above. We pair the synthetic samples obtained from the generator with the real fraud samples and real non-fraud samples to form positive and negative pairs simultaneously and pass it to the Siamese Network. The Siamese Network has two neural networks with shared weights. There are three fully connected layers with 30, 30, and 2 neurons, respectively. Each layer has Relu activation except for the last layer. It uses Contrastive divergence loss defined in Equation 5 to ensure that the positive pair of generated and real fraud samples are close and some margin separates the negative pair of generated fraud and real non-fraud samples. We set the hyperparameter margin to 1 and eps to 1e-9 in the Contrastive loss function defined in the Equation5. We train the entire network end-to-end using the Adam optimizer for 5000 epochs and observe the model saturation at around 3000 epochs.
5. **WGAN-GP with Classifier**
   In this experiment, we add a binary classifier module on top of the WGAN-GP model. We pass the generated fraud samples from the generator to the classifier module along with the real non-fraud samples from the training set. The classifier then distinguishes between the fraud and non-fraud samples. The classifier has three fully connected layers with 30, 30, and 2 neurons in each layer, respectively. All the layers have Relu activation, and the last layer has Softmax activation. We train the classifier and generator parameters on the loss function defined in Equation 7 using Adam optimizer with a learning rate of 0.001. Initially, we train only the WGAN-GP model for 1000 epochs. Later we train the entire network end-to-end for 5000 epochs and observe the model saturation at around 2500 epochs.

4 RESULTS

4.1 Performance metrics

In credit card fraud detection, the class of interest is the fraud (minority) class. Here, the cost of false positive and false negative are not equal. An ideal system should precisely identify the fraud samples while reducing the number of false positives. Accuracy is the ratio of samples correctly classified by the classifier, i.e. (TP+TN)/N. However, for the imbalanced dataset, accuracy is not the correct measure of the classifier's performance. We pay attention to the categorical prediction ability. Hence we report, Precision(specificity), Recall(sensitivity) and F1-Score to evaluate the performance of the model. Precision refers to the percentage of your results that are relevant, i.e., TP/(TP+FP). Recall refers to the percentage of total relevant results correctly classified by your algorithm. i.e., TP/(TP+FN). F1- score combine both the precision and recall metrics into one, and it is the harmonic mean of Precision and Recall.

The results of all the different configurations of WGAN-GP employed to solve the task of credit card fraud detection is illustrated in Table 1. First, we train an XGBoost classifier on the training set's transactions and test the performance on the testing set. Next, we use a WGAN-GP model to learn the distribution of fraud samples and used the trained generator of the WGAN-GP architecture to oversample the minority class (fraud) data and augment the training set. We further train an XGBoost classifier on the augmented training set and report the performance on the testing set. We can observe from Table1 that there is an absolute improvement of 5% in Recall in the XGBoost Classifier trained on the dataset augmented by WGAN-GP model as compared to the original dataset.

Augmentation Method		**Precision**	**Recall**	**F1-Score**
Without Augmentation		0.90	0.76	0.83
WGAN-GP		0.88	0.81	0.84
Conditional WGAN-GP	Labels from k-means	0.88	0.81	0.84
	Non-Fraud Samples	0.86	0.81	0.82
WGAN-GP + Triplet Network		0.89	0.82	0.85
WGAN-GP + Siamese Network		0.88	0.82	0.85
WGAN-GP + Classifier		0.92	0.78	0.84
WGAN-GP + DRS		0.90	0.82	0.86
WGAN-GP + Classifier + DRS		0.93	0.79	0.85

Table 1. Performance of XGBoost classifier trained on augmented set obtained from different configuration of WGAN-GP model.

We also use the conditional WGAN-GP model to generate fraud samples based on some conditions like class labels and non-fraud samples. Fraud samples are clustered into k classes using k-means clustering and corresponding cluster IDs are assigned as labels. In our experiment fraud samples are classified into 2 clusters. We pass these labels to the conditional WGAN-GP model as conditions to generate fraud samples. From Table 1, it can be observed that the performance of the classifier remains the same when trained on the augmented dataset obtained from the WGAN-GP model conditioned on labels from k-means clustering. In another setting where conditional WGAN-GP model is trained to learn the transformation of non-fraud samples to fraud did not perform better leading to absolute drop of 2% in Precision as observed in Table 1. Further investigation is required to identify the performance drop as the model output do not conform to our hypothesis where generating fraud from non-fraud samples should perform better.

The study proposed training of GANs with auxiliary loss functions using triplet loss or siamese network loss for effective synthetic data generation. Experimental results using both the loss function demonstrated an improvement in Recall by 1%. However, an absolute improvement of 2% and 1% can be observed in Precision with Triplet and Siamese Network, respectively, as compared to the simple WGAN-GP model. This further proves the benefits of incorporating auxiliary loss along with existing WGAN-GP training.

In WGAN-GP with classifier, the generative module is trained on two loss functions. The first loss corresponds to classifier which tries to distinguish between the fraud and non-fraud samples and another classification loss for discriminator that distinguish between the real and generated fraud samples. These two classifier modules, in turn, helps the generator to synthesize well-discriminative fraud samples that follow the fraud class distribution. Table 1 shows an absolute improvement of 3%in Precision and a reduction of 3% in the Recall compared to the simple WGAN-GP model. However, as compared to the XGBoost classifier trained on the original dataset, there is an improvement of 2% in Recall and Precision.

The performance of the XGBoost classifier trained on the augmented dataset depends on the quality of the generated fraud samples. Hence to improve the Recall, the generated fraud samples should be well-discriminative than the non-fraud samples. Recent

studies [4, 37] have shown that the samples generated from the trained generator are not similar to the real class samples, which discriminator would have otherwise rejected easily. We employ the discriminator rejection sampling method, proposed in [4]. The trained discriminator is used to filter out the poor-quality samples from the generator as a post-processing step and are used for training dataset augmentation. Table 1 shows an absolute improvement of 2% in Precision and 1% in Recall using discriminator rejection sampling with WGAN-GP to augment the dataset over the simple WGAN-GP model. However, compared to the XGBoost classifier trained on the original dataset, the performance is similar in Precision with a 6% absolute improvement in Recall.

A reduction in Precision may result in misclassification of legitimate transactions as fraudulent transactions, hence penalizing the banks in terms of customer trust and comfort. From the previous experiments, we have observed adding a classifier module on the WGAN-GP model results in an improvement in Precision. To improve the quality of samples injected into the augmented set, we used Discriminator Rejection Sampling (DRS) for all the configurations of the WGAN-GP model discussed above. With DRS on the WGAN-GP model, we observe an absolute improvement of 2% and 1% in Precision and Recall over the simple WGAN-GP model. In the case of the WGAN-GP+Classifier model, we observe an absolute improvement of 1% in both the Precision and Recall. However, for WGAN-GP+Triplet Network and WGAN-GP+Siamese Network model, no improvement was observed on applying DRS.

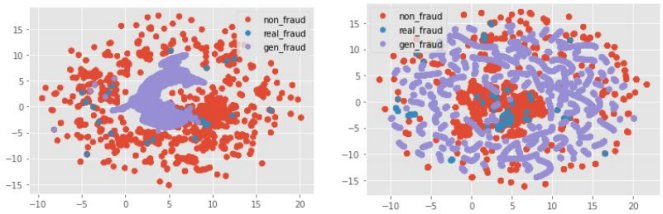

Figure 3. Samples generated from (a)WGAN-GP and (b)WGAN-GP+Classifier model

4.2 Comparison of samples generated by different models

A visualization of the distribution of fraud transactions learned by the simple WGAN-GP model and the WGAN-GP+Classifier model is shown in as shown in Figure 3. For this a 10000 syhnthetic fraud samples are drawn from both the trained models and plotted it against real fraud samples and 10000 real non-fraud samples from the training set. Figure 3 illustrates that the WGAN-GP model learns a class boundary from the fraud samples and sample synthetic fraud data from within the learned class boundary. Also, it can be observed that these samples are not uniformly distributed but are generated from the high population area. In the case of the WGAN-GP+Classifier model, Figure 3 shows that the generated fraud samples are uniformly distributed and more spread out as compared to the simple WGAN-GP model.

4.3 Effect of increasing the number of synthetic samples on the classifier's performance

There are 344 fraud samples in the training set, let us denote it by N_f. We generate fraud samples in the multiples ([1/4, 1/2, 1, 2, 4, 8, 16, 32, 64, 128, 256, 512]) of N_f to augment the dataset and study the effect on the classifier's performance. The Table 2 and Figure

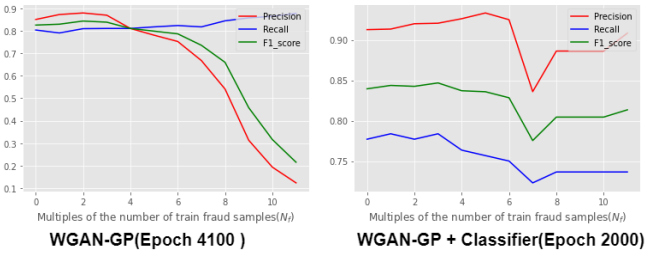

Figure 4. Effect of increasing generated fraud samples in the augmented set

4 shows the effect of increasing the generated fraud samples(N) in the augmented set on the classifier's performance. We report the values of performance metrics at Epoch 4100 for the WGAN-GP model and Epoch 2000 for the WGAN-GP+Classifier model. From Table 2, we observe the best performance of the WGAN-GP model when $N = N_f$, i.e., when the number of generated samples is equal to the number of real fraud samples. And for the WGAN-GP+Classifier model, the best performance is observed when $N = 2N_f$, i.e., when the number of generated samples is equal to twice the number of real fraud samples. Also, from Figure 4, we observe that as the number of generated fraud samples increases, the Recall of WGAN-GP model increases, but Precision and F1-Score drops. However, for the WGAN-GP+Classifier model, we can observe that the Precision and Recall drops after $4N_f$ and N_f, respectively.

N	WGAN-GP			WGAN-GP+Classifier		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
86	0.895	0.804	0.847	0.913	0.777	0.839
172	0.892	0.784	0.834	0.914	0.784	0.844
344	0.873	0.791	0.830	0.92	0.777	0.842
688	0.851	0.811	0.830	0.921	0.784	0.847
1376	0.811	0.811	0.811	0.926	0.764	0.837
2752	0.781	0.818	0.799	0.933	0.757	0.836
5504	0.753	0.824	0.787	0.925	0.75	0.828
11008	0.668	0.818	0.736	0.836	0.723	0.775
22016	0.541	0.845	0.660	0.886	0.736	0.804
44032	0.313	0.858	0.458	0.886	0.736	0.804
88064	0.193	0.865	0.316	0.886	0.736	0.804
176128	0.123	0.878	0.215	0.908	0.736	0.813

Table 2. Performance of XGBoost classifier as the number of generated samples(N) is varied in the augmented set obtained from WGAN-GP and WGAN-GP+Classifier model.

5 CONCLUSION AND FUTURE WORK

The paper presented a detailed study on applicability and effectiveness of GANs. Various GANs variants along with ones proposed in this study is compared to evaluate the efficacy of data augmentation for downstream classification task. Among different training procedures WGAN-GP when trained with a classifier in an end-to-end fashion performed well as shown in our study improving both precision and recall of XGBoost based fraud classifier. Further we found that Discrimiantor Rejection Sampling technique when applied for selection of synthetic samples generated using WGAN-GP with classifier provided an incremental lift. Next we also demonstrated the effect in the overall performance of fraud classifier with increase in synthetic samples used for training data augmentation. We believe the outcomes presented in this study would help readers in quickly identify the right settings of GANs utilised in fraud space.

A promising future research direction is to experiment with Reinforcement Learning based algorithm to automatically identify the quality and count of samples to be used for augmenting the training dataset leading to improved performance

REFERENCES

[1] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz, 'Applying support vector machines to imbalanced datasets', in *European conference on machine learning*, pp. 39–50. Springer, (2004).

[2] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz, 'Applying support vector machines to imbalanced datasets', in *European conference on machine learning*, pp. 39–50. Springer, (2004).

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou, 'Wasserstein gan', *arXiv preprint arXiv:1701.07875*, (2017).

[4] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena, 'Discriminator rejection sampling', *arXiv preprint arXiv:1810.06758*, (2018).

[5] Hung Ba, 'Improving detection of credit card fraudulent transactions using generative adversarial networks', *arXiv preprint arXiv:1907.03355*, (2019).

[6] Barry G Becker, 'Using mineset for knowledge discovery', *IEEE Computer Graphics and Applications*, **17**(4), 75–78, (1997).

[7] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian Tharakunnel, and J Christopher Westland, 'Data mining for credit card fraud: A comparative study', *Decision Support Systems*, **50**(3), 602–613, (2011).

[8] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap, 'Dbsmote: density-based synthetic minority over-sampling technique', *Applied Intelligence*, **36**(3), 664–684, (2012).

[9] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer, 'Smote: synthetic minority over-sampling technique', *Journal of artificial intelligence research*, **16**, 321–357, (2002).

[10] Tianqi Chen and Carlos Guestrin, 'Xgboost: A scalable tree boosting system', in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, (2016).

[11] Andrea Dal Pozzolo, Olivier Caelen, Reid A Johnson, and Gianluca Bontempi, 'Calibrating probability with undersampling for unbalanced classification', in *2015 IEEE Symposium Series on Computational Intelligence*, pp. 159–166. IEEE, (2015).

[12] Pedro Domingos, 'Metacost: A general method for making classifiers cost-sensitive', in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 155–164, (1999).

[13] Georgios Douzas and Fernando Bacao, 'Effective data generation for imbalanced learning using conditional generative adversarial networks', *Expert Systems with applications*, **91**, 464–471, (2018).

[14] Ugo Fiore, Alfredo De Santis, Francesca Perla, Paolo Zanetti, and Francesco Palmieri, 'Using generative adversarial networks for improving classification effectiveness in credit card fraud detection', *Information Sciences*, **479**, 448–455, (2019).

[15] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera, 'A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches', *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, **42**(4), 463–484, (2011).

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, 'Generative adversarial nets', in *Advances in neural information processing systems*, pp. 2672–2680, (2014).

[17] Sarah J Graves, Gregory P Asner, Roberta E Martin, Christopher B Anderson, Matthew S Colgan, Leila Kalantari, and Stephanie A Bohlman, 'Tree species abundance predictions in a tropical agricultural landscape with a supervised classification model and imbalanced data', *Remote Sensing*, **8**(2), 161, (2016).

[18] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville, 'Improved training of wasserstein gans', in *Advances in neural information processing systems*, pp. 5767–5777, (2017).

[19] Xin Guo, Johnny Hong, Tianyi Lin, and Nan Yang, 'Relaxed wasserstein with applications to gans', *arXiv preprint arXiv:1705.07164*, (2017).

[20] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao, 'Borderline-smote: a new over-sampling method in imbalanced data sets learning', in *International conference on intelligent computing*, pp. 878–887. Springer, (2005).

[21] Haibo He and Edwardo A Garcia, 'Learning from imbalanced data', *IEEE Transactions on knowledge and data engineering*, **21**(9), 1263–1284, (2009).

[22] Nathalie Japkowicz and Shaju Stephen, 'The class imbalance problem: A systematic study', *Intelligent data analysis*, **6**(5), 429–449, (2002).

[23] David Jensen, 'Prospective assessment of ai technologies for fraud detection: A case study', in *AAAI Workshop on AI Approaches to Fraud Detection and Risk Management*, pp. 34–38, (1997).

[24] Hyun-Chul Kim, Shaoning Pang, Hong-Mo Je, Daijin Kim, and Sung Yang Bang, 'Constructing support vector machine ensemble', *Pattern recognition*, **36**(12), 2757–2767, (2003).

[25] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov, 'Siamese neural networks for one-shot image recognition', in *ICML deep learning workshop*, volume 2. Lille, (2015).

[26] Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera, 'An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics', *Information sciences*, **250**, 113–141, (2013).

[27] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, and Bernard Manderick, 'Credit card fraud detection using bayesian and neural networks', in *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*, pp. 261–270, (2002).

[28] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley, 'Least squares generative adversarial networks', in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2794–2802, (2017).

[29] Mehdi Mirza and Simon Osindero, 'Conditional generative adversarial nets', *arXiv preprint arXiv:1411.1784*, (2014).

[30] Xuetong Niu, Li Wang, and Xulei Yang, 'A comparison study of credit card fraud detection: Supervised versus unsupervised', *arXiv preprint arXiv:1904.10604*, (2019).

[31] Saharon Rosset, Uzi Murad, Einat Neumann, Yizhak Idan, and Gadi Pinkas, 'Discovery of fraud rules for telecommunications—challenges and solutions', in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 409–413, (1999).

[32] Florian Schroff, Dmitry Kalenichenko, and James Philbin, 'Facenet: A unified embedding for face recognition and clustering', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, (2015).

[33] Akhil Sethia, Raj Patel, and Purva Raut, 'Data augmentation using generative models for credit card fraud detection', in *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, pp. 1–6. IEEE, (2018).

[34] Hua Shao, Hong Zhao, and Gui-Ran Chang, 'Applying data mining to detect fraud behavior in customs declaration', in *Proceedings. International Conference on Machine Learning and Cybernetics*, volume 3, pp. 1241–1244. IEEE, (2002).

[35] Erik Sherman, 'Fighting web fraud.', *Newsweek*, **139**(23), 32B–32B, (2002).

[36] Kai Ming Ting, 'An instance-weighting method to induce cost-sensitive trees', *IEEE Transactions on Knowledge and Data Engineering*, **14**(3), 659–665, (2002).

[37] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatci, and Jason Yosinski, 'Metropolis-hastings generative adversarial networks', *arXiv preprint arXiv:1811.11357*, (2018).

[38] Wouter Verbeke, Karel Dejaeger, David Martens, Joon Hur, and Bart Baesens, 'New insights into churn prediction in the telecommunication sector: A profit driven data mining approach', *European Journal of Operational Research*, **218**(1), 211–229, (2012).

[39] Xing-Ming Zhao, Xin Li, Luonan Chen, and Kazuyuki Aihara, 'Protein classification with imbalanced data', *Proteins: Structure, function, and bioinformatics*, **70**(4), 1125–1132, (2008).

[40] Panpan Zheng, Shuhan Yuan, Xintao Wu, Jun Li, and Aidong Lu, 'One-class adversarial nets for fraud detection', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 1286–1293, (2019).