SMOTified-GAN for Class Imbalanced Pattern Classification Problems

ANURAGANAND SHARMA[ID]1, **PRABHAT KUMAR SINGH**[ID]2,
AND ROHITASH CHANDRA[ID]3, (Senior Member, IEEE)
1School of IT, Engineering, Mathematics & Physics (STEMP), The University of the South Pacific, Suva, Fiji
2Department of Mechanical Engineering, IIT (BHU) Varanasi, Varanasi 221005, India
3School of Mathematics and Statistics, University of New South Wales, Sydney, NSW 2052, Australia

Corresponding author: Anuraganand Sharma (sharma_au@usp.ac.fj)

ABSTRACT Class imbalance in a dataset is a major problem for classifiers that results in poor prediction with a high true positive rate (TPR) but a low true negative rate (TNR) for a majority positive training dataset. Generally, the pre-processing technique of oversampling of minority class(es) are used to overcome this deficiency. Our focus is on using the hybridization of Generative Adversarial Network (GAN) and Synthetic Minority Over-Sampling Technique (SMOTE) to address class imbalanced problems. We propose a novel two-phase oversampling approach involving knowledge transfer that has the synergy of SMOTE and GAN. The unrealistic or overgeneralized samples of SMOTE are transformed into realistic distribution of data by GAN where there is not enough minority class data available for GAN to process them by itself effectively. We named it SMOTified-GAN as GAN works on pre-sampled minority data produced by SMOTE rather than randomly generating the samples itself. The experimental results prove the sample quality of minority class(es) has been improved in a variety of tested benchmark datasets. Its performance is improved by up to 9% from the next best algorithm tested on F1-score measurements. Its time complexity is also reasonable which is around $O(N^2 d^2 T)$ for a sequential algorithm.

INDEX TERMS Generative adversarial network (GAN), synthetic minority over-sampling technique (SMOTE), SMOTified-GAN, class imbalance problem.

I. INTRODUCTION

Class imbalance problem (CIP) refers to a type of classification problems where some classes are either majorly or moderately underrepresented in comparison to other classes [1]. The unequal distribution makes many conventional machine learning algorithms quite less effective, especially for the prediction of minority classes [2]. A number of solutions have been proposed at the data and algorithm levels to deal with class imbalance such as preprocessing for oversampling or under-sampling, data augmentation, cost-sensitive learning/model penalization and one-class classification [1], [3]–[5].

The imbalance dataset exhibits a major problem for the classifiers to be bias towards the majority class. The imbalanced class distribution results in the degradation of performance of the classifier model due to biased classification towards the majority class. It causes high true positive

The associate editor coordinating the review of this manuscript and approving it for publication was Eunil Park[ID].

rate (TPR) and a low true negative rate (TNR) when majority samples are positive [6]. Data imbalance can be commonly seen in fraud/fault/anomaly detection [3], [7]–[11], medical diagnosis of lethal and rare diseases [5], [12], [13], software defect prediction [14], natural disaster etc [4].

Commonly used pre-processing technique is oversampling as undersampling removes important information and does not result in accurate classification [15]. Oversampling too suffers from inclusion of illegitimate samples which is still an active area of research [16], [17]. Synthetic oversampling technique (SMOTE) [18] is considered a "de facto" standard for an oversampling method. It is simple and effective; however, it may not produce diverse sample. SMOTE uses interpolation to randomly generate new samples from the nearest neighborhood of minority class data. It has been successfully used in regression [19], and classification problems [20] for a wide range of models [21]. A review of SMOTE and applications has been given in [22].

The data samples in the case of imbalanced dataset can also be generated through classification models as well with

data augmentation approach. Generative Adversarial Network (GAN) and its variations are commonly used to generate new "fake" samples [23]–[26]. GAN was originally designed to generate the realistic-looking images for large datasets, however, it can also generate minority class samples thereby balancing the class distribution and avoiding over-fitting effectively [27]. Imbalanced data classification is ubiquitous in application domains. Data augmentation technique based on variations of GAN have been successfully applied on many applications such as skin lesion classification [13] for better diagnosis or pipeline leakage in petrochemical system [28].

Bayesian inference provides a principled framework to estimate unknown quantity represented by the posterior distribution (parameters of a model) which is updated via Bayes' theorem as more information gets available [29]–[31]. Markov Chain Monte Carlo (MCMC) sampling is typically used to implement Bayesian inference [32]. It features a likelihood function that takes into account the prior distribution to either accept/reject samples obtained from a proposal distribution to construct the posterior distribution of model parameters, such as weights of a neural network [31]–[34]. A major limitation for MCMC sampling technique is high computational complexity for sampling from the posterior distribution [35], [36]. There recently there has been much progress in MCMC sampling via the use of gradient-based proposals and parallel computing in Bayesian deep learning [37]–[39]. However, these have been mostly limited to model parameter (weights) uncertainty quantification rather than quantifying uncertainties in data or addressing class imbalanced problems. In the case of class imbalanced problems, MCMC sampling has been used for benchmark real-world imbalanced datasets [35], [40]. MCMC method have been applied for handing imbalanced categorical data [36]. Das *et al.* in [35] have used Gibbs sampling (an MCMC method) to generate new minority class samples.

Another example of oversampling method is data dependant cost matrix, where a weighted misclassification cost is assigned to the misclassified classes [4]. It is not easy to determine this cost [35]. The cost-sensitive loss function has penalty based weights for misclassification errors from both majority and minority classes. Hybrid neural network with a cost-sensitive support vector machine (hybrid NN-CSSVM) in [41] considers different cost related to each misclassification. Castro *et al.* in [2] have improved the misclassification error for the imbalanced data by using the cost parameter according to the ratio of majority samples in the training set. One-class problem [8], [42], [43] also has a "minority" class but generally it is considered outlier which is removed from the training data. One-class modeling usually uses feature mapping or feature fitting to enforce the feature learning process [43].

In this paper, we propose a novel hybrid approach that combines the strengths and overcomes the deficiency of two independent models that include SMOTE and GAN. SMOTE is known to produce some irregular or "out of distribution"

samples. Additionally, SMOTE has not been generally used with deep learning and GAN generally has not been used for small datasets (minority classes) [11]. Hence, we refer to our approach to it as SMOTified-GAN which is a two-phased process based on knowledge transfer or transfer learning [44]. Firstly, SMOTE generates promising samples which is then "transferred" to GAN which no longer uses random sample. Our approach may work well for both small and large datasets. This could lead to more feasible and diverse data which are further enhanced through GAN to prepare better quality samples. We have obtained impressive results for our proposed method on numerical benchmark CIP datasets mainly from UCI library [45]. Its efficiency is also reasonable which is the combination of SMOTE and GAN as discussed in Section 2 and Section 3. SMOTified-GAN, however, works on non-image data only in its current form.

The rest of the paper is organised as follows. Section 2 presents the state-of-the-art techniques to solve CIPs. Section 3 discusses the proposed method – SMOTified-GAN. Section 4 shows the experimental results and Section 5 discusses the outcome of the experiments. Lastly, Section 6 concludes the paper by summarizing the results and proposing some further extensions to the research.

II. RELATED WORK ON CLASS IMBALANCE PROBLEMS
A. SYNTHETIC MINORITY OVERSAMPLING TECHNIQUE (SMOTE)

The SMOTE is a "de facto" standard for pre-processing imbalanced data. This is not a complete random sampling whereas it uses interpolation among the neighboring minority class examples. It is efficient and easy to implement. Each minority example gets k-nearest neighbors (KNN) which are randomly selected to have interpolation to create new samples. The pseudocode is given in Algorithm 1. The parameters n and d are the size and dimension of the minority class respectively; N is the size of the majority class and parameter k for k-nearest neighbor. Lines 1-5 finds KNN for each minority sample then does the interpolation with them to create new samples. Lines 6-12 describes the interpolation step where $N - n$ samples are being created and added into minority class. Its time complexity for a single machine has the order of $O((N - n)dn\log k) \approx O(N^2 d\log k)$ [18], [22], [46].

There are many variants of SMOTE that have been successfully applied to various application domains such as bioinformatics, video surveillance, fault detection or high dimensional gene expression data sets [22], [47], [48]. There are many variants of SMOTE such as regular SMOTE, Borderline-SMOTE, SVM-SMOTE and KMeans-SMOTE [49]. Kovacs in [50] has shown the implementation of 85 variants of SMOTE in python library.

B. GENERATIVE ADVERSARIAL NETWORK (GAN)

GAN is a class of machine learning frameworks in which there is a contest between two neural networks with a continuous and simultaneous improvement of both neural networks.

Algorithm 1: Pseudocode for SMOTE

```
// Input: d-dimensional minority
   samples X of size n from a
   training data set of size N that
   requires N − n over-samples. k
   defines k-nearest neighbors.
```
1   $N \leftarrow N - n$
2   **for** $i = 1 : \|X\|$ **do**
3     $S \leftarrow \text{KNN}(x_i, k)$ // $x_i \in X$
4     $X \leftarrow \text{interpolate}(N/100, x_i, S)$ // for $N > 100$
5   **end**

```
// subroutine for interpolate
```
6   interpolate (N, x_i, S)
7   **while** $\|X\| < N$ **do**
8     $a \leftarrow R_I^{k \times 1}(1)$ // pick a random integer value from 1...k
9     $x_j \leftarrow S\{a\}$
```
   // Δx_ij = x_j − x_i ⇒ euclidean distance
      between x_i and x_j
   // R_D^{1×d} ⇒ a decimal random number
      between 0 to 1
```
10     $X \leftarrow X \bigcup (x_i + \Delta x_{ij} \times R_D^{1 \times d})$
11   **end**
12   **return** X

This technique learns to generate new data with the same statistics as the training set by capturing the true data distribution [51].

GAN has been successfully used for data augmentation. The two neural networks of GAN learn the target distribution and generate new samples to achieve similar distributive structure in its generated over-sampled data. A GAN is simply the synergy of two deep learning network that produce "fake" data examples emulating the properties of the real data [27], [52], [53].

GAN had not been designed for oversampling imbalanced classes but to create "fake" images of real images which should be hard to distinguish. However, its success in data augmentation for over-sampling has led to the introduction to many variations of GAN to solve CIP [24]–[26], [51], [54]–[56].

The first network is called Generator whose responsibility is to takes a vector of random values and generate the data similar to the real data used in training. The second network is called Discriminator that takes input data from both the real training data and the "fake" data from the generator, to classify them correctly. This process is shown in FIGURE 1.

The time-complexity of GAN can be roughly given as $O(nTLd^2)$ where the new parameters L and T are layer-size and total iterations for a GAN. Its convergence rate with the Stochastic Gradient Descent would be $O(\frac{1}{T} + \sigma^2)$ where σ^2 is the variance of the dataset [57]. See Section III for further details.
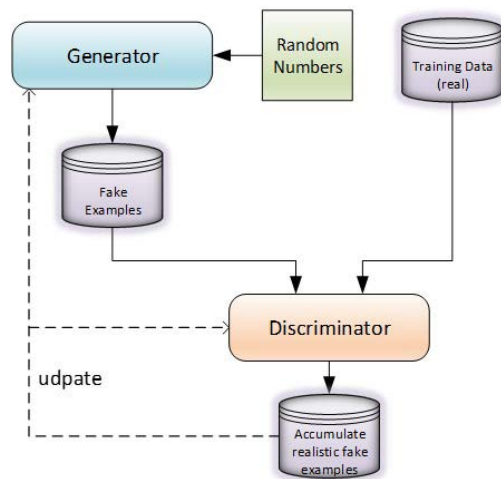

FIGURE 1. Process of "fake" sample generation with GAN.

III. SMOTified-GAN FOR CLASS IMBALANCE PROBLEM

Our proposed method tries to overcome the deficiency of both SMOTE and GAN by using a transfer learning concept where it first extracts the knowledge about minority class from SMOTE and then applies it to GAN. We have named it SMOTified-GAN as it tries to diversify the original samples produced by SMOTE through GAN. Additionally, the quality of the sample is further enhanced by emulating them with the realistic samples. The process of SMOTified-GAN is shown in FIGURE 3.

Even though SMOTE is widely used as an oversampling technique, it suffers with some deficiency. The major drawback of SMOTE is that it focuses on local information and therefore it does not generate diverse set of data as shown in FIGURE 2(a). Additionally, FIGURE 2(b) shows the 5 nearest neighbors of x_1, $\{x_2, \ldots, x_6\}$ are firstly, blindly chosen then interpolated (using Euclidean distance) to get the corresponding synthetic samples $\{a, \ldots, e\}$. Even, there there is a high chance of miss-classification for sample e with a majority sample y_1 [58]. The generated data are generally insufficiently realistic compared to GAN that captures the true data distribution in order to generate data for the minority class [25].
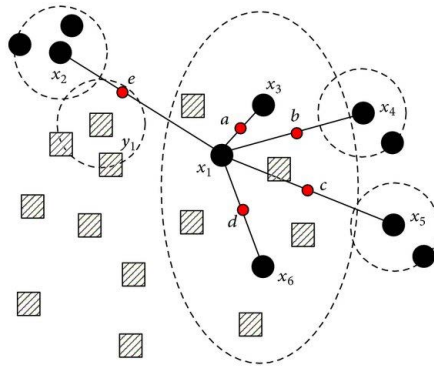
GAN is not ideally fit for oversampling as it has been originally designed for realistic looking images with convolutional neural networks (CNN) rather than producing over-samples for the minority class. Additionally, GAN may face data scarcity problem as minority class is already in reduced form where model training requires more of its data to be sacrificed for validation and testing purpose. Though, cross-validation techniques may solve this problem to some extent.

The architecture of GAN consists of two networks as mentioned in the previous section where the objective of the generator network is to generate data that fools the discriminator network to classifies as "real". To optimize its performance, maximize the loss of the discriminator when data is coming from the generator. The objective of the generator is to generate data that the discriminator classifies as "real".

Synthetic Minority Oversampling Technique

Original Dataset   Generating Samples   Resampled Dataset
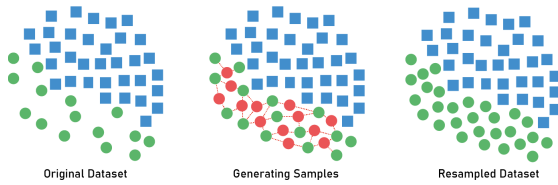
(a) Low-diversity with SMOTE taken from [62]

☒ Majority class samples
● Minority class samples
● Synthetic samples

(b) Interpolation with SMOTE taken from [60]

FIGURE 2. SMOTE processing for oversampling.

To optimize the performance of the discriminator, the loss of the discriminator is to be minimized when given batches of both real and generated data. The objective of the discriminator is to not be "fooled" by the generator [51], [60].

The discriminator score can be given as:

$$\max_{D} \mathbb{E}_x[\log D(x)] + \mathbb{E}_z[\log(1 - D(G(z)))]$$

or

$$\min_{D} \mathbb{E}_x[-\log D(x)] - \mathbb{E}_z[\log(1 - D(G(z)))]$$

$D(x)$ contains the discriminator output probabilities for the real data x and $D(G(z))$ contains the discriminator output probabilities for the generated data z.

The generator score is:

$$\min_{G} -\mathbb{E}_z[\log D(G(z))]$$

The pseudocode for the GAN algorithm is given in Algorithm 2 where *SGD* and *weights* are functions to determine gradient for a mini-batch using Stochastic Gradient Descent algorithm (SGD) optimizer [61] or its any other variation such as ADAM [62] or RMSprop [63], and update the weights respectively. Once the algorithm terminates 'good' fake samples are collected with *accumulateFakeEx* based on classification accuracy.

Algorithm 2: Pseudocode for GAN

```
// Input: training data set examples x
   and noise samples z from
   appropriate random number
   generator. An optional parameter
   can be the size n_fake of fake sample
   needed.
// initialize parameters
// m_i is minibatch indices for i^th
   index and T is total iterations.
```
1   GAN (x, z, n_{fake})
2   **for** $t = 1 : T$ **do**
```
      // generally step size S is 1
      // subscript d and g refers to
         discriminator and generator
         entity respectively
```
3   **for** $s = 1 : S$ **do**
4      $g_d \leftarrow$
      $SGD(-\log D(x) - \log(1 - D(G(z)), W_d, m_i)$
5      $W_d \leftarrow weights(g_d, W_d)$
6   **end**
7    $g_g \leftarrow SGD(-\log D(G(z)), W_g, m_i)$
8    $W_g \leftarrow weights(g_g, W_g)$
9   **end**
10   $x' \leftarrow$ accumulateFakeEx $(Model_d(W_d, x, z),$
    $Model_g(W_g, x, z), n_{fake})$
11   **return** x'

Goodfellow [64] has used *sigmoid* as the activation function that would result the following scores to minimize:
Discriminator:

$$\mathbb{E}_x[-\log(1 + e^{-y})] - \mathbb{E}_z[1 - \log(1 + e^{-\hat{y}})]$$

Generator:

$$\mathbb{E}_z[-\log(1 + e^{-\hat{y}})]$$

where y and $\hat{y}$ are the outputs of the Discriminator D and Generator model G respectively before the activation function is applied.

The formalization of SMOTified-GAN is not very different from the original GAN. Only the random generator function of GAN is replaced with the repertoire of oversample minority examples from SMOTE. The modified scores can be shown as:
discriminator score:

$$\max_{D} \mathbb{E}_{x^*}[\log D(x^*|x)] + \mathbb{E}_u[\log(1 - D(G(u)))]$$

Generator score:

$$\min_{G} -\mathbb{E}_u[\log D(G(u))]$$

where x^* is training samples of minority class(es) and u is over-sampled data of the same class(es) generated from

different algorithms such as SMOTE in this case. The pseudocode for SMOTified-GAN is given in Algorithm 3. Its implementation is not too difficult either. The Python code is available at https://github.com/anuraganands.

As illustrated in FIGURE 3, there are two sections of SMOTified-GAN. The first one replaces the random number generator (refer FIGURE 1) with the repertoire of oversamples from SMOTE. The second section continues with the process of GAN using the new samples from SMOTE. Algorithm 3 also shows this process in two steps. Line (1) calls SMOTE function given in Algorithm 1 and then Line (2) calls GAN function given in Algorithm 2. However, this time the generated samples u is used instead of random noise z.

Algorithm 3: Pseudocode for SMOTified-GAN

```
// Input: minority examples x* from a
   training data set x of size N that
   requires N − n over-samples;
// User-defined parameter k for
   k-nearest neighbors.

// First execute SMOTE given in
   Algorithm 1 then GAN given in
   Algorithm 2
1 u ← call Algorithm_1 (x*, k) // generate
   over-sampled minority examples u.
2 u ← call Algorithm_2 (x*, u, N − n).
```

Its time complexity for sequential algorithm is combination of SMOTE's and GAN's time complexity, i.e., $O(N^2 d \log k + nTLd^2) \approx O(N^2 d + TNd^2)$ Since n is a small part of N so it can be assumed nL is comparable to N. This can further simplify the complexity to $O(N^2 d + TNd^2) \leq O(N^2 d^2 (1/d + T/N)) \leq O(N^2 d^2 T)$.

The major difference between our proposed method SMOTEfied-GAN and GAN is the use of ready-made repertoire of samples generated from SMOTE instead of a set of random noise to begin with. Intuitively, this helps in improvement of the input samples that produces better over-samples. This natural synergy of SMOTE and GAN guides the naïve GAN to have a jump-start with promising data before going through further refinement of unrealistic data from SMOTE.

IV. EXPERIMENTS AND RESULTS

In this section, we provide experimental results of over-sampling methods, namely, SMOTE, GAN and SMOTified-GAN on different datasets that have been taken from the literature of CIP [65]–[67]. The over-sampled minority class data have been made equal to the majority class data which is then augmented into training data that are then fed into the Neural Networks (NN) for classification. We have also done the testing on original datasets without using any data augmentation technique.

A. DATASETS

We evaluate and compare our model on small to large datasets that feature class imbalance as shown in TABLE 1. The datasets were mainly obtained from the UCI machine learning repository [45] that have been used in a number of methods for CIPs [65]–[67]. Some datasets such as Credit-Card Fraud and Shuttle are highly imbalanced with minority class contribution as 0.17% and 0.29% respectively.

B. EXPERIMENTAL SETUP

We used naïve GAN model [51] and naïve SMOTE [18] in this paper. SMOTified-GAN uses the above two models, however, it is flexible enough to work with other combinations of different variations as well. The parameter settings such as learning rate, total epochs and loss functions are shown in TABLE 2. The GAN generator neural network features 3 hidden layers with 128 neurons in each layer. The GAN discriminator network is similar to the generator network with major difference of having only two layers first a linear layer followed by a leaky-ReLu layer with alpha = 0.2. The classifier architecture is given in FIGURE 5. In GAN training, we use binary cross-entropy activation function with training data batch-size of 128 and initial learning-rate of 0.00001 with Adam optimser.

After basic pre-processing steps, SMOTE oversampling is done with $k = 5$ neighbors. The stopping criteria for SMOTified-GAN and naïve GAN's training are based on validation error to avoid any over-learn. Additionally, it is ensured that the discriminator and generator loss remain significant and do not approach near zero.

C. PRELIMINARY INVESTIGATION

The experiment has been conducted on 11 benchmark imbalanced datasets that are trained on NN to test the efficacy of various oversampling techniques. We used SMOTE, GAN and our proposed method SMOTified-GAN for oversampling. We have also done the testing with original data without any data augmentation. The quality of classification and comparative results are shown in TABLE 3. As expected all datasets show high train and test accuracy due to high imbalance in the datasets. So it is important to look into F1 scores to determine high precision and recall measures. The best F1 scores have been shown with the bold font.

It is clear from the experimental results that SMOTified-GAN has outperformed other oversampling techniques. Only Connect4 is an outlier where all oversampling techniques are showing poor results compared to the non-oversampling technique. Surprisingly, SMOTE also performed poorly by 3.6% compared to the original training dataset without any data augmentation. This result can be attributed to the fact that the dataset is highly imbalanced where minority class constitutes only 3.84% of the training dataset. This does not provide enough data for generalization. So the minority class should not be over-sampled blindly for a given dataset. Conversely, no data augmentation with datasets such Ecoli
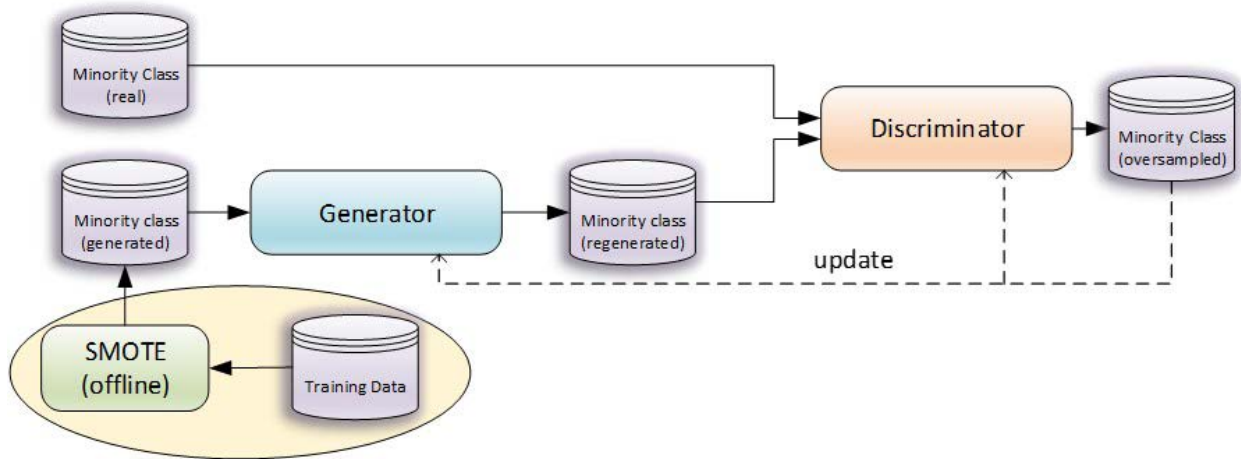
FIGURE 3. Process of sample generation with SMOTified-GAN.

TABLE 1. Description for small to large datasets.

Dataset	Features	Classes	Instances	Minority Class (%)	Description
Ecoli [45]	7	2	335	5.97	This data contains protein localization sites
Ionosphere [45]	34	2	351	35.71	Classification of radar returns from the ionosphere
Pageblocks [45]	10	2	471	5.94	Classifying all the blocks of the page layout of a document
Yeast [45]	8	2	513	9.94	Predicting the Cellular Localization Sites of Proteins
Wine [45]	11	2	655	2.74	Using chemical analysis determine the origin of wines
Poker [45]	10	2	1476	1.15	Purpose is to predict poker hands
Abalone [45]	8	2	4177	20.1	Predict the age of abalone from physical measurements
Spambase [45]	57	2	4601	39.39	Classifying Email as Spam or Non-Spam
Shuttle [45]	9	2	58000	0.294	Approximately 80 percent of the data belongs to class 1
Credit-Card Fraud [68]	30	2	284807	0.172	This dataset has 492 frauds out of 284,807 transactions.
Connect4 [45]	42	2	376640	3.84	Contains connect-4 positions

TABLE 2. Parameter settings.

Parameter	Neural Network	Generator	Discriminator
Total neurons per hidden layer:	256, 128	128, 256, 512, 1024	512, 256, 128
Optimizer :	Adam	Adam	Adam
Loss Function :	Mean Absolute Error	BCEWithLogitsLoss	BCEWithLogitsLoss
Activation :	ReLU	ReLU	LeakyReLU (0.2)
Normalization :	-	BatchNorm1d	-
Learning Rate :	0.00001	0.00001	0.00001

(6.0% minority class) and Wine (2.7% minority class) shows very poor and unacceptable results. Here data augmentation techniques especially with SMOTified-GAN show much better results of F1-score of 92.2% and 52.7% respectively.

SMOTified-GAN gives the best results – considering F1 score – for all other datasets with the diverse proportion of minority class such as Creditcard Fraud (0.2% minority class), Spambase (39.4% minority class), Yeast (9.9% minority class) and Wine (2.7% minority class). The rest of the datasets, Ionosphere, Shuttle, Ecoli, Pageblocks and Poker also favors SMOTified-GAN. FIGURE 4 on the comparative F1 score shows SMOTified-GAN outperforms other algorithms on 10/11 datasets. Its performance is significantly improved for Pageblocks by 9% and 10% for Ecoli. SMOTified-GAN has also produced better precision and

recall for most of the datasets. It has the best precision for all 11 datasets and the best recall for 9/11 datasets. GAN and SMOTE give mixed results on different datasets. GAN has produced 10/11 times better results than SMOTE. Notably, data augmentation less training is also better than GAN and SMOTE with 2/11 times and 4/11 times respectively.

Datasets like Yeast, Ecoli, Wine, Poker and Pageblocks have small number of minority class data instances relative to the majority class which allows SMOTified-GAN to show its potential over other algorithms as seen in the respective results. In datasets like Ecoli and Wine the minority instances are so low that the non-oversampling method completely fails to predict the minority class. All models give high train and test accuracy on all datasets which is attributed to the dominance of majority class in these datasets hence the true
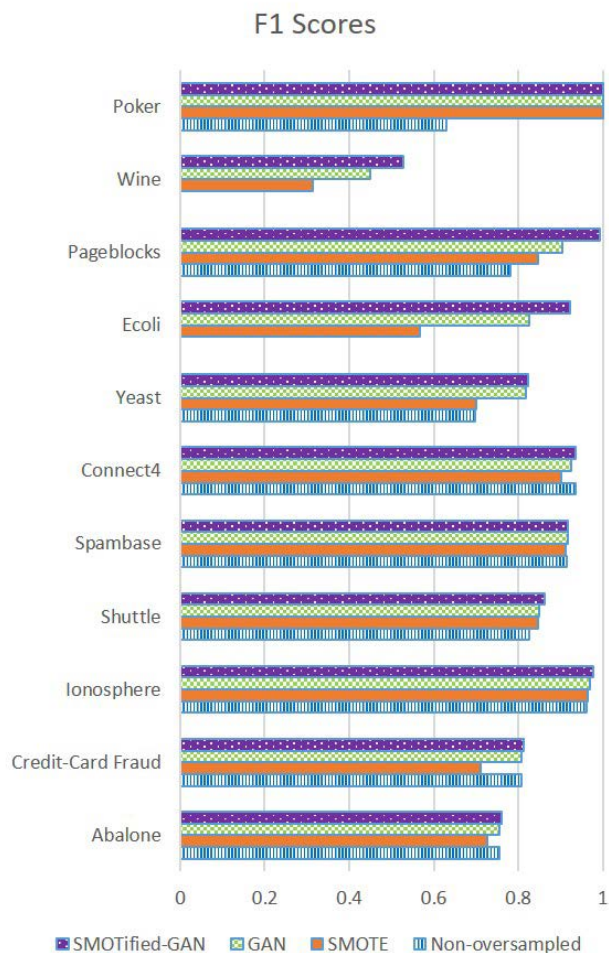
F1 Scores

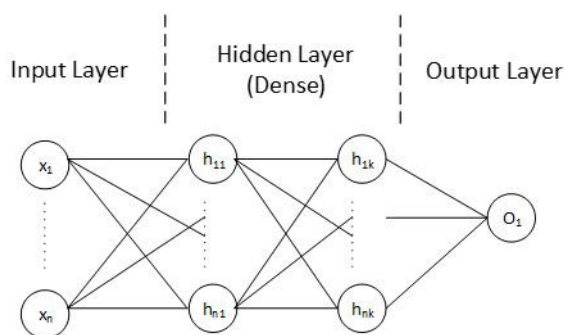**FIGURE 4.** Comparison of F1 scores.

FIGURE 5. Architecture of the classifier.

performance index measure is the minority F1-score which depends on both the precision and recall. Overall, the proposed model of SMOTified-GAN outperforms the others in terms of F1-score and a comparatively low standard deviation of results.

The training loss curves of SMOTified-GAN's generator and discriminator models w.r.t the number of epochs during training of selected datasets have been shown in FIGURE 7. In general, the Discriminator's loss curve converges fairly

quickly whereas the Generative loss curve demonstrates high fluctuations, however, these fluctuations generally gets steady at around 2000 epochs. We have also drawn validation F1-score in the same graph to determine the termination criterion. The training stops once the validation F1-score reaches its highest value to avoid any over-fitting.

D. RESULTS

Table 3 presents a summary for the experimental results with NN using the respective oversampling methods – SMOTE, GAN, SMOTified-GAN and also without no augmentation. It shows training and test accuracy and measurements of F1-score, precision and recall. Its purpose is to demonstrate the effectiveness of the methods for class imbalanced datasets. We report the mean, standard deviation, and best performance using the respective evaluation metrics using 30 experimental runs where each run has a different randomised initial position in weight space. This is done to incorporate model uncertainty in our results.

FIGURE 6 presents the receiver operating characteristic curve or ROC curve on precision and recall for the tested datasets. The results for nine datasets have been illustrated with all the tested algorithms. It also shows the measure for the area under the curve (AUC). This is a standard performance measure for imbalanced data. It is clear from the graph that our proposed SMOTified-GAN has highest AUC for all the datasets except Abalone. For example FIGURE 6(h) shows AUC for Shuttle dataset with SMOTified-GAN, GAN, no data augmentation and SMOTE have the result of 0.949, 0.911, 0.891 and 0.712 respectively in descending order. SMOTified-GAN is better than others by up to 0.038 to the next best algorithm. GAN and SMOTE shows the mixed results as discussed earlier with F1-scores.

V. DISCUSSION

A significant improvement in the quality of classification has been observed with the introduction of SMOTified-GAN as an oversampling technique. It has clearly outperformed naïve GAN and SMOTE in most of the datasets. The F1-score has been improved by up to 9% for Ecoli dataset from the next best oversampling technique where the precision has also shown significant growth of around {7% to 8%} for Wine and Yeast datasets. The recall has not been much improved. SMOTE generally has low precision and SMOTified-GAN has relatively better precision than the other models. Most notable improvement from GAN and SMOTE can be seen with {Abalone, Pageblock, Wine, and Shuttle datasets}, and {Credit-card Fraud and Wine datasets} respectively.

Considering the impact of the number of features, percentage of minority class and the size of the data on the quality of oversampling has shown mixed results. For example datasets with higher features such as Credit-card fraud (30 features, 0.172% of minority class) have shown good results but Connect4 (42 features, 3.84% of minority class) has shown poorer results. However, SMOTE generally performs poorly on F1-score even with raw data in 3/4 times when the

(a) abalone-1   (b) creditcard   (c) ecoli

(d) ionosphere   (e) connect4   (f) pageblocks

(g) poker   (h) shuttle   (i) spambase
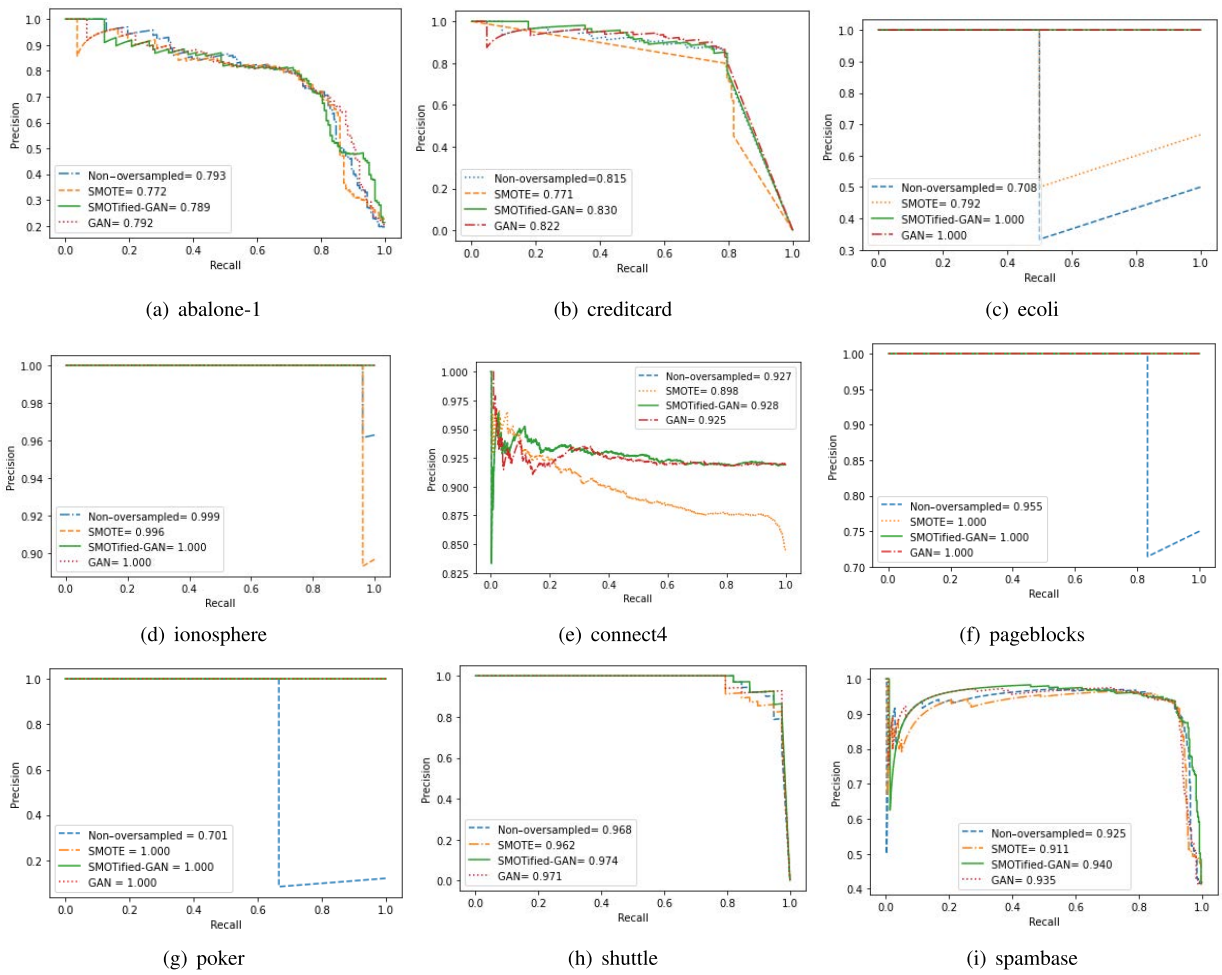
FIGURE 6. Precision vs Recall for AUC Curves.

(a) abalone-1   (b) creditcard   (c) ecoli

(d) ionosphere   (e) pageblocks   (f) poker
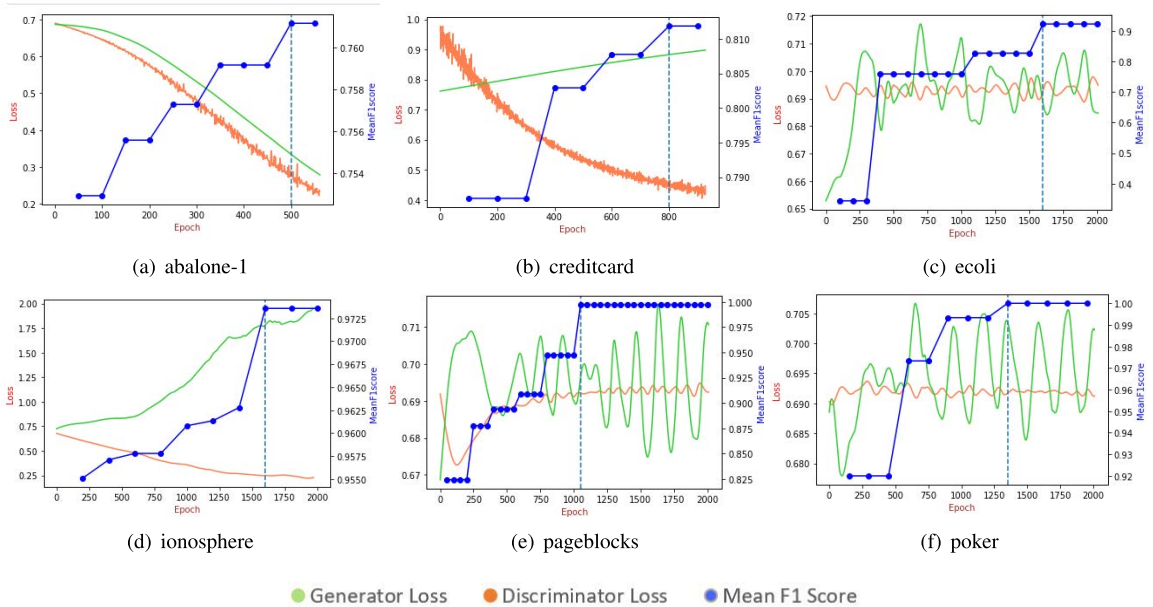
● Generator Loss   ● Discriminator Loss   ● Mean F1 Score

FIGURE 7. SMOTified-GAN's loss and epoch for the best F1-score.

TABLE 3. Comparison of experimental results on NN with baseline methods (SMOTE, GAN, SMOTified-GAN and non-oversampled original data).

Dataset	Oversampling methods	Train Mean (Best, SD)	Test Mean (Best, SD)	F1 Mean (Best, SD)	Precision	Recall
Ecoli	Non-oversampled	0.9328 (0.9328,0.0000)	0.9701 (0.9701,0.0000)	0.0000 (0.0000,0.0000)	0.00	0.00
	SMOTE	0.9905 (0.9959,0.0020)	0.9577 (0.9701,0.0100)	0.5684 (0.6666,0.0890)	0.50	1.00
	GAN	0.9885 (0.9919,0.0013)	0.9880 (1.0000,0.0099)	0.8266 (1.0000,0.1964)	1.00	1.00
	SMOTified-GAN	0.9861 (0.9879,0.0010)	0.9960 (1.0000,0.0077)	**0.9222 (1.0000,0.1433)**	1.00	1.00
Ionosphere	Non-oversampled	0.9878 (0.9928,0.0027)	0.9728 (0.9857,0.0126)	0.9621 (0.9803,0.0179)	0.98	0.98
	SMOTE	0.9914 (0.9916,0.0007)	0.9738 (0.9857,0.0113)	0.9632 (0.9803,0.0164)	0.97	0.99
	GAN	0.9901 (0.9944,0.0017)	0.9767 (1.0000,0.0086)	0.9701 (1.0000,0.0210)	1.00	1.00
	SMOTified-GAN	0.9903 (0.9944,0.0023)	0.9823 (1.0000,0.0068)	**0.9777 (1.0000,0.0169)**	1.00	1.00
Pageblocks	Non-oversampled	0.9627 (0.9627,0.0000)	0.9775 (0.9894,0.0050)	0.7803 (0.9090,0.0700)	1.00	0.82
	SMOTE	0.9955 (1.0000,0.0030)	0.9761 (1.0000,0.0130)	0.8480 (1.0000,0.0780)	1.00	1.00
	GAN	0.9943 (0.9972,0.0020)	0.9858 (1.0000,0.0098)	0.9038 (1.0000,0.0793)	1.00	1.00
	SMOTified-GAN	0.9943 (1.0000,0.0043)	0.9989 (1.0000,0.0042)	**0.9926 (1.0000,0.0291)**	1.00	1.00
Yeast	Non-oversampled	0.9748 (0.9780,0.0013)	0.9323 (0.9417,0.0097)	0.6987 (0.7272,0.0305)	0.80	0.67
	SMOTE	0.9638 (0.9757,0.0083)	0.9139 (0.9417,0.0152)	0.7012 (0.7857,0.0446)	0.82	0.75
	GAN	0.9782 (0.9811,0.0028)	0.9595 (0.9514,0.0036)	0.8173 (0.8333,0.0169)	0.83	0.83
	SMOTified-GAN	0.9663 (0.9703,0.0023)	0.9611 (0.9611,0.0044)	**0.8221 (0.8695,0.0223)**	**0.91**	**0.83**
Wine	Non-oversampled	0.9770 (0.9770,0.0000)	0.9541 (0.9541,0.0000)	0.0000 (0.0000,0.0000)	0.00	0.00
	SMOTE	0.9806 (0.9843,0.0020)	0.9081 (0.9389,0.0090)	0.3149 (0.5000,0.0651)	0.39	0.67
	GAN	0.9841 (0.9873,0.0020)	0.9549 (0.9618,0.0073)	0.4489 (0.5454,0.1112)	0.46	0.67
	SMOTified-GAN	0.9854 (0.9873,0.0010)	0.9558 (0.9694,0.0090)	**0.5274 (0.6000,0.0780)**	**0.53**	**0.69**
Poker	Non-oversampled	0.9902 (0.9906,0.0008)	0.9949 (0.9966,0.0020)	0.6300 (0.8000,0.2530)	1.00	0.67
	SMOTE	1.0000 (1.0000,0.0000)	1.0000 (1.0000,0.0000)	**1.0000 (1.0000,0.0000)**	1.00	1.00
	GAN	1.0000 (1.0000,0.0000)	1.0000 (1.0000,0.0000)	**1.0000 (1.0000,0.0000)**	1.00	1.00
	SMOTified-GAN	1.0000 (1.0000,0.0000)	1.0000 (1.0000,0.0000)	**1.0000 (1.0000,0.0000)**	1.00	1.00
Abalone	Non-oversampled	0.9080 (0.9108,0.0019)	0.9072 (0.9114,0.0028)	0.7556 (0.7658,0.0090)	0.80	0.73
	SMOTE	0.8969 (0.9022,0.0035)	0.8622 (0.8827,0.0105)	0.7259 (0.7566,0.0200)	0.78	0.72
	GAN	0.9422 (0.9439,0.0013)	0.9070 (0.9125,0.0040)	0.7555 (0.7687,0.0061)	0.80	0.74
	SMOTified-GAN	0.9427 (0.9441,0.0008)	0.9075 (0.9126,0.0036)	**0.7612 (0.7711,0.0065)**	0.80	**0.75**
Spambase	Non-oversampled	0.9476 (0.9527,0.0020)	0.9309 (0.9380,0.0027)	0.9152 (0.9213,0.0032)	0.91	0.92
	SMOTE	0.9455 (0.9526,0.0026)	0.9276 (0.9336,0.0031)	0.9129 (0.9204,0.0049)	0.92	0.92
	GAN	0.9571 (0.9599,0.0019)	0.9319 (0.9380,0.0030)	0.9172 (0.9222,0.0036)	0.93	0.92
	SMOTified-GAN	0.9583 (0.9602,0.0012)	0.9323 (0.9380,0.0026)	**0.9174 (0.9222,0.0031)**	**0.94**	0.91
Shuttle	Non-oversampled	0.9994 (0.9998,0.0006)	0.9992 (0.9996,0.0005)	0.8256 (0.9350,0.2294)	0.92	0.96
	SMOTE	0.9996 (0.9996,0.0000)	0.9990 (0.9993,0.0001)	0.8465 (0.8837,0.0220)	0.83	**0.97**
	GAN	0.9995 (0.9999,0.0005)	0.9989 (0.9996,0.0009)	0.8497 (0.9367,0.2240)	0.93	0.95
	SMOTified-GAN	0.9996 (0.9997,0.0004)	0.9993 (0.9996,0.0006)	**0.8632 (0.9368,0.2009)**	0.93	0.95
Credit-Card Fraud	Non-oversampled	0.9996 (0.9997,0.0001)	0.9991 (0.9993,0.0001)	0.8066 (0.8214,0.0327)	0.84	0.80
	SMOTE	0.9996 (0.9997,0.0001)	0.9990 (0.9991,0.0001)	0.7099 (0.7409,0.0210)	0.80	0.69
	GAN	0.9995 (0.9997,0.0001)	0.9991 (0.9994,0.0001)	0.8069 (0.8214,0.0241)	0.84	0.80
	SMOTified-GAN	0.9993 (0.9994,0.0001)	0.9992 (0.9993,0.0001)	**0.8118 (0.8243,0.0202)**	**0.85**	0.80
Connect4	Non-oversampled	0.9947 (0.9966,0.0014)	0.9948 (0.9965,0.0013)	**0.9361 (0.9578,0.0151)**	0.92	1.00
	SMOTE	0.9967 (0.9970,0.0001)	0.9912 (0.9930,0.0007)	0.9011 (0.9167,0.0068)	0.85	1.00
	GAN	0.9962 (0.9982,0.0010)	0.9938 (0.9965,0.0021)	0.9251 (0.9577,0.0180)	0.92	1.00
	SMOTified-GAN	0.9966 (0.9986,0.0009)	0.9946 (0.9965,0.0017)	0.9355 (0.9578,0.0158)	0.92	1.00

features are high as in Connect4 (42 features), Credit-card fraud (30 features), Ionosphere (34 features) and Spambase (57 features).

Furthermore, the best algorithm may not be clearly visible with ROC curves in FIGURE 6, however, the AUC-ROC measures for each graph shows that SMOTified-GAN outperforms other algorithms. The larger area the ROC curve occupies the better the algorithm which is shown by the AUC measures. For example, it is somewhat clear from the Shuttle that shows the best to worst in the order of SMOTified-GAN (0.949), GAN (0.911), no oversampling technique (0.891) and then SMOTE (0.712). So SMOTified-GAN is 3.5% better than the next best algorithm. Similarly, it is 2.1% better than the second best algorithm for Spambase.

VI. CONCLUSION

We presented a data augmentation approach that addressed class imbalanced pattern classification problems by combining features from GAN and SMOTE. Our results show that the proposed framework significantly improves the majority of the class imbalanced problems. There were improvements of up to 9% on the F1 score for the benchmark datasets. Since it is an offline pre-processing technique with a reasonable time complexity order of $O(N^2d^2T)$, it does not affect the efficiency of the training process. We also visualized the learning process and found out that the AUC of SMOTified-GAN is better than the 2nd best algorithm up to 2.1% (for Spambase) and 3.5% (for Shuttle).

There are several possible future directions from this work such as applying SMOTified-GAN to other neural networks

such as CNNs and recurrent neural networks (RNNs) to oversample imbalanced image datasets and time-series data, respectively. Furthermore, different variations and combinations of SMOTE and GAN for the new model of SMOTified-GAN can improve it even further.

It will be interesting to investigate the conjoining of GAN with other over-sampling techniques such as MCMC. Its sampling method on a Bayesian framework can be used to incorporate uncertainty in the predictions and develop a prob-abilistic data generation process via GANs. The proposed framework can be used in a wide range of problems that face challenges when it comes to class imbalance issues. This framework can also be used to improve few-shot learning [69] to address problems where the model finds it difficult to draw decision boundaries due to a lack of data. Moreover, we can also investigate if the method can be used to address the bias-variance problems in order to improve the generalization ability of the model given that the training data differs significantly from the test dataset.

CODE AND DATA

We provide Python code and data for extending this work further.[1]

REFERENCES

[1] C. X. Ling and V. S. Sheng, "Class imbalance problem," in *Encyclopedia of Machine Learning*, C. Sammut and G. I. Webb, Eds. Boston, MA, USA: Springer, 2010, p. 171.

[2] C. L. Castro and A. P. Braga, "Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 6, pp. 888–899, Jun. 2013.

[3] S. Makki, Z. Assaghir, Y. Taher, R. Haque, M.-S. Hacid, and H. Zeineddine, "An experimental study with imbalanced classification approaches for credit card fraud detection," *IEEE Access*, vol. 7, pp. 93010–93022, 2019.

[4] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data: Review of methods and applications," *Expert Syst. Appl.*, vol. 73, pp. 220–239, May 2017.

[5] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *J. Big Data*, vol. 6, no. 1, p. 27, Mar. 2019.

[6] Q. Wei and R. L. Dunbrack, "The role of balanced training and testing data sets for binary classifiers in bioinformatics," *PLoS ONE*, vol. 8, no. 7, Jul. 2013, Art. no. e67863.

[7] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, vol. 16, no. 4, pp. 449–475, Jul. 2013.

[8] J. Lee, Y. C. Lee, and J. T. Kim, "Fault detection based on one-class deep learning for manufacturing applications limited to an imbalanced database," *J. Manuf. Syst.*, vol. 57, pp. 357–366, Oct. 2020.

[9] Y. Zhuo and Z. Ge, "Gaussian discriminative analysis aided GAN for imbalanced big data augmentation and fault classification," *J. Process Control*, vol. 92, pp. 271–287, Aug. 2020.

[10] S. Huang and K. Lei, "IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks," *Ad Hoc Netw.*, vol. 105, Aug. 2020, Art. no. 102177.

[11] S. S. Mullick, S. Datta, and S. Das, "Generative adversarial minority over-sampling," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1695–1704.

[12] A. Bria, C. Marrocco, and F. Tortorella, "Addressing class imbalance in deep learning for small lesion detection on medical images," *Comput. Biol. Med.*, vol. 120, May 2020, Art. no. 103735.

[13] Z. Qin, Z. Liu, P. Zhu, and Y. Xue, "A GAN-based image synthesis method for skin lesion classification," *Comput. Methods Programs Biomed.*, vol. 195, Oct. 2020, Art. no. 105568.

[14] D. Rodriguez, I. Herraiz, R. Harrison, J. Dolado, and J. C. Riquelme, "Preliminary comparison of techniques for dealing with imbalance in software defect prediction," in *Proc. 18th Int. Conf. Eval. Assessment Softw. Eng. (EASE)*. New York, NY, USA: Association for Computing Machinery, 2014.

[15] A. F. Hilario, S. G. Lopez, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning From Imbalanced Data Sets*. Cham, Switzerland: Springer, 2018.

[16] T. Zhu, Y. Lin, and Y. Liu, "Improving interpolation-based oversampling for imbalanced data learning," *Knowl.-Based Syst.*, vol. 187, Jan. 2020, Art. no. 104826.

[17] X. Tao, Q. Li, W. Guo, C. Ren, Q. He, R. Liu, and J. Zou, "Adaptive weighted over-sampling for imbalanced datasets based on density peaks clustering with heuristic filtering," *Inf. Sci.*, vol. 519, pp. 43–73, May 2020.

[18] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002.

[19] L. Torgo, R. P. Ribeiro, B. Pfahringer, and P. Branco, "Smote for regression," in *Proc. Portuguese Conf. Artif. Intell.* Berlin, Germany: Springer, 2013, pp. 378–389.

[20] P. Jeatrakul, K. W. Wong, and C. C. Fung, "Classification of imbalanced data by combining the complementary neural network and SMOTE algorithm," in *Proc. Int. Conf. Neural Inf. Process.* Berlin, Germany: Springer, 2010, pp. 152–159.

[21] S. Li, S. Huang, and Y. Zhou, "Toxic behaviour detection based on improved SMOTE algorithm and bi-LSTM network," *Int. J. Intell. Internet Things Comput.*, vol. 1, no. 2, pp. 114–128, 2020.

[22] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla, "SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary," *J. Artif. Intell. Res.*, vol. 61, pp. 863–905, Apr. 2018.

[23] M. Zareapoor, P. Shamsolmoali, and J. Yang, "Oversampling adversarial network for class-imbalanced fault diagnosis," *Mech. Syst. Signal Process.*, vol. 149, Feb. 2021, Art. no. 107175.

[24] G. Douzas and F. Bacao, "Effective data generation for imbalanced learning using conditional generative adversarial networks," *Expert Syst. Appl.*, vol. 91, pp. 464–471, Jan. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417417306346

[25] M. Zheng, T. Li, R. Zhu, Y. Tang, M. Tang, L. Lin, and Z. Ma, "Conditional Wasserstein generative adversarial network-gradient penalty-based approach to alleviating imbalanced data classification," *Inf. Sci.*, vol. 512, pp. 1009–1023, Feb. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025519309715

[26] J. Zhai, J. Qi, and C. Shen, "Binary imbalanced data classification based on diversity oversampling by generative models," *Inf. Sci.*, vol. 585, pp. 313–343, Mar. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025521011804

[27] L. Zhang, H. Yang, and Z. Jiang, "Imbalanced biomedical data classification using self-adaptive multilayer ELM combined with dynamic GAN," *Biomed. Eng. OnLine*, vol. 17, no. 1, p. 181, Dec. 2018.

[28] P. Xu, R. Du, and Z. Zhang, "Predicting pipeline leakage in petrochemical system through GAN and LSTM," *Knowl.-Based Syst.*, vol. 175, pp. 50–61, Jul. 2019.

[29] D. C. Knill and W. Richards, *Perception as Bayesian Inference*. Cambridge, U.K.: Cambridge Univ. Press, 1996.

[30] G. E. P. Box and G. C. Tiao, *Bayesian Inference in Statistical Analysis*. Hoboken, NJ, USA: Wiley, Jan. 2011.

[31] C. Andrieu, N. De Freitas, A. Doucet, and M. I. Jordan, "An introduction to MCMC for machine learning," *Mach. Learn.*, vol. 50, no. 1, pp. 5–43, Jan. 2003.

[32] S. Brooks, A. Gelman, G. Jones, and Xiao-Li, "MCMC using Hamiltonian dynamics," in *Handbook of Markov Chain Monte Carlo*, vol. 2, no. 11. Boca Raton, FL, USA: Chapman & Hall, 2011, p. 2.

[33] D. J. C. MacKay, "A practical Bayesian framework for backpropagation networks," *Neural Comput.*, vol. 4, no. 3, pp. 448–472, May 1992.

[34] D. van Ravenzwaaij, P. Cassey, and S. D. Brown, "A simple introduction to Markov chain Monte-Carlo sampling," *Psychonomic Bull. Rev.*, vol. 25, no. 1, pp. 143–154, Feb. 2018.

[35] B. Das, N. C. Krishnan, and D. J. Cook, "RACOG and wRACOG: Two probabilistic oversampling techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 1, pp. 222–234, Jan. 2015.

[36] J. E. Johndrow, A. Smith, N. Pillai, and D. B. Dunson, "MCMC for imbalanced categorical data," *J. Amer. Stat. Assoc.*, vol. 114, no. 527, pp. 1394–1403, Jul. 2019.

[37] R. Chandra, K. Jain, R. V. Deo, and S. Cripps, "Langevin-gradient parallel tempering for Bayesian neural learning," *Neurocomputing*, vol. 359, pp. 315–326, Sep. 2019.

---

[1] https://github.com/sydney-machine-learning/GANclassimbalanced

[38] R. Chandra, M. Jain, M. Maharana, and P. N. Krivitsky, "Revisiting Bayesian autoencoders with MCMC," 2021, *arXiv:2104.05915*.

[39] R. Chandra, A. Bhagat, M. Maharana, and P. N. Krivitsky, "Bayesian graph convolutional neural networks via tempered MCMC," 2021, *arXiv:2104.08438*.

[40] B. Das, N. C. Krishnan, and D. J. Cook, "WRACOG: A Gibbs sampling-based oversampling technique," in *Proc. IEEE 13th Int. Conf. Data Mining*, Dec. 2013, pp. 111–120.

[41] K. H. Kim and S. Y. Sohn, "Hybrid neural network with cost-sensitive support vector machine for class-imbalanced multimodal data," *Neural Netw.*, vol. 130, pp. 176–184, Oct. 2020.

[42] I. Irigoien, B. Sierra, and C. Arenas, "Towards application of one-class classification methods to medical data," *Sci. World J.*, vol. 2014, Mar. 2014, Art. no. e730712.

[43] L. Gao, L. Zhang, C. Liu, and S. Wu, "Handling imbalanced medical image data: A deep-learning-based one-class classification approach," *Artif. Intell. Med.*, vol. 108, Aug. 2020, Art. no. 101935.

[44] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, pp. 1345–1359, Nov. 2010.

[45] D. Dheeru and E. K. Taniskidou. (2019). UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. [Online]. Available: http://archive.ics.uci.edu/ml

[46] S. Raschka, "STAT 479—Machine learning (fall 2018)," Univ. Wisconsin, Madison, WI, USA, Tech. Rep. STAT 479, 2018. [Online]. Available: http://pages.stat.wisc.edu/~sraschka/teaching/stat479-fs2018/

[47] Y.-S. Won, D. Jap, and S. Bhasin, "Push for more: On comparison of data augmentation and SMOTE with optimised deep learning architecture for side-channel," in *Information Security Applications* (Lecture Notes in Computer Science), I. You, Ed. Cham, Switzerland: Springer, 2020, pp. 227–241.

[48] R. Blagus and L. Lusa, "Evaluation of SMOTE for high-dimensional class-imbalanced microarray data," in *Proc. 11th Int. Conf. Mach. Learn. Appl.*, vol. 2, Dec. 2012, pp. 89–94.

[49] X. Zheng, "SMOTE variants for imbalanced binary classification: Heart disease prediction," Ph.D. dissertation, UCLA, Los Angeles, CA, USA, 2020. [Online]. Available: https://escholarship.org/uc/item/99x0w9w0

[50] G. Kovács, "Smote-variants: A Python implementation of 85 minority oversampling techniques," *Neurocomputing*, vol. 366, pp. 352–354, Nov. 2019.

[51] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, vol. 27, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2014.

[52] S. Suh, H. Lee, P. Lukowicz, and Y. O. Lee, "CEGAN: Classification enhancement generative adversarial networks for unraveling data imbalance problems," *Neural Netw., Off. J. Int. Neural Netw. Soc.*, vol. 133, pp. 69–86, Jan. 2021.

[53] A. Ali-Gombe and E. Elyan, "MFC-GAN: Class-imbalanced dataset classification using multiple fake class generative adversarial network," *Neurocomputing*, vol. 361, pp. 212–221, Oct. 2019.

[54] V. Sorin, Y. Barash, E. Konen, and E. Klang, "Creating artificial images for radiology applications using generative adversarial networks (GANs)—A systematic review," *Academic Radiol.*, vol. 27, no. 8, pp. 1175–1185, Aug. 2020.

[55] J. Lin, Y. Li, and G. Yang, "FPGAN: Face de-identification method with generative adversarial networks for social robots," *Neural Netw.*, vol. 133, pp. 132–147, Jan. 2021.

[56] P. Yi, Z. Wang, K. Jiang, J. Jiang, T. Lu, and J. Ma, "A progressive fusion generative adversarial network for realistic and consistent video super-resolution," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Dec. 3, 2021, doi: 10.1109/TPAMI.2020.3042298.

[57] A. Sharma, "Guided parallelized stochastic gradient descent for delay compensation," *Appl. Soft Comput.*, vol. 102, Apr. 2021, Art. no. 107084.

[58] F. Hu and H. Li, "A novel boundary oversampling algorithm based on neighborhood rough set model: NRSBoundary-SMOTE," *Math. Problems Eng.*, vol. 2013, Nov. 2013, Art. no. e694809.

[59] Z. Jefferson. (Aug. 2020). *Bank Data: SMOTE*. [Online]. Available: https://medium.com/analytics-vidhya/bank-data-smote-b5cb01a5e0a2

[60] (2021). *Train Generative Adversarial Network (GAN)—MATLAB & Simulink—MathWorks*. [Online]. Available: https://au.mathworks.com/help/deeplearning/ug/train-generative-adversarial-network.html

[61] A. Sharma, "Guided stochastic gradient descent algorithm for inconsistent datasets," *Appl. Soft Comput.*, vol. 73, pp. 1068–1080, Dec. 2018.

[62] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," Dec. 2014, *arXiv:1412.6980*.

[63] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," Dec. 2012, *arXiv:1212.5701*.

[64] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, vol. 27, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2014.

[65] U. Bhowan, M. Johnston, M. Zhang, and X. Yao, "Reusing genetic programming for ensemble selection in classification of unbalanced data," *IEEE Trans. Evol. Comput.*, vol. 18, no. 6, pp. 893–908, Dec. 2014.

[66] H. Núñez, L. Gonzalez-Abril, and C. Angulo, "Improving SVM classification on imbalanced datasets by introducing a new bias," *J. Classification*, vol. 34, no. 3, pp. 427–443, Oct. 2017.

[67] K. Napierala and J. Stefanowski, "Types of minority class examples and their influence on learning classifiers from imbalanced data," *J. Intell. Inf. Syst.*, vol. 46, no. 3, pp. 563–597, Jun. 2016.

[68] *Dataset: Credit Card Fraud Detection*. Accessed: Aug. 1, 2021. [Online]. Available: https://kaggle.com/mlg-ulb/creditcardfraud

[69] Y. Wang, Q. Yao, J. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," Mar. 2020, *arXiv:1904.05046*.

ANURAGANAND SHARMA received the B.S. and M.S. degrees in computer science from The University of the South Pacific, Fiji, and the Ph.D. degree in artificial intelligence entitled "Information Discovery from Constraints for Evolutionary Computational Models" from the University of Canberra, Australia, in 2014.

From 2003 to 2005, he was a Software Developer with Colonial, Fiji. Since 2007, he has been an Assistant Lecturer to a Senior Lecturer in computer science with The University of the South Pacific. He is currently the DHoS (research) with the School of Information Technology, Engineering, Mathematics & Physics (STEMP). His research interests include deep learning with CNN and constraint optimization with meta-heuristic algorithms. His recent work includes optimization and enhancement of SGD for gradient-based learning systems.

PRABHAT KUMAR SINGH is currently pursuing the B.E. degree in mechanical engineering with the Department of Mechanical Engineering, IIT (BHU) Varanasi, India. He was a Research Intern with UNSW Sydney, Australia, for this project. His research interests are in deep learning and class imbalanced problems.

ROHITASH CHANDRA (Senior Member, IEEE) is a Senior Lecturer in Data Science at the School of Mathematics and Statistics, UNSW. He leads a program of research encircling methodologies and applications of artificial intelligence, particularly in areas of Bayesian deep learning, climate extremes, geoscientific models, and mineral exploration. His current research focus is in the application of artificial intelligence methods for COVID-19 modeling, bioinformatics, and philosophy of religion. His research has been funded by the Australian Research Council (ARC), National Health and Medical Research Council (NHMRC), and the Food and Drug Administration (FDA). He is an Associate Editor of IEEE Transactions on Neural Networks and Learning Systems, *Neurocomputing*, and *Geoscientific Model Development*.

• • •