Generative Adversarial Networks for Actuarial Use

K.S. Ngwenduna sngwenduna@gmail.com
School of Computer Science and Applied Mathematics, University of the Witwatersrand, Braamfotein, 2001, South Africa

R. Mbuvha rendani.mbuvha@wits.ac.za
School of Statistics and Actuarial Science, University of the Witwatersrand, Braamfontein, 2001, South Africa

Abstract

Gaining an advantage in competitive markets through offerings of suitable tailored products to customers relies on adequate predictive models. To build these models, a substantial amount of data and a sizeable number of records is desirable. However, when expanding to new or unexplored territories, that level of information is rarely always available. To build such models, actuarial firms have to buy data from a local provider, through a re-insurer, through limited unsuitable industry and public research or rely from extrapolations from other better known markets. In this work, we show how an implicit model using Generative Adversarial Network (GAN) can alleviate this problem through the generation of adequate quality data even from very limited small samples, from difficult domains or without alignment. Through the paper, we explain what GANs are and how they can be used to synthesize data in order to accurately enhance very infrequent events and create better prediction models. In addition, this work provides theoretical and practical applications of GANs. Overall, we show a significant superiority of GANs for predictive models and stochastic simulations compared to current approaches on example data sets using Python. This work offers a number of contributions to actuaries for data augmentation, boosting predictive models, attention prediction, anomaly detection, domain adaptation, data manipulation, privacy preservation, missing data imputation and discriminative modelling using GANs.

Keywords

Actuarial Science, Generative models, Generative Adversarial Network (GAN), SMOTE, Wasserstein GAN.

1 Introduction

"...in the process of training generative models, we will endow the computer with the understanding of the world and what is made up of."

OpenAI

1.1 Background

Gaining an advantage in competitive markets through offerings of suitable tailored products on customers relies on building and maintaining adequate predictive models. To build these models, a substantial amount of data and a sizeable number of records is desirable. However, when expanding to new or unexplored markets, that level of information is rarely always available. To build such models, actuarial firms have to buy

data from a local provider, through a re-insurer, through limited unsuitable industry and public research or rely from extrapolations from other better known markets. In this work, we show how an implicit model using the Generative Adversarial Network (GAN) [69] can alleviate this problem through the generation of adequate quality data even from very limited small samples, from difficult domains or without alignment.

A GAN is an example of a generative model that is used to create new samples from a latent noise space. A generative model describes how a data set is generated in terms of a probabilistic model [68]. This generative model p_{model} mimics the training data distribution p_{data} as close as possible. If this is achieved, then we can sample from p_{model} to generate realistic samples that appear to have been drawn from p_{data}. We are satisfied if our model can also generate diverse samples that are suitable different from the training data. In some cases, the model can be estimated explicitly and sometimes it can generate samples implicitly. Other models are capable of doing both.

GANs are useful for learning the structure of the data and can generate new samples without explicitly postulating the model. GANs were invented in 2014 and have generated more interest since then. They are better than other generative models, used for data augmentation, boosting predictive models, attention prediction, anomaly detection, domain adaptation, data manipulation, privacy preservation, missing data imputation and discriminative modelling. GANs have been highly successful in computer vision [23, 94, 207], e-commerce [104], medicine [7], anime character creation [91], video generation [184], super resolutions [200], music generation [192], text/speech generation [156], missing data imputation [110, 169, 196], time series generation [49] and tabular data sets [145], with remarkable results, but their application to the actuarial discipline remains largely still unexplored.

1.2 Aims and Objectives

Through the paper, we briefly explain what GANs are and how they can be used to synthesize data in order to accurately enhance very infrequent events and create better prediction models. In addition, this work offers to provide theoretical and practical applications of GANs. Specifically, this paper covers the following aims and objectives:

- Deep overview of generative models and why GANs are of better quality than other generative models;
- An overview of the GAN architecture with practical applications in a number of areas, especially for an actuarial use; and
- Cover some open challenges with GANs, including recent advances and scope for the future.

We demonstrate a popular GAN architecture to a typical problem resembling an actuarial use on a number of data sets using Python [54]. This task can equally be accomplished and replicated using insurance, health care, banking and investment data such as claim frequency, claim severity, stock prediction and so on. Overall, we show a significant superiority of GANs for predictive models and stochastic simulations compared to current approaches.

The rest of the paper is organised as follows. Section 2 reviews the literature on neural networks and generative models, with particular emphasis on GANs. Section 3 describes GANs theoretically as well as the popular SMOTE [26] synthetic data generative scheme for comparative purposes with GAN. Section 4 outlines the example experiments conducted. Section 5 presents the results and discusses them, while Section 6 gives conclusions, limitations and possible future work.

2 Literature Review

Generative models have received lots of interests from researchers for creating new samples. This section covers a taxonomy of these techniques, with a particular focus on GANs.

2.1 Deep Learning

This section describes Artificial Neural Networks (ANNs) as they form the foundation work for GANs.

2.1.1 Layers

From a statistical viewpoint, an ANN represents a nested combination of several functions stacked sequentially to yield a desired output [68]. An example of an ANN is shown in Figure 1.

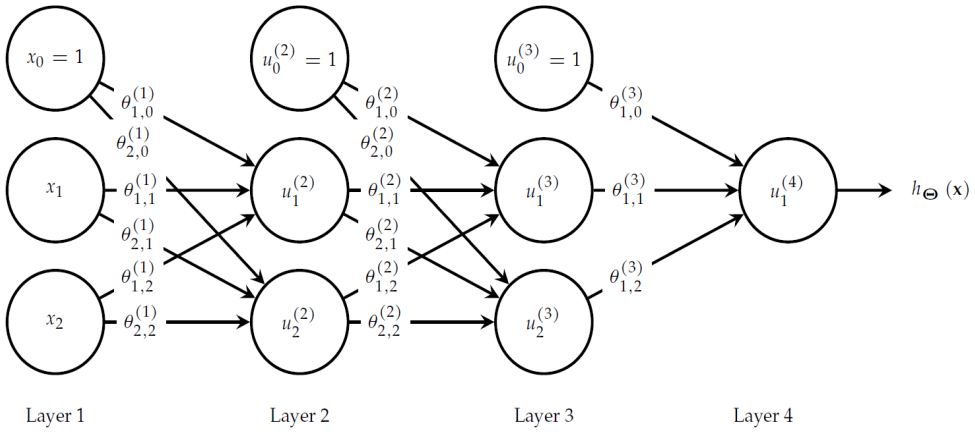


Figure 1: A fully connected MLP example with three layers

An ANN has input features, hidden layers and the target which gives the resulting output [68]. Deep Learning is concerned with many complex layers of the ANN. Each input feature is assigned a weight θ which represents its importance. Hidden layers receive inputs from prior nodes beneath them and propagate the output to other hidden layers above them [68, 131].

As an illustration, Figure 1 shows an ANN with three layers, with two-dimensional input, two layers with three units and one output layer with one unit, where $u_i^l = g_l(\Theta_l + b_l)$, x_i is the input, x_0 is the bias term, $\Theta_{l,i}^l$ is a weight parameter for each layer and $h_\theta(x^n)$ is the predicted target. This ANN can be a regression or a classifier, depending on the activation function in the output.

2.1.2 Activation Functions

An input X is multiplied by a weight θ, results added together and the resulting sum flows through an activation function. Activation functions introduce non-linearities and transmit the resulting output into the target output [68, 131], restricting the output to a certain finite value [57]. A key characteristic of activation functions is that they must be continuously differentiable. Table 1 lists common activation functions where z is the result of the weight matrix Θ multiplied by the feature vector X and $g(.)$ is the activation function.

Activation Function	Formula
Linear	$g(z) = z$
Sigmoid	$g(z) = \frac{1}{1+\exp^{-z}}$
Hyperbolic tangent (tanh)	$g(z) = \frac{\exp^{z} - \exp^{-z}}{\exp^{z} + \exp^{-z}}$
Rectified Linear Unit (ReLU) [66]	$g(z) = max\left(0, z\right)$
Leaky ReLU [121]	$g(z) = max\left(0.01z, z\right)$
PReLU [76]	$g(z) = max\left(\alpha z, z\right)$
Softplus [137]	$g(z) = \log\left(1 + \exp^{z}\right)$
Swish [154]	$g(z) = z.sigmoid\left(\beta z\right)$

Table 1: Taxonomy of activation functions for ANNs

The sigmoid outputs a vector where each element is a probability, bounded between 0 and 1 and is typically adopted in the final layer for binary problems. ReLU ranges between zero and infinity and is known for being robust against vanishing gradients [33, 66, 121]. Leaky ReLU [121] solves the dying ReLU problem. The tanh function is bounded between -1 and 1 and has been a default activation function in the hidden layers until ReLU was proposed [66]. The tanh function gives values of different signs which makes it easier to decide which scores to consider in the next layer and which to ignore. However, it shares the unfortunate weakness of vanishing gradients with the sigmoid activation function [33, 66, 121].

Parametric ReLU (PReLU) [76] is of the same form as Leaky ReLU except that it has a scalable and learnable parameter α. Softplus [137] is a smoother version of ReLU [154]. Ramachandran, Zoph, and Le [154] show that the Swish activation function behaves in a similar manner as the ReLUs and worked better on many challenging data sets. It remains to be seen if recent activation functions such as Swish [154] and Mish [130] will replace ReLU and Leaky ReLU in the future.

2.1.3 Gradient Descent

The weights Θ are optimised to minimise a loss function [68]. This means that training an ANN means to show it many examples, make predictions through feed-forward computations and then compare them with the actual labels to compute the resulting loss. Finally, the ANN adjusts these weights from all nodes until it gets a desired minimum loss value and thus optimal weights. Mathematically, for a binary problem, the loss function $J(\theta)$ to be minimised is:

$$J\left(\Theta\right) = \frac{-1}{N}\left[\sum_{i=1}^{N}\sum_{k=1}^{K} y_k^{(i)} \log\left(h_\theta(x^{(i)})_k\right) + \left(1 - y_k^{(i)}\right) \log\left(1 - h_\theta(x^{(i)})_k\right)\right] \tag{1}$$

where N is the size of the data set, $h_\theta(x^{(i)})_k$ is the predicted target, θ_k's are the unknown coefficients, X is the feature vector and $y_k^{(i)}$ is the actual target. Gradient Descent (GD) optimises the above loss function [161]. GD finds the most optimal weights Θ iteratively [78] using the following process:

- Initialise weights $\theta_k^{(0)}$ randomly using He or Xavier initialisation;
- Loop until convergence i.e. until sufficient number of epochs t are reached:
- Compute the gradient $\frac{\partial J(\Theta)}{\partial \theta_k}$;
- Update weights θ_k using the learning rate η to move towards the minimum loss,

$\theta_k^{(t)} = \theta_k^{(t-1)} - \eta \frac{\partial J(\Theta)}{\partial \theta_k}$;

- Return weights Θ.

Other optimisation approaches can be used such as second order approximations i.e. Newton's method. However, these methods tend to be infeasible for high dimensions and large training data sets [161]. Thus GD is the most popular and common approach for solving ANN weights.

The above process is called Batch GD (BGD) as the weights are updated using the entire data set [161]. This can be very slow, intractable and does not allow to update weights online [47, 98, 198]. Stochastic GD (SGD) updates the weights one sample at a time [161].

However, SGD usually performs frequent updates and this leads to volatility as there might be fluctuations and over-shooting [116]. A compromise between BGD and SGD is called mini-batch GD and this updates weights using a batch of m training samples. Typically, mini-batch training samples can be anything from 50 to 256 but this could vary with different domains [161].

However, even though mini-batch GD tends to be better than BGD or SGD, it may still be slow in convergence due to η [47, 98]. The learning rate η specifies how fast an ANN updates its weights. If η is too small, the model may not converge or descend slowly and this can be computationally expensive [116]. If η is too large, the model may take gigantic descents and miss the global minimum [46, 98, 139]. Adaptive learning rates have been proposed to improve η (shown in Table 2).

Basically, the algorithms incorporate a term to adapt η or use exponential moving average of current and/or past gradients [98, 130, 203]. Adaptive Moment estimation (Adam) [98] is the most popular and recommended algorithm for solving weights of an ANN [116, 155, 161]. Table 2 shows a taxonomy of GD optimisers, differing on two ways on either modifying η or modifying the gradient component or both.

GD optimiser	Year	Learning rate	Gradient
Momentum [151]	1964		√
Adaptive gradient (AdaGrad) [47]	2011	√	
RMSprop	2012	√	
Adaptive delta (Adadelta) [198]	2012	√	
Nesterov Accelerated Gradient (NAG) [139]	2013		√
Adam [98]	2014	√	√
AdaMax [98]	2015	√	√
Nadam [46]	2015	√	√
AMSGrad [155]	2018	√	√
Rectified Adam (RAdam) [116]	2019	√	√
LookAhead, Ranger [203]	2019	√	√

Table 2: Gradient descent optimisation variants

Other algorithms include SGD with Momentum [151], Adaptive gradient (AdaGrad) [47], Root mean square prop (RMSprop) [1], Adaptive delta (Adadelta) [198], Nesterov Accelerated Gradient (NAG) [139], AdaMax [98], Nadam [46] and AMSGrad which is another variant of Adam [155]. We describe three popular GD optimisation variants as these are typically used in most deep ANNs: Momentum, RMSprop and Adam.

[1] https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

2.1.3.1 Momentum Momentum helps in accelerating SGD in the correct direction and dampening oscillations [151] using the following equation:

$$\theta_{t+1,i} = \theta_{t,i} - V_{t,i}$$
$$V_{t,i} = \gamma V_{t-1,i} + \eta \frac{\partial J(\Theta)}{\partial \theta_{t,i}}$$

where V_t is the velocity representing the exponential moving average of past gradients and γ is the momentum (typically 0.9) [161]. Values for V_t are typically initialised close to zero.

2.1.3.2 RMSprop Root Mean Square propagation (RMSprop[2]) chooses a different η for each weight Θ. RMSprop is an unpublished GD optimiser that was invented by Geoff Hinton. RMSprop is formulated as:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{E\left[g_{t,i}^2\right]} + \varepsilon}$$
$$E[g_{t,i}^2] = \gamma E\left[g_{t-1,i}^2\right] + (1-\gamma)\, g_{t,i}^2$$
$$g_{t,i} = \frac{\partial J(\Theta)}{\partial \theta_{t,i}}$$

where typically $\gamma = 0.9$, $\eta = 0.001$, ε avoids null division and $E[g^2]_{t,i}$ represents the running average of past gradients at time step t.

2.1.3.3 Adam Adam is a combination of Momentum and RMSprop. Kingma and Ba [98] show a superior performance of Adam over other optimisers. Adam is defined below:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon}\hat{m}_t$$
$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$
$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$
$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\, g_t$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\, g_t^2$$

where $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected first m_t and second moment estimates v_t of the gradients respectively, typically initialised to 0's. The parameters can be estimated via cross-validation approach or using default values proposed by the authors as per Keras documentation [55, 98, 161].

In general, Adam has been empirically shown to work well in practise and compares fairly well with other optimisers [55, 98, 116, 153, 155, 161]. However, it remains to be seen if recent optimisers such as Rectified Adam (RAdam) [116], LookAhead and Ranger [203] will consistently outperform Adam in the future.

[2] https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

2.2 Weight Initialisation

To conduct GD, weights Θ needs to be initialised. The weights can affect how quickly or if at all the local minimum is found by the network training algorithm [67], known as the exploding gradient problem.

Two popular approaches are the He (if using ReLU/Leaky ReLU in the hidden layers) [76] and Xavier (if using tanh in the hidden layers) [65] initialisation. He initialisation initialises weights from a standard Gaussian distribution and then multiplied by the square root of $(2/n_i)$ where n_i is number of input units for that layer. Xavier initialisation works by replacing 2 with a 1 instead.

2.3 Regularisation

The specification of many of the hyper-parameters in ANNs could often cause over-fitting or under-fitting [68]. Regularisation is a practice in ML that is used to curb over-fitting. Typically, **batch normalisation** [88], **drop-out** [174], **early stopping**, **L1** and **L2** regularisation [57, 179] can be used.

2.3.1 Batch normalisation

Batch normalisation standardizes hidden layers such that they have a mean 0 and unit standard deviation for each training mini-batch as units flow through each layer [88]. In practise, this results in faster, more stable training and a regularization effect [88, 153].

2.3.2 Drop-out

In drop-out, some units in the layer are temporarily excluded at random from the training [174]. The drop-out parameter is typically in the range $[0, 1]$ with 0.5 the most popular value for retaining the output of each layer [67, 153, 174]. This can be implemented per layer in the network. This forces the training process to be more noisy, allowing each layer to take flexible responsibility for the inputs.

2.3.3 Early Stop

GD proceeds in epochs which consist of using the training set entirely to update each parameter [161]. Initially, weights Θ are initialised. Then at each epoch, the weights are updated using partial derivatives using any GD optimiser until the process the weights do not change much i.e. until convergence [65, 76]. Typically, we require many epochs until this convergence and then we stop. Early stop is a practice where training is stopped when the cost starts increasing steadily instead of decreasing. One can then stop training the model at that epoch.

2.3.4 L1 and L2 regularisation

Regularisation enforces the ANN to learn a less complex model by adding a penalising term Equation 1 [179].

L1 regularisation performs sparse modelling by adding $\lambda \sum_{k=1}^{d} \theta_k$ to Equation 1 where λ is the importance parameter. This shrinks some coefficients to zero, yielding to implicit variable selection. This method is preferred for model explainability. L2 regularisation or ridge regression adds $\lambda \sum_{k=1}^{d} \theta_k^2$ to Equation 1. This is typically preferred for maximising model performance [57]. Elastic net combines both L1 and L2 regularisations. Hyper-parameterisation can be done in order to choose which approach is desirable.

2.3.5 Summary

There are a number of parameters to tune in ANNs. Typically, weight initialisation, activation function, η, the number of layers, regularisation approach, the number of neurons, GD optimiser and the number of epochs are required before training an ANN. There is no best approach but some heuristics and best practise are typically followed. In this work, ANNs provide theoretical foundations for GANs.

2.4 Generative Modelling

Whilst machine learning (ML) has gained significant prevalence in the past few decades, class imbalance, limited data sets and missing data remain pervasive problems. These issues occur due to the nature of the data space, data collection costs, limitations, new markets and absolute rarity [89, 119]. These issues create problems for building models as they lead to inadequate data, leading to inaccurate or misleading models, biased accuracy measures and subjective uncertainty margins which create further model risk [52].

This research is concerned with handling these problems through generative modelling. Other approaches exist such as synthetic sampling, however, these approaches do not take into account the underlying structure of the data distribution and often lead to over-fitting and over-lapping cases [62, 199]. Generative models are flexible models capable of learning the data distribution and sampling from this data distribution, thereby creating new synthetic cases. In this section, we review generative models and explain why GANs are of better quality than other deep generative models.

2.4.1 Definition

Given a data set with observations X, we assume that X has been generated from an unknown probability density function (PDF) p_{data}. A generative model p_{model} mimics p_{data} as close as possible. If this is achieved, then we can sample from p_{model} to generate realistic samples that appear to have been drawn from p_{data}. We are satisfied if our model can also generate diverse samples that are suitable different from X. In some cases, the model can be estimated explicitly and sometimes it can generate samples implicitly. Other models are capable of doing both. GANs provide no estimate of the model but are capable of generating new data without knowing it.

Goodfellow [67] provides a taxonomy of common deep generative models show in Figure 2, divided into implicit and explicit models. GANs are designed to remedy most of the disadvantages that come with explicit models and other Markov chain models.

2.4.2 Explicit models

Explicit models specify or approximate a parameterised log-likelihood representation of the data [69]. Parameters are then estimated and learned from the data and this requires a maximum likelihood estimation which integrates over the entire data space and this may be intractable [111]. These approximation techniques may not always yield the best results as some of them rely on Markov chains which are time-consuming [69].

Two popular tractable models are fully visible belief networks (FVBNs) [56] and nonlinear independent component analysis (ICA). Approximate methods improve on the design of tractable models which can be computational intensive and limited [69, 123, 158]. Approximate methods use either deterministic i.e. variational inference or stochastic approximations i.e. MCMC approaches. Variational inference involves the use of Variational Autoencoders (VAEs) [99, 158] to approximate $p_{model}(x)$ using lower bounds.

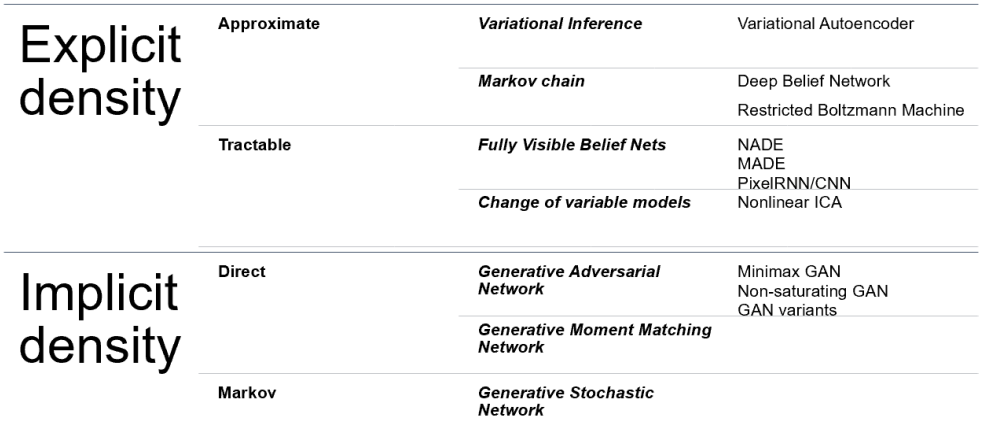


Figure 2: Taxonomy of generative models

2.4.2.1 FVBNs FVBN estimates the PDF of the training data $p_{model}(x)$ into a decomposed product of one-dimensional probability distributions. This model outputs a probability for each possible value if x is discrete and outputs a network of parameters of a simple distribution if x is continuous. Using the generated model, sampling is done one step at a time, conditioned on all previous steps [69].

The problem with these models is their computational complexities as they need to generate one point at a time. Other problems include poor learning representations, over-emphasizing details over global data and not closely reflecting the true generation process [67]. Moreover, these models have been more useful for image synthesis than structured data sets such as tabular data [144]. GANs are known to provide new samples in parallel, thus yielding greater speed of generation [67, 111].

2.4.2.2 Non-linear ICA Non-linear ICA involves defining some continuous non-linear transformations of data between high dimensions and lower dimensional spaces. The distribution of the data p_{model} is transformed into a distribution of a latent space z defined by $p_z(g)$ where g is some tractable transformed version of p_z. The challenge in ICA is finding tractable distributions in the latent space and these are limited [68]. GANs are known to have fewer restrictions than these models [14, 67, 69].

2.4.2.3 Variational Autoencoders VAEs, along with FVBNs and GANs, are three of the most popular approaches for sample generation. VAEs are an extension to AEs [12, 105, 158]. AE learns useful representations of the data by encoding X into a compressed latent space z using $q(z|x)$ and then decoding z back into X using $p(x|z)$ by minimising the reconstruction error between the original data and the deconstructed data [12]. VAE maximizes the following function :

$$\log p(x) \geq E_{z\sim q(z|x)}\left[\log p(x|z) + \log p(z) - \log q(z)\right] \tag{2}$$

Unlike auto-regressive models, VAEs are normally easy to run in parallel during training and inference [68, 105, 158]. Conversely, they are normally harder to optimize than auto-regressive models [68, 123]. The encoder converts the input to latent space representations through the mean and variance and samples can be created from the learned representation. VAEs have been criticised to be generating blurry samples and are intractable [68, 166].

2.4.2.4 Boltzmann Machines Boltzmann machines rely on the use of Markov chains to model $p_{model}(x)$ and to sample from it [1, 78, 165]. A Markov chain is a process that is used to generate samples by repeatedly drawing a sample from a transition operator [64]. A Boltzmann machine is an energy-based function defined as:

$$p_{model}(x) = \exp\left(-E(x)\right)/Z \tag{3}$$

where $E(x)$ is an energy function and Z is a normalizing factor to ensure that $p_{model}(x)$ sums to one [1, 68].

These methods include Restricted Boltzmann machine (RBM) [1] and Deep Belief Networks (DBNs) [79, 80]. DBNs and RBMs are generative stochastic neural networks that can estimate a PDF [1]. Samples are obtained through MCMC runs to convergence and this can be very expensive to run [111]. These models were pioneers in early 2006 for deep generative models but they have been rarely used because of poor scale-ability for higher dimension problems and high computational costs [68].

2.4.3 Implicit models

Implicit models learn to model the true distribution and define a stochastic procedure to directly generate new data from a latent space. These models can be trained indirectly without needing an explicit density function to be learned or defined. Some of these models such as the Generative Stochastic Network (GSN) [14] involve MCMC methods which impose greater computational cost and often fail to scale to higher dimensional spaces [68]. Generative Adversarial Networks (GANs) [69] and Generative Moment Matching Networks (GMMNs) [111] are one of the few implicit probabilistic models capable of sampling in parallel and in a single step.

2.4.3.1 GANs GANs were originally invented in a landmark paper by Ian Goodfellow in 2014 [69]. The setup of the framework uses an adversarial process to estimate the parameters of two artificial neural network (ANN) [162] models by iteratively and concomitantly training a discriminator (D) and a generator (G), as shown in Figure 3.

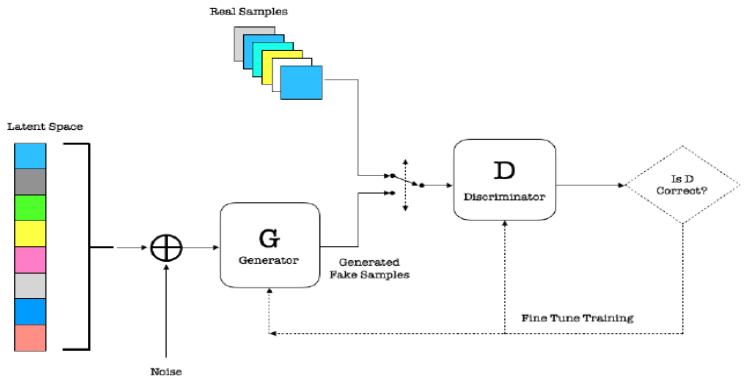


Figure 3: GAN operation

Through multiple cycles of generation and discrimination, both networks train each other, while simultaneously trying to outwit each other [69, 126, 143, 207]. GANs have two adversarial ANNs:

- G picks z from the prior latent space Z and then generates samples from this distribution using ANN;

- D receives generated samples from G and the true data examples, and must distinguish between the two for authenticity.

Both D and G are ANNs which play a zero-sum game, where G learns to produce realistic-looking samples and D learns to get better at discriminating between the generated samples and the true data. Once G is trained to optimality, it can create new samples and augment the training data set. GANs can sample in parallel better than other generative models, have fewer restrictions on the generator function, assume no use of Markov Chains, no variational bounds unlike VAE and produce subjectively better quality samples than other generative models [6, 67, 68, 69, 153, 166].

Whilst GANs are gaining popularity in many applications, they have notable issues. GANs are notoriously difficult to train properly, difficult to evaluate, the likelihood cannot be easily be computed, suffer from the vanishing gradient problem, mode collapse, boundary distortion and over-fitting [6, 68, 166].

Mode collapse is when many latent noise values z are mapped to the same data point x, leading to a lack of diversity in the samples that are created i.e. under-fitting. The vanishing gradient problem occurs when D becomes perfect in its training without giving G the chance to improve. As a result, GANs may fail to converge and thereby leading to poor generated samples [6]. Figure 4 provides a non-exhaustive taxonomy of GAN variants and improved training, including common examples [35, 81, 85, 188].

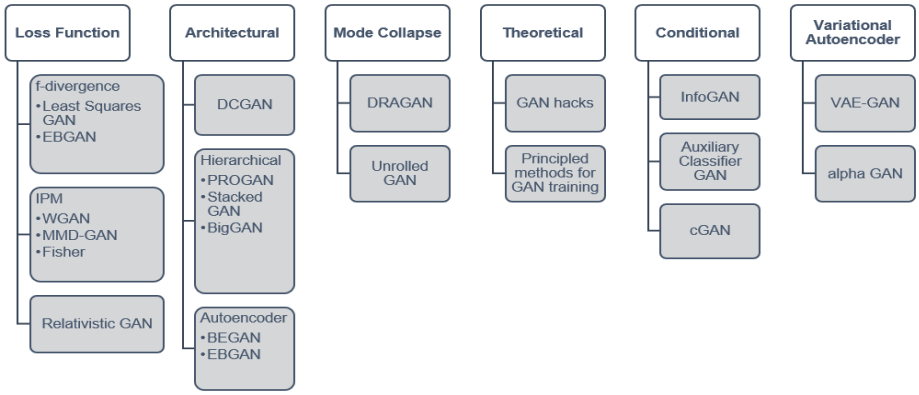


Figure 4: Taxonomy of GAN variants

Salimans et al. [166] look at ways to improve GANs (called hacks) while other authors propose variants to the vanilla GAN by changing the cost function, adding gradient penalties (GPs), adding labels, avoiding over-fitting and finding better ways of optimising GANs. The first extension of GAN was the Conditional GAN (cGAN) which gave the generator the label in the latent space, making them class conditional [129]. Until the introduction of Deep Convolutional GAN (DCGAN) [153], training GANs was very unstable. DCGANs provide some further tricks using convolutional and deconvolutional layers. Since then, more variants and heuristics were proposed.

Wasserstein GAN (WGAN) [6] proposes a different loss function, becoming the most studied and widely used GAN architecture ever since [85]. WGAN has been shown to give better quality of generated synthetic data than the vanilla GAN and alleviating most of the GAN issues [6]. Gulrajani et al. [72] further amend WGAN

through an addition of a GP to the cost function, coming with WGAN-GP.

In recent years, other loss functions which unify the GAN loss framework including f-divergence [141], Integral Probability Metrics (IPMs) [85] and Relativistic GANs (RGANs) [92], were proposed. The f-divergence measures the difference between p_{data} and p_g with a specific convex function f [141]. f-divergence GAN, IPMs and RGAN are considered unified frameworks suitable for stronger generalization to other loss-variants [124]. WGAN is a special class of IPMs and the most studied GAN. Other popular GAN loss variants include Least Squares GAN (LSGAN) [125], Boundary Equilibrium GAN (BEGAN) [17], Loss Sensitive GAN (LS-GAN) [150] and Energy-Based GAN (EBGAN) [204].

Other advanced GANs include the revolutionary Progressive Growing GAN (ProGAN) [94] which proposes a progressive growing and steps towards GAN performance. Other variants include Self-attention GAN [200] and BigGAN [23] which achieved tremendous performances on Imagenet data sets. There have been hybrids of GANs and VAEs where VAEs are used to encode the latent space to come up with VAE-GAN [105].

For further GAN reviews, Creswell et al. [35], Hitawala [81] and Hong et al. [85] provide a comparative overview. Lucic et al. [120] conduct an in-depth study on GANs and note no significant performance differences on the GANs studied. There are over 300 GAN variants and it is impossible to review all of them.

2.4.3.2 GMMNs GMMNs minimize the maximum mean discrepancy (MMD) between the moments of p_{data} and p_{model} and are known to be simpler than other generative models [111]. Moment matching evaluates whether the moments of the true distribution $p_{true}(x)$ match those of the data $p_{data}(x)$ through MMD. This approach is similar to GANs in terms of training except using a different loss function which leads to faster sampling. However, GMMNs have received less attention than GANs and VAEs, limiting their sample generative scheme [6, 68, 81].

2.4.4 Summary

There are a number of deep generative models for synthetic sample generation. Some of the models are explicit with an intractable likelihood and inference. Some models are only approximate and generate blurry samples. Other methods do not sample in parallel, are complex and rely on Markov chains which are time-consuming. GANs are attractive as they do not make any explicit density estimation and they remedy most of these issues. GANs have generated extremely good examples in many domains. Section 2.5 reviews these GAN applications.

2.5 Applications of GANs

GANs are powerful generative models which generate realistic-looking samples with a random vector z. We do not to know the format and the structure of the model nor do we make any mathematical assumptions. This allows GANs to be widely applied in many areas. This section reviews these varied GAN applications, with a particular focus on how GANs may be extended for actuarial use.

Table 3 provides a summary of various areas where GANs have been applied. The most successful applications of GANs are in computer vision but there has been wider extensive applications in other domains. The details of these applications are introduced as follows.

Application	Method
Image translation	DiscoGAN [97], DualGAN [193], Pix2pix [195], CycleGAN [207]
Super resolution	SRGAN [106], Cycle-in-Cycle GAN [177], ESRGAN [190]
Image synthesis	BigGAN [23], ProGAN [94], ACGAN [143], SAGAN [200]
Object detection	MTGAN [9], Perceptual GAN [109], Segan [147]
Object transfiguration	GP-GAN [41], GeneGAN [205]
Joint image generation	Coupled GAN [117]
Video generation	VGAN [184], MoCoGAN [181], Pose-GAN [186]
Text to images	PSGAN [16], PS-GAN [90], TAC-GAN [152], Stack GAN [201]
Facial attributes	Face aging [4, 63, 77], StarGAN [30], DR-GAN [180], DCGAN [153]
Music generation	MuseGAN [44], ORGAN [71], SeqGAN [108, 197],
Text generation	RankGAN [115], C-RNN-GAN [133], RelGAN [140]
Speech conversion	VAW-GAN [86]
Speech enhancement	SpecGAN [22], WaveGAN [43], GANSynth [48], SEGAN [147]
Time series	RCGAN [49], Value-at-Risk [59], Stock prediction [206]
Information retrieval	IRGAN [187]
Domain adaptation	DANN [2], CyCADA [83], DualGAN [193]
Semi-supervised learning	Triple-GAN [31], [60], CatGAN [118], VAT [132]
Missing data imputation	HEXAGAN [87], MisGAN [110], VIGAN [169], GAIN [196]
Privacy preservation	Privacy-preserving GANs [10], Privbayes [202]
Anomaly detection	GANomaly [3], DOPING [113], AnoGAN [168]
Reinforcement Learning	GAIL [172]
Medical segmentation	SCAN [36], SegAN [191]
Fashion	[194], StyleGAN [93]
Art	GauGAN [146]
Medicine	Health records [7, 8], Drug discovery [15], DNA design [96]
e-Commerce	eCommerceGAN [104]
Data augmentation	DAGAN [5], BAGAN [126], GAMO [136], tableGAN [145]
Joint distribution	DiscoGAN [97], Coupled GAN [117], CycleGAN [207]
Continual learning	Deep generative replay [170]
Steganography	Steganographic GAN [183], CycleGAN [32, 207]

Table 3: Taxonomy of GANs applied in various topics

2.5.1 Image Generation

GANs have been highly successful for the generation of realistic images after being trained on samples images. For example, if we want to generate new images of cats, we can train a GAN on thousands of samples of samples of cats. Once the training has finished, the generator network will be able to generate new images that are different from the images in the training set.

The original GAN was formulate for this purpose but it had mode collapse, vanishing gradients and unstable training issues. Conditional GANs [129, 143] and DCGAN [153] significantly improve the vanilla GAN, thereby increasing sample quality and diversity. Since then, more GANs have been devised, with impressive results for image generation. Image generation finds applications in marketing, face aging, logo generation, entertainment, social media, anime character creation and so on. This finds immediate applications for short-term insurance and banking.

2.5.2 Image Translation

To convert an image content from one domain to another, an image-to-image translation approach was proposed by Yoo et al. [195] using cGANs, which is named pix2pix. Experiments have shown that pix2pix can be effective not only in graphics tasks but also in vision tasks. Pix2pix requires the training space to be strictly paired in the X and Y spaces. However, such paired data is hard to find.

Based on this situation, DiscoGAN [97], DualGAN [193] and CycleGAN [207] adopt the idea of cyclic consistency, which can use unpaired data to train the mapping from X space to Y space. Choi et al. [30] proposed a StarGAN, which can solve the problem of image translation among multi-domains by learning one model. The use of StarGAN in the tasks of facial expression synthesis and facial attribute transfer has had surprising effects. Figure 5 shows an example of CycleGAN application, converting different images to another.

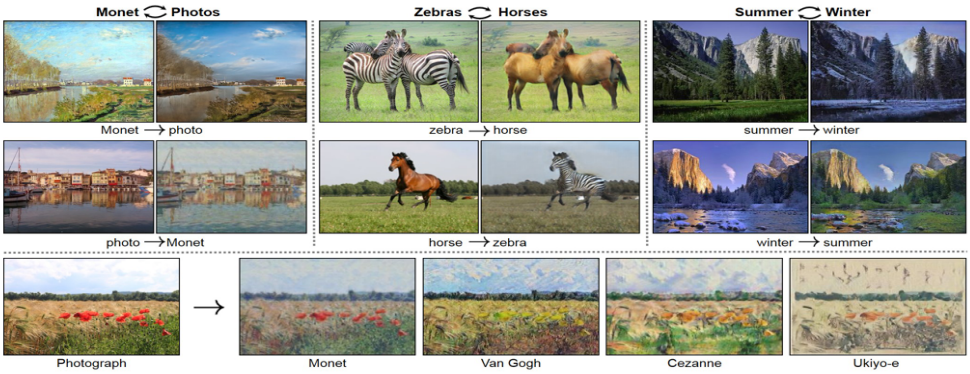


Figure 5: Images generated by CycleGAN from the original paper [207]

These GANs can be useful in short-term insurance domains such as in agriculture, weather patterns, satellite images, telematics, claims assessment and in medicine for medical imaging. CycleGANs can be used to generate even data sets that have no alignment, e.g. convert zebras to horses, winter to summer and so on.[3] In insurance, these GANs can be used to convert claims frequency data to claims amounts and vice versa, thereby transferring styles between different data sets.

Similarly, one can use DualGAN or DiscoGANs to understand profiles of different insurance lives by switching between claiming and non-claiming lives, lapsing and non-lapsing lives, mortality and morbidity etc., through learning one model which describes the two data spaces. Image translation is a very successful area of GANs and this task can enhance short-term insurance images.

2.5.3 Image Super Resolution and Synthesis

Image synthesis is an important direction since the invention of GANs. How to generate realistic image and face samples has always been the problem that needs to be addressed. GANs have been highly successful on image synthesis since the improvement of GANs using DCGAN [153] and cGAN [129]. Most structures are loosely based on DCGAN, especially for images.

[3] https://junyanz.github.io/CycleGAN/

A Super-Resolution Generative Adversarial Networks (SRGAN) [106] takes a low-resolution image as input, and generates a high-resolution image with 4x up-scaling. Wang et al. [190] proposed an Enhanced Super-Resolution Generative Adversarial Networks (ESRGAN) to improve the texture information generated by SRGAN, which was shown to not being real enough and accompanied by some noise. The ESRGAN achieved better performance than SRGAN. These GANs can be used to improve the resolution of any image data set.

ProGAN [94] improves the speed, stability and variation of GAN training. ProGAN progressively trains the generator and the discriminator on low-resolution images and then incrementally adding more layers throughout the training process to increase the resolution. ProGAN manages to generate full-high-definition photo-realistic images, with deep learning grandfather Yoshua Bengio lauding it as "too good to be true". An example practical application of ProGANs is the high-resolution mammogram synthesis for breast cancer screening conducted at Kheiron Medical Technologies in London [100, 101]. A paper entitled "GANs for Medical Image Analysis" [95] shows how these GANs are used in 62 other medical applications.

Self-Attention GAN (SAGAN) [200] allows attention-driven long-range dependency modelling for task generations. BigGAN is similar to SAGAN but has high scaling capabilities. BigGAN [23] and Style GAN [93] have since made tremendous advances in the quality of GANs. These are current state-of-the-art models with quite high quality results. Style GAN has been used to generate high-quality fashion model images wearing custom outfits [194]. For telematics and short-term insurance such as car, property and agricultural insurance, these GANs would have tremendous advantages for image synthesis. Other immediate applications are in banking for facial recognition.

2.5.4 Object Detection and Video Generation

GANs have been used for generating videos, video targeting and in video applications [40, 181, 184, 189]. Other applications include texture synthesis using Periodic Spatial GAN (PSGAN) [16], object detection using Segan [191] or Perceptual GAN [109], portrait drawing from face photos [180] and many more image and vision others. For object detection, Segan has immediate applications for semantic segmentation in driving conditions, thereby boosting car insurance claim predictability.

2.5.5 Sequences

GANs have tremendous achievements in natural language processing (NLP), music, speech, voice and time series. This section briefly touches on some of the proposed GANs for these tasks, the most interesting being time series generation for actuarial consideration.

2.5.5.1 Natural Language Processing In NLP, an information retrieval GAN [187] is proposed while text generation using RankGAN [115] and speech language processing using SEGAN [147] were also accomplished. GANs have also been used for music generation such as MuseGAN [44], Object-Reinforced GAN (ORGAN) [71], SeqGAN [108], continuous recurrrent neural network GAN (C-RNN-GAN) [133] and MidiNet [192]. Such applications have immediate applications for the entertainment industry such as movie theme songs, reducing costs and improving operational efficiency.

GANs have also been used for speech and audio analysis, such as synthesis [164], enhancement [147] and recognition [42]. GANs also learn text-to-image generation [39, 152] or image-to-text [28], too. Such applications can be used to convert an image taken

from a car insurance damage/scratch, to useful text description with ease.

2.5.5.2 Time Series The modeling and statistical distribution generation, time series and stochastic processes are widely used by financial firms for risk management, financial projections, stock prediction, extreme event monitoring and monetary policy making [59]. Traditionally, Autoregressive and Moving Average (ARMA), Exponential smoothing, Generalized Autoregressive Conditional Heteroscedasticity (GARCH), Vector Auto Regression (VAR) and their variants, and more recently Recurrent Neural Networks (RNN) [82], have been introduced and intensively studied and applied for time series data [18].

However, most of these models are reliant on strong dependence on model assumptions and model parameter estimation and, thus, are less effective in the estimation of complex distributions with time-varying features [206]. GANs do not make any explicit assumptions and are capable of learning the distributions and their dependence structures in a non-parametric fashion. A number of studies have demonstrated the use of GANs for time series prediction. Such models such as cGAN and Recurrent GANs are appealing for calculating Value-at-Risk and Expected Shortfall for market risk management [59], economic modelling [49], stock prediction [206] and others.

Suppose we wanted to simulate the evolution of a stock price for some particular asset using traditional simulations such as Monte Carlo. We would need to estimate the mean and volatility of the returns using past price evolution and then simulate new prices under the assumption that the returns follow a Gaussian distribution with the estimated parameters. However, this normality assumption may not be entirely true in practice where there is a tendency for higher observed probabilities for the tail events than those predicted by the Gaussian distribution. We could change our assumption, say into a student-t distribution, but neither would that assumption completely describe the reality. GANs are capable of replicating the price evolution without making any model assumptions.

An immediate actuarial use is stochastic simulations and financial projections for capital modelling, mortality projections, reserving, asset and liability management, solvency projection and other time series generation tasks. GANs can be used to replace Monte Carlo or stochastic simulations without the use of distributional assumptions.

2.5.6 Semi-Supervised Learning

The purpose of a Semi-Supervised GAN (SGAN) is to train the discriminator into a classifier which can achieve superior classification accuracy from as few labeled examples as possible [173], thereby reducing the dependency of classification tasks on enormous labeled data sets. It has been shown that an SGAN generalizes from a small number of training examples much better than a comparable, fully-supervised classifier [31, 118, 132]. This has been lauded as the most useful GAN application with good performance with a small number of labels on data sets [142, 166].

For imbalanced data sets such as mortality, morbidity, fraud, lapses, extreme events, large claims and sub-standard risks, SGAN may offer a superior alternative predictive model compared to ML models which require significant training data for improved accuracy. Typically, one has to deal with imbalanced classes either through synthetic sample generation using some heuristic method such as SMOTE [26], cost sensitive adjustment to the evaluation metric or adding uncertainty margins which can be subjective. Through the training of an SGAN, it is possible to have a sample generative scheme whilst having a classifier as well. This has tremendous advantages over many generative and ML models.

2.5.7 Data Augmentation

The availability of sufficient data in many domains is a necessity, especially where predictive models are needed to make business decisions. Such models are built on adequate training data for better generalization and meaningful accuracy [68]. Data augmentation is a procedure to create synthetic cases to augment the training data and increase its size, especially for those data points that are lacking. This is where GAN shines - the ability to create new samples and adequate data sets [53, 69].

There are two main strategies to check if this augmentation really helped something: we can train our model on fake data and check how well it performs on real samples. We can also train our model on real data to do some classification task and only after check how well it performs on generated data. If it works well in both cases — you can feel free to add samples from the generative model to your real data and retrain it again — you should expect gain of performance.

Recently, a number of papers have applied GANs to augment various data sets, with remarkable results on the performance of the predictive models applied after [5, 45, 53, 126, 136]. Similarly, GANs can be used to augment actuarial data sets and boost actuarial models, making them more accurate and less biased. In this work, we demonstrate how this can be done for a number of data sets, described in section 4.

2.5.8 Anomaly Detection

Anomaly detection is the identification of rare items, events or observations which raise suspicions by differing significantly from the majority of the data. Anomaly detection finds extensive use in a wide variety of applications such as fraud detection for credit cards, insurance or health care, intrusion detection for cyber-security and fault detection in safety critical systems.

The importance of anomaly detection is due to the fact that anomalies in data translate to significant (and often critical) actionable information in a wide variety of application domains. There are a number of these methods such as clustering-based, classification-task, nearest neighbor, spectral or statistical, but most of them have rather strong assumptions and long training times.

Main generative models like VAE or GAN consist of two parts. VAE has an encoder and the decoder, where the encoder basically models the distribution and the decoder reconstructs from it [105]. GAN consists of the generator and the discriminator, where the generator models the distribution and the discriminator judges if it's close to the training data [69]. They are pretty similar in some way — there is modeling and judging part (in VAE we can consider reconstructing as some kind of judgement).

The modeling part is supposed to learn the data distribution. What will happen to the judging part if we give it some sample not from the training distribution? In case of a well trained GAN, the discriminator will tell us 0, and reconstruction error of VAE will be higher than average one on the training data [3]. Our unsupervised anomaly detector is then easily trained and evaluated. We can feed it with some steroids like statistical distances if we want.

In medicine, Schlegl et al. [168] proposed an AnoGAN for anomaly detection of medical images, and learned the characteristics of lesions by learning the characteristics of health data sets. Figure 6 shows how AnoGAN works. Akcay, Atapour-Abarghouei, and Breckon [3] present GANomaly for anomaly detection in visual noise, noting a significant improvement on detecting anomalies on various data sets. These methods have potential applications in bio-medicine, fin-tech, video surveillance, network systems, fraud detection, lapse prediction and claiming likelihood in insurance etc. A

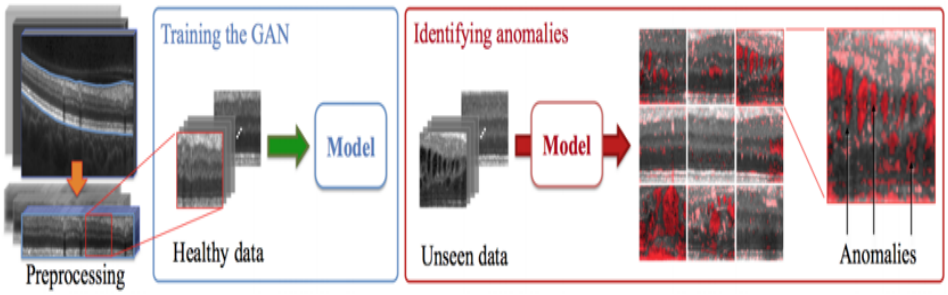


Figure 6: AnoGAN for medical images anomalies from the original paper [168]

GAN useful for anomaly detection can rival other anomaly detection techniques.

2.5.9 Privacy Preservation

Data of a lot of companies can be secret (like financial data that makes money), confidential or sensitive (medical data that contains patients diagnoses). Sometimes we need to share it with third parties like consultants or researchers. If we want to share a general idea about our data that includes the most important patterns, details and shapes of the objects, we can use GANs directly to sample examples of our data to share with other people without sharing identifiable features. This way we won't share any exact confidential data, just something what looks exactly like it.

Privacy-preservation GANs are capable of accomplishing this task [10]. In actuarial valuation models where model points are used to determine the amount of money to hold for an individual/groups, such GANs may be useful for the creation of synthetic samples to be fed into the valuation model, without needing the details of any policy.

2.5.10 Missing Data Imputation

Missing data causes an issue in analysis as most standard data analytic methods are not designed for missing data. Techniques such as single imputation (SI) and multiple imputation (MI) [24, 159, 175] exist but there is no consensus on which of the MI method is superior even though MI is known to be better SI [107, 159, 167].

Generative Adversarial Imputation Net (GAIN) [196] provides an alternative generative modelling approach to create new cases that can be used to impute missing information. MisGAN [110] also creates new imputed cases. VIGAN [169] deals with data that are collected from heterogeneous sources, resulting in multi-view or multimodal data sets where missing data occurs in a number of these sources. These methods were shown to be better than SI/MI methods, thereby improving the effectiveness of ML algorithms trained after. If you also have an image that has some missing parts, GANs can help you to recover these sections.

2.5.11 Domain Adaptation

It is quite possible that the training data used to learn a classifier has different distribution from the data which is used for testing. This results in degradation of the classifier performance and highlights the problem known as domain adaptation [84]. In domain

adaptation, the training and test data are from similar but different distributions. This area has become interesting for GANs in the past few years.

These methods include CoGAN, CycleGAN, DiscoGAN, DualGAN and StarGAN which can be used for multiple domains. With these methods, one can transfer an algorithm learned from a different data set to a new one and achieve similar performance. Such approaches are also able to learn representation adaptation, which is learning feature representations that a discriminator cannot differentiate which domain they belong to [182]. By using synthetic data and domain adaptation, the number of real-world examples that are needed to achieve a given level of performance is reduced significantly, utilising only randomly generated simulated points [83]. Domain adaptation can learn transfers between different domains, by synthesising different data sets. This can be useful in combing public data sets or other market data with internal company data in actuarial firms.

2.5.12 Joint Distributions

Coupled GAN (CoGAN) [117] was proposed to learn the joint distribution of two-domain images. This GAN architecture is composed of two GANs, each of which synthesises images in one domain. While CycleGAN, DiscoGAN and DualGAN focus on image-to-image translation using two domains, StarGAN translates to multiple domains using a single model.

These GANs allow the possibility of combining two different data sets into one. For example, one can combine mortality experience of limited underwritten lives with fully underwritten and still be able to learn those features peculiar to a particular product. This task may have advantages compared to correlation measures or copulas. This GAN direction will have tremendous applications for combining multiple data sets and from different domains [32, 117, 193].

2.5.13 Actuarial Science

Given the above taxonomy of GAN applications, we are interested in whether there is scope for GAN applications in actuarial science, by borrowing some of the architectures and applying them on actuarial disciplines. Figure 7 depicts actuarial areas where GANs can be useful.

To our knowledge, there has been limited applications of GANs in traditional actuarial areas such as insurance and health care. This is compounded by the fact that GANs have been highly successful on computer vision, with less emphasis on tabular data sets. However, there have been recent applications of GANs on other tabular data sets such as airline passengers [134] and medical records [7]. GANs can equally be adopted for similar tasks to boost limited data sets and improve actuarial models, especially in areas where models are needed to make business decisions. In particular, GANs could allow less reliance on using stochastic simulations that are based on subjective distributions and err less on margins used.

3 Methodology

This section describes in detail the theoretical operation of GANs, their challenges and tricks to improve their training. Throughout this paper, it is assumed that both GAN networks are implemented with ANNs. For comparative purposes, we also implement a popular synthetic data generative mechanism using Synthetic Minority Oversampling Technique (SMOTE) [26].

Actuarial Discipline	Example
Product Design, Development, Propensity & Customer Behaviour	Big data on consumer information; Create wider and more model points; Boost propensity models with more data; Leverage external data sources
Experience Analysis and Basis Setting/Pricing	Experience monitoring and experience rates derived using a large credible data set. Deep learning and high level decision making. Boost machine learning models for pricing models. Semi-supervised learning; Domain adaptation; Attention prediction; Anomaly detection;
Capital Modelling	Network modelling by looking at driving dependences rather than correlation assumptions i.e. use generative models; Strategic fleixble and more decision-based model based on the environment. More GAN-based time series models which make no assumptions and are driven by the environment
Exposure management	Build predictive models based on weather patterns i.e use computer vision. GANs are useful for image synthesis and domain adaptation/attention prediction.
Reserving/Valuation	Make projections more predictive through a large enough credible data at all model points i.e accurate assumptions with less margins
Surplus distribution	More granular individual information from alternative data sources through leveraging generative models. More sophisticated longevity models derived implicitly
Asset Allocation i.e. strategic/tactical	Use Cycle GANs, utilising alternative data e.g. text-heavy data, social media feeds, satellite images etc. Improvements to portfolio optimisation
Asset & Liability Management	More granular data for asset/liability modelling. Generative models can be used to simulate scenarios that depend entirely on adopted investment strategy and boosting the model. Enhanced market risk monitoring
Data Cleaning	Reduce errors; fill in gaps using imputation; increase the sample size using GANs; query other data sets and verify patterns using CycleGANs; Semi-supervised Learning
External Data sources	Leverage other data sets through combining multiple data sets. For example, CoupleGANs, DualGAN, DANN and CycleGANs can be used to learn a representation that encompasses different data sets. Domain adaptation is particularly interesting
Wider Fields	Spatial Statistics; Retail;Fashion; Gaming; Arts; Entertainment; Medicine; Sports; e-Commerce; Manufacturing; Engineering;

Figure 7: Potential GAN applications in actuarial disciplines

3.1 SMOTE

This section describes the theoretical operation of SMOTE, for comparative purposes with the GAN applied in this work.

Considering a random minority instance x, a new instance s is generated by considering its k-nearest neighbors (NNs). These k-NNs are found by using the Euclidean distance metric. Initially, an instance y is generated at random from the k-NNs. Then a new synthetic minority instance s is generated as follows:

$$s = x + \alpha\,(y - x) \tag{4}$$

where α is randomly generated from the Uniform distribution $[0, 1]$. SMOTE parameters are the value of k and the number of minority cases to generate. These can be tuned to ensure an optimal metric is achieved. The number of k-NNs can be varied such that an optimal metric is found, whilst restricting the number of generated instances to ensure a balanced class distribution.

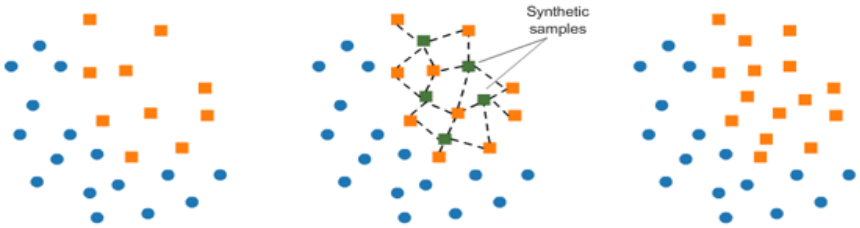


Figure 8: SMOTE operation

3.2 Vanilla GAN

This section describes the original GAN formulation, called MiniMax GAN (MM-GAN). This is the baseline model over which all other variants are based.

3.2.1 The Discriminator

The discriminator (D) receives generated samples from a generator G and the true data examples from $p_{data}(x)$, and must distinguish between the two for authenticity through a deep ANN [69]. The resulting output $D_{\theta_d}(x)$ for an input x is the probability of x being sampled from $p_{data}(x)$ instead of p_g, where p_g is the implicit distribution defined by G. The vector Θ_d represents learned parameters from D.

The discriminator's goal is to yield $D(x)$ near 1 for $x \sim p_{data}$ and $D(G(z))$ closer to 0 for $p_z(z)$ using the sigmoid function in the output layer. This is achieved by maximising D's loss over θ_d:

$$J_D^{MM-GAN} = E_{X\sim p_{data}(x)}\left[\log D_{\theta d}(x)\right] + E_{Z\sim p_z(z)}\left[\log(1 - D_{\theta d}(G_{\theta g}(z))\right] \tag{5}$$

3.2.2 The Generator

The generator (G) randomly picks a sample z from the prior latent space defined by $p(z)$ and then generates samples from this distribution using an ANN. This deep ANN must learn the parameters Θ_g given an input $z \sim p_z(z)$, that will give the output $G_{\theta g}(z)$. G is trained to fool D i.e. to make D's output for fake/generated sample $D(G(z))$ closer to 1. The parameters of G are learned by minimising G' loss over Θ_g:

$$J_G^{MM-GAN} = E_{Z\sim p_z(z)}\left[\log(1 - D_{\theta d}(G_{\theta g}(z))\right] \tag{6}$$

3.2.3 GAN Loss

Combining the losses for D and G, GANs solve the following minimax game in alternate steps through GD:

$$\min_{\theta_g}\max_{\theta_d} E_{X\sim p_{data}(x)}\left[\log D_{\theta_d}(x)\right] + E_{Z\sim p_z(z)}\left[\log(1 - D_{\theta_d}(G_{\theta g}(z))\right] \tag{7}$$

The above losses for D and G are the original formulation proposed by Goodfellow in 2014, called minimax GAN (MM-GAN). Since we are minimising over θ_g and maximising over θ_d, training of GANs alternate between GD on G and gradient ascent on D [68]. Typically, for every training of G, D is trained k times although an optimal choice is debatable among researchers. This is shown in Algorithm 1.

Remark 1. *Gradient based updates on can be accomplished using any of the GD optimisers reviewed earlier. Typically, SGD with Momentum for D, RMSProp or Adam for G tend to work well in practise [69, 153].*

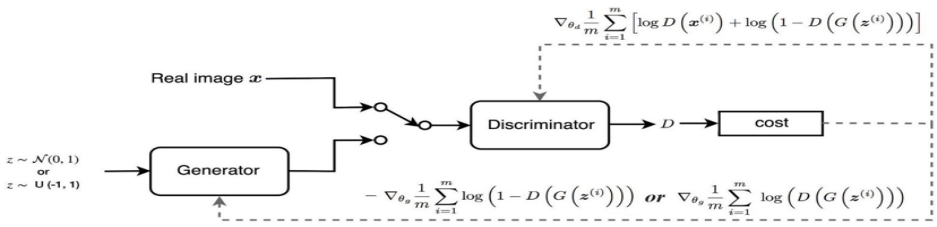


Figure 9: GAN training

3.2.4 Non-Saturating GAN

While the above loss function is useful for theoretical results, unfortunately it does not work well in practise and there are challenges getting the GAN to convergence, stabilise its training and getting diverse samples [6, 69, 129, 153, 166]. In practice, rather than training the above loss function for G, to provide better gradients in earlier training, Goodfellow et al. [69] suggest to maximise the following objective function for G instead:

$$J_G^{LS-GAN} = E_{Z \sim p_z(z)} \log \left(D_{\theta d}(G_{\theta g}(z)) \right) \tag{8}$$

This version of GAN is called non-saturating GAN (NS-GAN) and is typically used as the benchmark in most studies and in practise. This leads to the following NS-GAN loss function:

$$\max_{\theta_g} \max_{\theta_d} E_{X \sim p_{data}(x)} \left[\log D_{\theta_d}(x) \right] + E_{Z \sim p_z(z)} \log \left(D_{\theta d}(G_{\theta g}(z)) \right) \tag{9}$$

With this new loss function, we alternate between gradient ascent on D and gradient ascent on G. The algorithm presented below is based on the original MM-GAN formulation, however, it can easily be tweaked to represent NS-GAN.

3.2.5 Optimal Solution

Theoretically, it can be shown that for $p_g = p_{data}$, the GAN zero-sum game in Equation 9 has a global optima. Given enough capacity for both networks and D is trained to optimality for a fixed G, convergence of the GAN algorithm is guaranteed [69, 124, 129, 141, 153]. The optimal discriminator $D_G^*(x)$ for a fixed G is:

$$D_G^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \tag{10}$$

Assuming that D is perfectly trained and if we substitute $D_G^*(x)$ into Equation 9 for G's loss, this gives rise to the Jensen-Shannon (JS) divergence [114]. The JS divergence can be written as a function of the Kullback-Leibler (KL) divergence [102, 103].

Definition 1. *The KL divergence between two probability distributions p_{data} and p_g is defined as*

$$KL(p_{data}, p_g) = D_{KL}\left(p_{data} || p_g\right) = \int p_{data}(x) \log \left(\frac{p_{data}(x)}{p_g(x)} \right) dx$$

Algorithm 1 Mini-batch SG ascent of GANs with the original objective for MM-GAN. The number of steps to apply to D, k, is a hyper-parameter. For every training of G, we train D k times. Goodfellow et al. [69] used $k = 1$.

1: **for** *number of epochs* **do**
2: **update the discriminator**
3: **for** k *steps* **do**
4:
- Sample mini-batch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from the noise prior $p_g(z)$.
- Sample mini-batch of m true examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from the training data distribution $p_{data}(x)$.
- Update the discriminator D by ascending its stochastic gradient on these mini-batches:

$$\Delta_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(x^i\right) + \log\left(1 - D\left(G(z^i)\right)\right)\right].$$

5: **end for**
6: **update the generator**
7:
- Sample mini-batch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from the noise prior $p_g(z)$.
- Update the generator by descending its stochastic gradient computed on this mini-batch:

$$\Delta_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G(z^i)\right)\right).$$

8: **end for**

Definition 2. *The JS divergence between two probability distributions p_{data} and p_g is defined as*

$$JS(p_{data}, p_g) = D_{JS}\left(p_{data} || p_g\right) = \frac{1}{2} KL\left(p_{data}, \frac{p_{data} + p_g}{2}\right) + \frac{1}{2} KL\left(p_g, \frac{p_{data} + p_g}{2}\right)$$

If we substitute $D^*_G(x)$ into Equation 9, the minimum loss for G is reached if and only if $p_g = p_{data}$, thus one can show that:

$$J_G = -\log 4 + 2JS\left(p_{data}, p_g\right) \tag{11}$$

This equation tells us that when D has no capacity limitation and is optimal, the GAN loss function measures the similarity between p_{data} and p_g using JS divergence. However, although the above results provide a nice theoretical result, in practise, D is rarely ever fully optimal when optimising G [69]. Thus alternative GAN architectures have been proposed to fix this issue and to get closer to optimality. Below we describe what causes this failure to convergence and how to fix it.

3.3 Challenges with GANs

GANs are notoriously difficult to train properly, difficult to evaluate, the likelihood cannot be easily be computed, suffer from the vanishing gradient problem, mode collapse, boundary distortion and over-fitting [6, 35, 68, 81, 85, 112, 166]. This section describes key challenges on GAN training.

3.3.1 Mode collapse

Mode collapse is when many latent noise values z are mapped to the same data point x, leading to a lack of diversity in the samples that are created i.e. under-fitting. This is regarded as the most significant problem with GANs [112, 124]. Many studies have spent lots of time in varied contexts to fix this.

3.3.2 Vanishing gradient

This occurs when D becomes perfect in its training without giving G the chance to improve. As a result, GANs may fail to converge and thereby leading to poor generated samples [6].

3.4 Improved GAN Training

There are many GAN architectures which avoid the problems that come with the vanilla GAN. We briefly describe some of the most common and popular GAN solutions. Given the vast number of taxonomies, we are not able to cover all of them but only discuss the most popular and those subsequently used in this work.

3.4.1 Conditional GANs

The first extension of GAN was the Conditional GAN (cGAN) which gave the generator the label Y in the latent space, making them class conditional [29, 129, 143]. Most of the GAN variants can be modified to include cGAN. cGAN allows to create diversified samples and forcing G to create specific samples and thereby fixing mode collapse problem. There are other conditional GANs such as Auxiliary Classifier GAN [143] and InfoGAN [29] for various tasks. These versions have been highly useful in many domains such as image synthesis and face aging.

3.4.2 Deep Convolutional GAN

Given that DCGANs use CNNs which are typically for images, we do not review this architecture in detail as our main focus in on tabular data. However, the following can be noted from the DCGAN paper: (1) Use Adam optimiser instead of SGD with Momentum as it seems to outperform other GD optimisers in practice; (2) Where possible, use batch normalisation in most hidden layers of both networks and (3) Use leaky ReLU for D and ReLU for D in the hidden layers. This process tends to lead to better speed, stable training and better performance of GANs [153]. As a result, most GAN variants have the structure of a DCGAN or use these recommendations one way or another.

3.4.3 Loss Variants

There are a number of GAN architectures which change the loss function to improve GAN training and stability. The loss function for GAN measures the similarity between p_{data} and p_g using JS. Unfortunately, JS tends not to be smooth enough to ensure a stable training [85, 124]. There are a number of GAN loss variants which have been proposed over the years. Broadly, there are two loss function groups with better properties i.e. f-divergence [141] and IPM [135]. Figure 10 shows some of these loss variants.

Among these loss groups, WGAN is arguably the most popular and well-studied [81, 85, 188]. WGAN is considered a general unified framework under the recently proposed Relativistic GAN (RGAN) [92]. Thus we adopt to describe WGAN as it has become the most widely used GAN architecture since DCGANs.

3.5 WGAN

This section describes WGAN and its improved training using WGAN-GP.

Name	Value Function
GAN	$L_D^{GAN} = E[\log(D(x))] + E[\log(1 - D(G(z)))]$ $L_G^{GAN} = E[\log(D(G(z)))]$
LSGAN	$L_D^{LSGAN} = E[(D(x) - 1)^2] + E[D(G(z))^2]$ $L_G^{LSGAN} = E[(D(G(z)) - 1)^2]$
WGAN	$L_D^{WGAN} = E[D(x)] - E[D(G(z))]$ $L_G^{WGAN} = E[D(G(z))]$ $W_D \leftarrow clip\_by\_value(W_D, -0.01, 0.01)$
WGAN_GP	$L_D^{WGAN\_GP} = L_D^{WGAN} + \lambda E[(\lvert\nabla D(\alpha x - (1 - \alpha G(z)))\rvert - 1)^2]$ $L_G^{WGAN\_GP} = L_G^{WGAN}$
DRAGAN	$L_D^{DRAGAN} = L_D^{GAN} + \lambda E[(\lvert\nabla D(\alpha x - (1 - \alpha x_p))\rvert - 1)^2]$ $L_G^{DRAGAN} = L_G^{GAN}$
CGAN	$L_D^{CGAN} = E[\log(D(x, c))] + E[\log(1 - D(G(z), c))]$ $L_G^{CGAN} = E[\log(D(G(z), c))]$
infoGAN	$L_{D,Q}^{infoGAN} = L_D^{GAN} - \lambda L_I(c, c')$ $L_G^{infoGAN} = L_G^{GAN} - \lambda L_I(c, c')$
ACGAN	$L_{D,Q}^{ACGAN} = L_D^{GAN} + E[P(class = c\lvert x)] + E[P(class = c\lvert G(z))]$ $L_G^{ACGAN} = L_G^{GAN} + E[P(class = c\lvert G(z))]$
EBGAN	$L_D^{EBGAN} = D_{AE}(x) + \max(0, m - D_{AE}(G(z)))$ $L_G^{EBGAN} = D_{AE}(G(z)) + \lambda \cdot PT$
BEGAN	$L_D^{BEGAN} = D_{AE}(x) - k_t D_{AE}(G(z))$ $L_G^{BEGAN} = D_{AE}(G(z))$ $k_{t+1} = k_t + \lambda(\gamma D_{AE}(x) - D_{AE}(G(z)))$

Figure 10: GAN loss variants

3.5.1 Wasserstein Distance

IPM generalises a critic function f belonging to an arbitrary function class where IPM measures the maximal distance between two distributions under some functional frame f [81]. Among the IPMs, the Wasserstein distance is the most common and widely used metric [124]. Informally, the Earth mover (EM) [160] distance $W(p_{data}, p_g)$ measures the minimal changes needed to transform p_g into p_{data}. More formally, EM between two probability distributions p_{data} and p_g is:

$$W\left(p_{data}, p_g\right) = \inf_{\gamma \sim \Pi(p_{data}, p_g)} E_{(x,y)\sim_\gamma}\left[\| x - y \|\right] \tag{12}$$

where $\Pi_{(p_{data}, p_g)}$ represents a set of all joint probability distributions whose marginal distributions are respectively $p_{data}(x)$ and $p_g(x)$. Precisely, $\gamma(x, y)$ is a transport plan

i.e. percentage of mass that should be moved from x to y to transform p_g into p_{data}. The infimum in Equation 12 is intractable as it is tricky to exhaust all the elements of $\Pi_{(p_{data},p_g)}$ [6]. This is solved using the following functional format:

$$W\left(p_{data},p_g\right)=\sup_{\|f\|_L\leq 1} E_{x\sim pdata}\left[f(x)\right]-E_{x\sim pg}\left[f(x)\right] \tag{13}$$

where the supremum is taken over a 1-Lipschitz function f. A function f is 1-Lipschitz if for all $x_1, x_2 : |f(x_1) - f(x_2)| \leq |x_1 - x_2|$.

3.5.2 The Critic

In WGAN, D's output is not a probability anymore but can instead be any number and for this reason, D is typically called the *critic*. The WGAN critic tries to maximise the difference between its predictions for real samples and generated samples, with real samples scoring higher. Arjovsky, Chintala, and Bottou [6] force the critic to be *1-Lipschitz continuous* for the loss function to work well:

$$J_{WGAN}=\max_{w\in W} E_{X\sim p_{data}(x)}\left[D(x)\right]+E_{Z\sim p_z(z)}\left[1-D(G(z))\right] \tag{14}$$

where W is the set of 1-Lipschitz continuous functions. Typically, to enforce the Lipschitz constraint, the critic weights w are clipped to lie within a small range, usually $[-0.01, 0.01]$ after each training batch [6, 72].

The critic is trained to convergence so that the gradients of G are accurate, thus removing the need to balance the training of G and D by simply training D several times between G's updates, to ensure it is close to convergence. Typically, 5 critic updates to 1 generator update is used [6]. The WGAN training algorithm is shown in Algorithm 2 as per the original paper [6]. WGAN used the RMSProp version of gradient GD with a small learning rate and no momentum [6]. However, Adam may also be used as it is a combination of RMSProp with Momentum.

3.6 Improved WGAN Training

Even though WGAN has been shown to stabilise GAN training, it is not generalized for deeper training due to weight clipping which tends to localise most parameters at -0.01 and 0.01 [72, 124]. This effect dramatically reduces the modelling capacity for D. Gulrajani et al. [72] further amends WGAN through an addition of a gradient-penalty (GP) to the loss function, coming with WGAN-GP. In total, three changes are made to WGAN critic to convert it to WGAN-GP: include a GP to the loss function; do not clip critic weights; and do not use batch normalisation layers in the critic. WGAN-GP defined using the following loss function:

$$E_{X\sim p_{data}(x)}\left[D(x)\right]+E_{Z\sim p_z(z)}\left[1-D(G(z))\right]+\lambda E_{\tilde{x}\sim p_{data}}\left[\left(\|\ \Delta D(\tilde{x})\ \|_2-1\right)^2\right] \tag{15}$$

where $\tilde{x}$ samples uniformly along the straight line between points sampled from p_{data} and p_g and λ is the GP term. Gulrajani et al. [72] show a better distribution of learned parameters compared to WGAN and this method has been the default method in most GAN loss variants.

We adopt the conditional version of WGAN-GP, called WCGAN-GP, as an alternative to current actuarial/statistical approaches for synthetic sample generation. Once WGAN-GP is trained to convergence, G can be used to create new samples by feeding it the latent space Z.

Algorithm 2 Wasserstein GAN. Default experiments used $\eta = 0.00005$, $c = 0.01$, $m = 64$ and $n_{critic} = 5$.

Require: η, the clipping parameter c, the batch size m, the number of iterations of the critic per generator iteration n_d.
Require: initial critic parameters w_0, initial parameters of the generator Θ_0
while *θ has not converged* **do**
for $t = 0, \cdots, n_{critic}$ **do**
Sample mini-batch of m noise samples $\{z^{(1)}, \ldots, z^{(m)}\}$ from the noise prior $p_g(z)$.
Sample mini-batch of m true examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from the training data distribution $p_{data}(x)$.
$g_w \leftarrow \Delta_w \left[\frac{1}{m} \sum_{i=1}^{m} f_w \left(x^{(i)} \right) - \frac{1}{m} \sum_{i=1}^{m} f_w \left(g_\theta \left(z^{(i)} \right) \right) \right]$
$w \leftarrow w + \eta.RMSProp\left(w, g_w\right)$
$w \leftarrow clip\left(w, -c, c\right)$
end for
Sample mini-batch of m true examples $\{x^{(1)}, \ldots, x^{(m)}\}$ from the training data distribution $p_{data}(x)$.
$g_\theta \leftarrow -\Delta_\theta \frac{1}{m} \sum_{i=1}^{m} f_w \left(g_\theta(z^{(i)}) \right)$
$\theta \leftarrow \theta + \eta.RMSProp\left(\Theta, g_\theta\right)$
end while

4 Experiments

This section outlines the experiments conducted, showing a popular GAN application for data augmentation and boosting predictive models. We compare WGAN with a popular synthetic data generation mechanism i.e. SMOTE [26]. This exercise can be similarly adopted for any actuarial modelling problem such as mortality, morbidity, medical segmentation, credit risk, extreme events, regression, Value-at-Risk, and anomaly detection in insurance, banking, investment, banking and health care.

4.1 Data Sets

We considered 5 publicly available imbalanced data sets from the Machine Learning Repository UCI. The data sets are described below and shown in Table 4.

4.1.1 Credit Card Fraud

European public credit card fraud transactions made in 2013 are utilised [37]. This data is highly imbalanced, with 492 fraudulent transactions out of a total of 284, 807 transactions,representing a mere 0.172% of fraud cases. This data set contains 31 anonymised features *(Time, Amount, V0,V1,...V28)* and the Class indicator showing 1 for frauds and 0 for non-fraudulent cases. All the variables are numeric.

4.1.2 Pima Indians Diabetes

This data set contains the prediction of the onset of diabetes within 5 years in Pima Indians given some medical details, representing 34.90% of diabetic cases out of a total of 768 samples [171]. There are 8 independent variables.

4.1.3 German Credit Scoring

This data comes from the German credit scoring from the UCL Machine Learning Repository. There are 1000 observations with 20 independent variables. The dependent variable is the evaluation of customer's current credit status which indicates whether a borrower's credit risk is good or bad.

4.1.4 Breast Cancer Wisconsin

This data represents the characteristics of a cell nuclei that is present in the digitised image of a breast mass [176]. The data is used to predict the presence of benign or malignant cancer, with 37.25% being malignant samples from a total of 569 cases.

4.1.5 Glass Identification

This data set determines whether the glass type is float or not in term of their oxide content [51]. There are 32.71% of float glass types out of a total of 214 cases.

Data set	Majority cases	Minority cases	Number of features
Credit Card Fraud	284,807	492	31
Pima Indians Diabetes	500	268	8
Glass Identification	144	70	9
German Credit Scoring	700	300	20
Breast Cancer Wisconsin	357	212	9

Table 4: Data sets used in the experiments

4.2 Scaling the data

Many ML methods expect data to be of the same scale to avoid the dominance of certain variables and this can affect the accuracy of specific models [88, 131]. Normalisation re-scales the data to the range between 0 and 1. Standardisation centers the data distribution to $N(0, 1)$. We adopt normalisation as it does not assume any specific distribution. This will potentially speed up convergence [68, 131].

4.3 Train-Test Split

ML models are usually trained and tested on unseen data. Two approaches to split the data are cross-validation (CV) and train-test split [57]. CV divides the data into K subsets that can lack sufficient credibility and can result in higher variability of predictions, if the data size is too small [57]. Train-test split, however, can allow a larger subset of the data to be used for estimating model coefficients and results in more reasonable results [131].

Existing literature typically uses a 70%-90% train-test split, especially if the data is large. This technique is simple, easy to understand and widely used, despite giving noisy estimates sometimes [57, 68, 131]. CV is typically used to optimise parameters of a classifier. This work adopts 75% training data and 25% testing data.

4.4 SMOTE Implementation

Over-sampling is performed on the 75% training data using the R *imbalance* library [34]. The R imbalance library contains functions for performing SMOTE and other variants. The two parameters to tune are the number of neighbors and the over-sampling rate. We used a default of 5 K-NNs for SMOTE [26]. We kept the over-sampling rate the

same to ensure balanced class distributions within each data set. Using SMOTE, we create additional synthetic cases to supplement the above data sets.

4.5 GAN Implementation

Given its popularity and wide use, WGAN is adopted for an alternative synthetic sample generation. Specifically, we adopt the conditional version of WGAN with GP, thus we use WCGAN-GP [72, 129]. Below we describe how parameters are chosen and results generated.

4.5.1 Software

GANs can be implemented in a number of open-source neural-network libraries in Python [54]. Due to its simplicity and faster computations, the high-level Keras library [55] with Tensorflow [178] back-end is chosen to implement WCGAN-GP. This is trained using all minority cases of each data set.

4.5.2 The Generator

This section describes how the parameters for G are chosen.

4.5.2.1 Latent Noise The random noise for z is generated from $N(0, 1)$ with 100 dimensions. This is based from GAN hacks which suggest to sample from a spherical distribution [166].

4.5.2.2 Activation Function ReLU is adopted in the hidden layers [153, 166]. For G's output later, tanh is adopted. No drop out or batch normalisation is applied following advise from Gulrajani et al. [72] for WGAN-GP.

4.5.2.3 Layers The layers are chosen such that they are ordered in an ascending manner for G. For simplicity, after a number of iterations, 3 layers were chosen for each data set. In the first layer, there were 128 units, in the second layer 256 units and in the third layer 512 units. These layers worked well in the experiments conducted. The output layer had the data dimension of the data as the number of units.

Weights are initialised using the He initialisation method and ReLU is adopted [76]. Adam is used to optimise the weights of G [153, 166]. We used default values with $\beta_1 = 0.5$ and $\beta_2 = 0.9$ for G [98]. We used a batch size of 128 when optimising the gradients for faster training [88]. Initial η for G was fixed at 0.00004. The number of epochs were found to be $5,000$ where the GAN training was found to be stable.

4.5.3 The Critic

Leaky ReLu is adopted with a negative slope of 0.2 [121, 153]. As per the generator, 3 layers were used in the hidden layers. The layers were arranged in a descending manner, with 512 units in the first layer, 256 units in the second layer and 128 units for the last layer. The critic gives the output a single value using a linear function [6]. Adam was used with default parameters in Keras [55] as follows:

Parameter	Value
η	0.00001
β_1	0.5
β_2	0.90
ε	10^{-8}

Table 5: Adam parameters for the critic

Critic weights were also initialised using the He method and a similar batch size as in the generator was used. We pre-trained the critic 100 times at each adversarial training step [6]. This ensures faster convergence at each step before G is updated. We used WGAN with a GP with the default values as per the original paper [72]. The GP value was left unchanged at 10. We call this model WGAN-GP. We found that after 5000 epochs, the losses plateaued and did not change much.

4.5.3.1 Labels Typically, to boost faster training and fix mode collapse, additional information can be incorporated in both G and D using cGAN [129]. We used the conditional version of WGAN-GP where class labels were added to the minority cases. To accomplish this, clustering was done on the minority cases in order to induce class labels on the training data.

We explored a number of common mechanisms considering k-means, Agglomerative Hierarchical Clustering (AHC) [185], Hierarchical DBSCAN [50] and t-distributed Stochastic Neighborhood Embedding (t-SNE) [122]. The details of these algorithms are beyond the scope of this work. Due to its wide use and simplicity, we adopted k-means clustering with 2 clusters for each data set. This yielded labels that could be fed into G and D to induce generated samples. We call the final model WCGAN-GP after incorporating these class labels into the training.

4.5.4 Training WGAN-GP

Figure 11 presents the experiments of training WCGAN with GP. For comparative purposes, using similar parameters, we show the quality of samples generated for WCGAN[4] with GP, WGAN, cGAN and non-saturating GAN on the credit card fraud data. We consider this for two combinations of the features for illustrative purposes up to 5000 epochs.

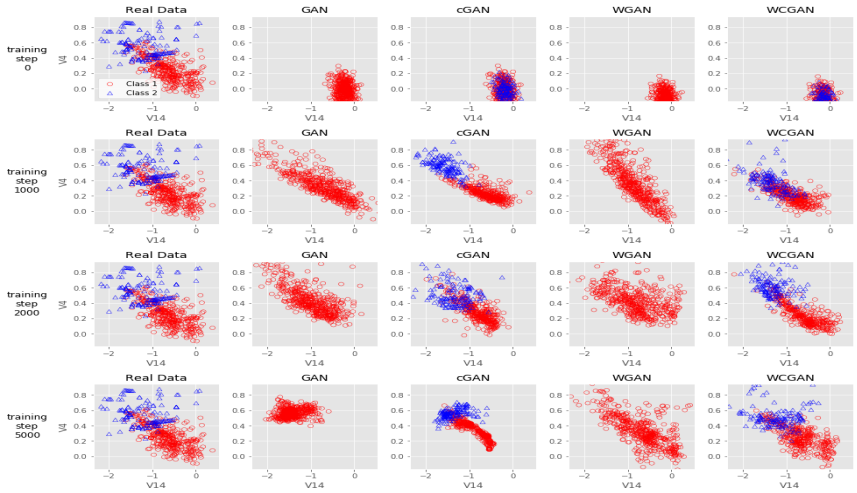


Figure 11: Comparison of GAN experiments ran on fraud data cases

The results show the superiority of samples generated by WCGAN with GP. There is a clear mode collapse problem on the vanilla GAN and cGAN. WGAN and WCGAN with GP show better samples. There is also clear damped oscillations and unstable

[4]The version of the WCGAN was incorporated with an improved WGAN training using the GP term as per the paper by Gulrajani et al. [72].

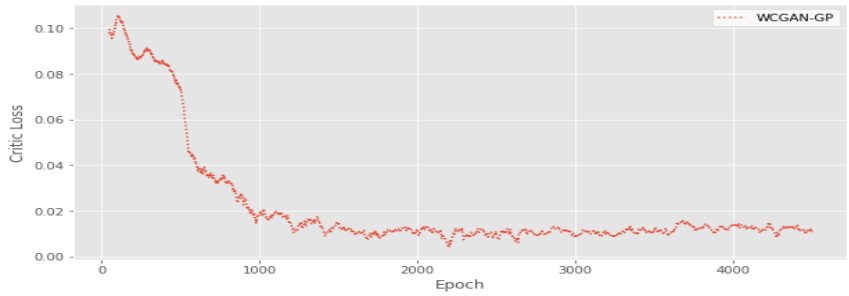


Figure 12: Difference between generated and real data critic loss

losses for GAN and cGAN where Wasserstein GANs exhibit stable training and losses, especially after 1000 iterations where it seems to settle and stabilise. Figure 12 shows the critic loss for each epoch, where after 1000 epochs, the loss starts to plateau. Thus we decided to stop the training after 5000 epochs. We repeated this experiment for each data set and adopted WCGAN with GP after $5,000$ epochs as the model to use for synthetic sample generation.

4.5.5 Generating Synthetic samples

Once the WCGAN with GP is trained to 5000 steps, the learned generator distribution is used to create more synthetic samples by feeding it the number of samples to output.

4.6 Logistic Regression

For simplicity and given the wide use with actuaries, a Logistic Regression (LR) [128] model is trained using Python 3.7 [54] on both the imbalanced training data and over-sampled data sets to predict the likelihood of each minority case using this equation:

$$\log\left(\frac{h_\theta(x^{(d)})}{1-h_\theta(x^{(d)})}\right) = \theta_0 + \sum_{i=1}^{d}\theta_i X_i,\ 0 < h_\theta(x^{(d)}) < 1 \tag{16}$$

where $h_\theta(x^{(d)})$ is the probability of the given minority case, θ_i's are the estimated coefficients using SGD, X_i is the feature vector for sample i and d is the number of features to include in the LR model. The coefficients are estimated by minimising a loss function through SGD in Equation 1. Typically, classification is such that when $h_\theta(x^{(d)}) \geq 50\%$ for each instance, assign the minority case, otherwise the majority case.

4.7 Evaluation

The confusion matrix returns a report showing how predicted classes on unseen test data using the LR model compare to actual observed classes, as depicted in Table 6.

Confusion Matrix	**Predicted: Minority**	**Predicted: Majority**
Actual: Minority	True Positive (TP)	False Negative (FN)
Actual: Majority	False Positive (FP)	True Negative (TN)

Table 6: The confusion matrix

TN is the number of majority cases that were correctly classified as such. **FP** is the number of majority cases that were incorrectly classified as minority. **TP** is the number

of minority cases that were correctly classified as minority. **FN** is the number of minority cases that were incorrectly classified as majority. Using these definitions, Table 7 presents the most well known evaluation metrics for binary problems.

Metric	Formula
Accuracy	$\left(\frac{TP+TN}{TP+TN+FP+FN}\right)$
Precision	$\left(\frac{TP}{TP+FP}\right)$
Recall	$\left(\frac{TP}{TP+FN}\right)$
F1-Score	$2 * \left(\frac{Precision*Recall}{Precision+Recall}\right)$

Table 7: Evaluation metrics for binary problems

Precision is the ability of the LR model not to label a minority case that is actually majority. Recall is the ability of the LR model to find all minority cases. F1-Score is a harmonic mean between Precision and Recall [75]. F1-Score puts equal weight to both Precision and Recall. Accuracy can be misleading and inappropriate when there are imbalanced classes and thus may be biased towards majority cases [26, 61, 75]. Thus we do not use rely on it in this work. Accuracy, Precision, Recall and F1-Score should be close to 100% for a LR model to do well on the testing data. However, these scores are influenced by what threshold is used to decide between the two binary classes.

The Receiver Operating Characteristic (ROC) curve [19, 74] measures a classifier's performance on a test set over different decision thresholds by varying the Precision and the FP rate. The Area under the Curve (AUC) measures the performance of the LR model trained on both imbalanced and over-sampled data sets and tested on unseen data with values close to 100% considered excellent performance [11, 74]. We also compute the Precision-Recall curve and compute the Area Under the Precision-Recall Curve (AUPRC) to get a weighted score. A method that gives the highest score is better.

4.8 Statistical Hypothesis Testing

Friedman test [58] followed by a post-hoc Nemenyi test [138] are performed to verify the statistical significant differences between WCGAN-GP and SMOTE.

4.8.1 Friedman test

The Friedman test is a non-parametric ranking test to determine whether SMOTE and WCGAN-GP methods perform similarly in mean performance rankings based on the measures above, when normality does not hold [58].

4.8.2 Post-hoc Nemenyi test

If the null hypothesis is rejected, a post-hoc test can be applied where WCGAN-GP is considered as the control method. The post-hoc Nemenyi test evaluates pairwise comparisons between the over-sampling methods if the Friedman test suggests that there is a difference in performance [138, 149]. We adopt WCGAN-GP as the control method.

4.8.3 Implementation

Both tests are conducted using the Pairwise Multiple Comparison Ranks Package (PMCMR) [149] available in R. We assume statistical significance of the alternative hypothesis at p-values < 0.05. In other words, we fail to reject the null hypothesis when the

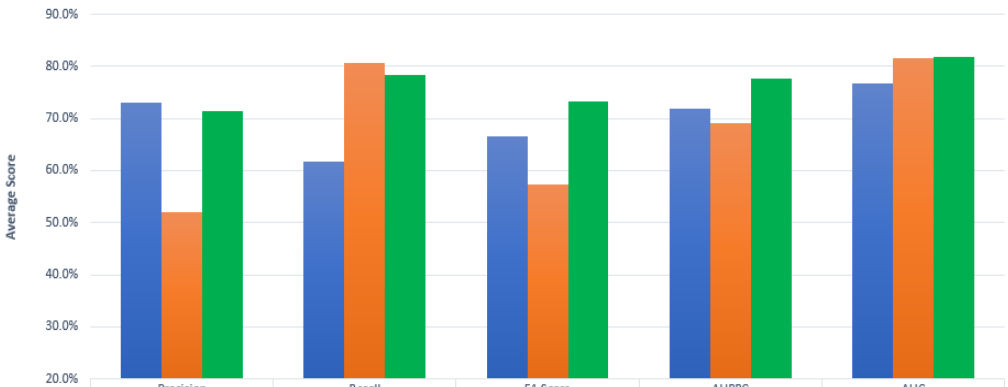
resulting p-value is higher than 0.05, suggesting that there is no difference between SMOTE and WCGAN-GP.

5 Results and Discussion

This section presents the results of all the LR models applied on the baseline and over-sampled data sets, with metrics on Precision, Recall, F1-Score, AUC and AUPRC computed on the same unseen test data.

5.1 Comparisons

Table 8 presents the evaluation metrics (based on the testing set) of the LR model applied on the baseline and over-sampled data sets for a default threshold of 50%. Bold shows an algorithm that performs the best for that data set i.e. a higher score for that metric. Figure 13 shows the average performance across all data sets from each evaluation metric.



	Precision	Recall	F1-Score	AUPRC	AUC
Baseline	73.0%	61.7%	66.6%	71.9%	76.7%
SMOTE	52.0%	80.8%	57.4%	69.0%	81.6%
WCGAN-GP	71.5%	78.4%	73.4%	77.7%	81.9%

Figure 13: Average performance across all data sets

In general, SMOTE improves Recall at the expense of a lower Precision. This results in a lower F1-Score than Baseline results. As a result of a much lower Precision for SMOTE, AUPRC is penalised and lower than both Baseline and WCGAN-GP. SMOTE compromises the Precision significantly, whereas WCGAN-GP improves Recall while not significantly penalising Precision.

Overall, WCGAN-GP shows a higher F1-Score. Thus using a default threshold, WCGAN-GP performs the best on F1-Score, followed by Baseline and SMOTE being last (on the average). The lower Precision on SMOTE may be due to the strict assumed probability distributions and possible creation of over-lapping and noisy samples [13, 38, 62, 127, 199]. While the univariate results on Precision, Recall and F1-Score are useful, they do not give the entire picture over different thresholds [11].

Since AUC and AUPRC are based on varied thresholds, these metrics are typically preferred over one dimension measurements such as Precision, Recall and F1-Score [11, 61, 119]. Since we are also comparing the above results with the Baseline model, these metrics are impacted by class imbalance [75]. Thus we rely on the AUC and AUPRC.

Method	Precision	Recall	F1-Score	AUPRC	AUC
Credit Card Fraud					
Baseline	85.71%	63.41%	72.90%	74.60%	81.70%
SMOTE	5.70%	**90.24**%	10.71%	47.98%	**93.83**%
WCGAN-GP	**86.24**%	76.42%	**81.03**%	**81.35**%	88.20%
Pima Indians Diabetes					
Baseline	74.47%	56.45%	64.22%	72.49%	73.61%
SMOTE	62.86%	**70.97**%	66.67%	71.6%	**75.48**%
WCGAN-GP	**75.51**%	59.68%	**66.67**%	**74.10**%	75.22%
German Credit Scoring					
Baseline	**60.31**%	51.34%	55.47%	63.02%	68.57%
SMOTE	54.22%	60.80%	57.32%	63.31%	69.61%
WCGAN-GP	46.51%	**81.08**%	**59.11**%	**66.60**%	**70.94**%
Glass Identification					
Baseline	50.00%	42.86%	46.15%	53.83%	63.93%
SMOTE	42.86%	**85.71**%	57.14%	66.14%	72.86%
WCGAN-GP	**55.00**%	78.57%	**64.71**%	**69.56**%	**78.03**%
Breast Cancer Wisconsin					
Baseline	94.34%	94.34%	94.34%	95.39%	95.50%
SMOTE	94.44%	96.23%	95.33%	96.03%	96.45%
WCGAN-GP	**96.23**%	**96.23**%	**96.23**%	**96.93**%	**97.00**%

Table 8: Evaluation metrics based on a default threshold of 50%

5.1.1 AUC

The ROC curve represents the trade-off between Precision and the FP rate while the AUC is the area under the ROC curve [11]. SMOTE reports higher AUC values than the Baseline. In general, WCGAN-GP is better on 3 of the 5 data sets except on Credit card fraud and Diabetes data sets. Overall, the average AUC value is not too different between WCGAN-GP and SMOTE. This result conflicts the AUPRC scores where WCGAN-GP shows a clear dominant superiority over SMOTE.

Whilst AUC may be useful, it does not consider Recall, which may be the most important metric for minority cases. AUC may be affected by skewed data sets and the data distribution [75]. ROC curves are appropriate when the data is balanced, whereas Precision-Recall curves are appropriate for imbalanced data sets [11, 75]. AUC may tend to provide an overly optimistic view than AUPRC [75].

In general, an algorithm that dominates in AUC may not necessarily dominate the AUPRC space [75]. Saito and Rehmsmeier [163] suggest that the Precision-Recall curve and AUPRC are more informative than the ROC curve and AUC. Since we are also comparing with the Baseline which is imbalanced, ROC and AUC may be inappropriate, thus AUPRC provides a sensible measure for all methods.

5.1.2 AUPRC

AUPRC has all the characteristics of the AUC and thus for the purposes of this work, we rely more on AUPRC than AUC [75, 163]. Overall, WCGAN-GP shows better improvements over SMOTE. WCGAN-GP is highest on AUPRC, suggesting this algorithm performs the best across many thresholds and all the data sets used. On the average, SMOTE does not provide a superior predictive performance than the Base-

line on all the metrics. Below we further provide conclusive evidence on the statistical significance of the above results on the AUPRC.

5.2 Statistical Hypothesis Testing

Table 9 shows the results of the Friedman test applied on AUPRC to verify the statistical significance of WCGAN-GP compared to SMOTE. There is enough evidence at 5% significance level to reject the null hypothesis on 3 of the data sets, except German credit scoring and Glass identification, suggesting that over-sampling methods are not performing similarly and are different.

Data set	P-value	Significance
Credit Card Fraud	2.9560e-23	Yes
Pima Indians Diabetes	0.188386	No
German Credit Scoring	1.0683e-11	Yes
Glass Identification	0.465622	No
Breast Cancer Wisconsin	4.0085e-12	Yes

Table 9: Results for Friedman's test

Since the null hypothesis was rejected for 3 of the data sets, a post-hoc test was applied to further determine pairwise comparisons using the Nemenyi test where WCGAN-GP is the control method.

Test	Credit Card Fraud	German Credit	Breast Cancer
WCGAN-GP vs. SMOTE	0.001000	0.003000	0.001000

Table 10: Results for the post-hoc test

The above results confirm the significant superiority of WCGAN-GP over SMOTE as all the p-values are less than 0.05 for the 3 data sets where Friedman's test suggested a difference. These results confirm the findings shown in figure 13 and table 8 where the average performance seen on both the AUC and AUPRC was lower for SMOTE compared to WCGAN-GP. In general, WCGAN-GP provides statistically significant better performance on 3 of the 5 data sets.

5.3 Discussion

Overall, SMOTE improves the AUC/AUPRC when applied on the imbalanced data set but significantly penalises Precision, leading to a lower AUPRC on 2 of the data sets used. SMOTE samples synthetic points along line segments joining minority instances using the Euclidean distance. This approach may end up using majority instances and thus creating noisy examples and over-lapping cases [73, 148]. SMOTE is not based on the true distribution of the minority class data [38]. The poor performance of SMOTE (especially on Precision on the Credit card fraud data set) may be attributed to these effects. Overall, SMOTE alters the data distribution as was observed by the significant compromise on Precision and generally lower F1-Score, AUPRC and AUC values.

Other SMOTE variants such as density-based approaches are meant to improve the above SMOTE weaknesses [62, 199]. However, they make strict assumptions about the structure and distribution of the minority class data. SMOTE was the quickest to over-sample. WCGAN-GP requires a significant pre-training of both the critic and the generator.

GANs are well-known for their training and computing powers [35, 120]. Thus they have expensive run-times. However, current GANs such as WGAN and WGAN-GP remedy this impact with stable training. The quality of generated samples may be worth it compared to the training times. In this study, the GANs reached stable training even for small samples such as credit card fraud cases. This means that GANs may still be used even for smaller data sets with enough training capacity.

Using WCGAN-GP to over-sample minority cases provided the best performance on the AUPRC and on 3 of the data used on AUC. GANs do not make explicit assumptions about the probability distribution of the minority class data. This idea has been used to create new samples for images, music, arts and videos [63, 184, 195]. There is a significant potential to create new samples using GANs and augment limited actuarial data sets. Recent work on this [45, 53] report GAN superior performances over SMOTE and other variants. While GANs are notoriously difficult to train and optimise, in this study, using a simple architecture provided stable significant results after 5000 epochs.

Given the current surge in interest for GANs, optimising and training GANs is becoming straightforward as there are many implementations in Keras [55], Pytorch and Tensorflow [178]. Thus running times for GANs might not necessarily be an issue, enabling GANs to provide a superior over-sampling approach to supplement imbalanced data sets.

Given the superiority of GANs over other generative models and their wide applications, the scope for actuarial use is extensive. The most obvious use is data augmentation and boosting predictive models as demonstrated in the experiments conducted. Other applications include anomaly detection, discriminative modelling, semi-supervised learning, domain adaptation, attention prediction, data manipulation, missing data imputation, time series generation and privacy preservation. There are many available GANs for use in each of the above domains. The most interesting use case is semi-supervised learning as it can do both data augmentation and also acting as a classifier. Research for GANs grows each year and actuaries may need to add GANs to their toolkit as this will significantly improve their models and aid on decision-making.

6 Conclusion and Future Research

This section concludes this work and provides scope for future research.

6.1 Conclusions

Gaining an advantage in competitive markets through offerings of suitable tailored products on customers relies on building and maintaining adequate predictive models. To build these models, a substantial amount of data and a sizeable number of records is desirable. However, when expanding to new or unexplored markets, that level of information is rarely always available. As a result, actuarial firms have to buy data from a local provider, through purchasing reinsurance from a re-insurer, through limited unsuitable industry and public research or rely from extrapolations from other better known markets. In this work, we show how an implicit model using GANs can alleviate this problem through the generation of adequate quality data even from very limited small samples, from difficult domains or without alignment.

This example is a classic data augmentation application of GANs where we showed their superiority of SMOTE and improving the original results. SMOTE improved the classification performance. However, SMOTE is not based on the true underlying minority class distribution. SMOTE density estimation approaches remedy this issue, however, they are also not based on the true data distribution as they make

strong data assumptions.

Using WCGAN-GP, it is possible to create synthetic cases implicitly and this turned out to offer a significantly better improvement over SMOTE. This work comprehensively reviews GAN theory and applications in a number of domains, with possible adoption for actuarial use. GANs are highly successful for data augmentation, discriminative modelling, semi-supervised learning, privacy preservation, domain adaptation, attention prediction, anomaly detection, imputation and time series generation. We believe that these applications have scope for actuarial science and actuaries can add them to their toolkit to add predictive models.

6.2 Limitations

This work focused on using example continuous data. Other data sets have mixed data types such as count, ordinal and categorical features. This work also considered binary cases whereas other data sets may have multiple classes. The analysis can be repeated for other actuarial data sets such as mortality, claims frequency and amounts etc. We repeated training and testing of each over-sampling method 30 times to minimise stochastic effects - this sample size can be increased for more robustness. Alternatively a bootstrapping approach can be applied to better understand the distributional attributes of the model errors.

We were also limited to provide adequate practical examples for each GAN application domain. Our future work includes comparing current traditional actuarial approaches such as stochastic simulations and pricing models versus each GAN approach in each domain, extensively.

Because GANs have become so popular, their limitations have been improved tremendously. However, there are still open challenges for GANs. GANs rely on the generated examples being completely differentiable with respect to the generative parameters. As a result, GANs cannot product discrete data directly, such as one-hot word. Thus, solving this problem would unlock the significant application of GANs in NLP, albeit there have been some sequence success on text generations [156, 157].

Although GANs are useful for sample generation, there is not a useful metric that can be used to measure the uncertainty of the well-trained generator. This is another interesting future direction. Another key challenge is the evaluation of GANs after training even though there are measures to compute the quality of results generated.

There are many GAN variants and we point the reader to comprehensive papers by Creswell et al. [35], Gui et al. [70], Hitawala [81], Hong et al. [85] and Wang et al. [188]. Goodfellow [67] provides a thorough theoretical GAN formulation, including key issues and some new directions.

6.3 Future Research

Below are possible future research to improve this work:

- Consideration on other data sets to apply the same techniques, especially complex data sets that include small disjuncts, over-lapping, mixed data types and multiple classes.
- Alternative consideration for other ML algorithms such as ANN, Support Vector Machines [25], Random Forests [21] and Gradient Boosting Machines [20, 27]. Such a comprehensive study would show which ML technique is best and for which data set and domain.
- Empirical comparison of these results with other tabular data sets where GAN was applied.

- New Adam variants were recently proposed called Rectified Adam (RAdam) [116], AMSGrad [155] and LookAhead or Ranger [203], which seem to show better results. However, if remains to be seen whether these optimisers will consistently surpass Adam and others. These optimisers could further improve GAN training and stability.

In our opinion, the future of GANs will be characterised by open acceptance of GANs and their applications by the research community and being used in commercial applications. Given their impressive results and advancement in deep learning techniques, we expect a wider extensive use of GANs. The training instability of GANs will soon be done without any problems as the maturity of the training improves with new techniques being invented at a rapid speed. There are many potential future applications of GANs, with significant potential for actuarial use and improving existing models and being applied in other areas by actuaries. Given the surge in marketing and social promotions, info-graphics are the main ingredient of social media marketing. Artificial intelligence and GANs can help marketers and designers in the creative process.

"...GANs and their variants..the most interesting idea in the last 10 years in machine learning..."

Facebook's AI research director, Yann LeCun

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work originally emanates from a Masters in Data Science degree completed at the University of the Witwatersrand between 2018 and 2019, funded by the Department of Science and Technology National Research Fund, through the National e-Science Postgraduate teaching and training platform. The first author is grateful to the co-author Mr. Rendani Mbuvha for the guidance, providing inspiring insights and supervision throughout.

References

[1] D.H. Ackley, G.E. Hinton, and T.J. Sejnowski. "A learning algorithm for Boltzmann machines". In: *Cognitive Science* 9.1 (1985), pp. 147–169.

[2] H. Ajakan et al. "Domain-adversarial neural networks". In: *arXiv preprint arXiv:1412.4446* (2014).

[3] S. Akcay, A. Atapour-Abarghouei, and T.P. Breckon. "Ganomaly: Semi-supervised anomaly detection via adversarial training". In: *Asian Conference on Computer Vision*. Springer. 2018, pp. 622–637.

[4] G. Antipov, M. Baccouche, and J. Dugelay. "Face aging with conditional generative adversarial networks". In: *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2017, pp. 2089–2093.

[5] A. Antoniou, A. Storkey, and H. Edwards. "Data augmentation generative adversarial networks". In: *arXiv preprint arXiv:1711.04340* (2017).

[6] M. Arjovsky, S. Chintala, and L. Bottou. "Wasserstein GAN". In: *arXiv preprint arXiv:1701.07875* (2017).

[7] K. Armanious et al. "MedGAN: Medical image translation using GANs". In: *arXiv preprint arXiv:1806.06397* (2018).

[8] L. Aviñó, M. Ruffini, and R. Gavaldà. "Generating Synthetic but Plausible Healthcare Record Datasets". In: *arXiv preprint arXiv:1807.01514* (2018).

[9] Y. Bai et al. "Sod-mtgan: Small object detection via multi-task generative adversarial network". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 206–221.

[10] B.K. Beaulieu-Jones et al. "Privacy-preserving generative deep neural networks support clinical data sharing". In: *Circulation: Cardiovascular Quality and Outcomes* 12.7 (2019), e005122.

[11] M. Bekkar, H. Kheliouane Djemaa, and Taklit A. A. "Evaluation measures for models assessment over imbalanced data sets". In: *Journal of Information Engineering and Applications* 3.10 (2013).

[12] C. Bellinger, C. Drummond, and N. Japkowicz. "Beyond the Boundaries of SMOTE". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2016, pp. 248–263.

[13] C. Bellinger, N. Japkowicz, and C. Drummond. "Synthetic oversampling for advanced radioactive threat detection". In: *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE. 2015, pp. 948–953.

[14] Y. Bengio et al. "Deep generative stochastic networks trainable by backprop". In: *International Conference on Machine Learning*. 2014, pp. 226–234.

[15] M. Benhenda. "ChemGAN challenge for drug discovery: can AI reproduce natural chemical diversity?" In: *arXiv preprint arXiv:1708.08227* (2017).

[16] U. Bergmann, N. Jetchev, and R. Vollgraf. "Learning texture manifolds with the periodic spatial GAN". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 469–477.

[17] D. Berthelot, T. Schumm, and L. Metz. "Began: Boundary equilibrium generative adversarial networks". In: *arXiv preprint arXiv:1703.10717* (2017).

[18] G.E.P. Box et al. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[19] A.P. Bradley. "The use of the area under the ROC curve in the evaluation of machine learning algorithms". In: *Pattern Recognition* 30.7 (1997), pp. 1145–1159.

[20] L. Breiman. "Bagging predictors". In: *Journal of Machine Learning* 24.2 (1996), pp. 123–140.

[21] L. Breiman. "Random forests". In: *Journal of Machine Learning* 45.1 (2001), pp. 5–32.

[22] J. Briot, G. Hadjeres, and journal=arXiv preprint arXiv:1709.01620 year=2017 Pachet F. "Deep learning techniques for music generation–a survey". In: ().

[23] A. Brock, J. Donahue, and K. Simonyan. "Large scale gan training for high fidelity natural image synthesis". In: *arXiv preprint arXiv:1809.11096* (2018).

[24] S. Buuren and K. Groothuis-Oudshoorn. "mice: Multivariate imputation by chained equations in R". In: *Journal of statistical software* (2010), pp. 1–68.

[25] C. Chang and C. Lin. "LIBSVM: a library for support vector machines". In: *ACM transactions on intelligent systems and technology (TIST)* 2.3 (2011), p. 27.

[26] N.V. Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of Artificial Intelligence Research* 16 (2002), pp. 321–357.

[27] T. Chen and C. Guestrin. “Xgboost: A scalable tree boosting system”. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM. 2016, pp. 785–794.

[28] T. Chen et al. “Show, adapt and tell: Adversarial training of cross-domain image captioner”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 521–530.

[29] X. Chen et al. “Infogan: Interpretable representation learning by information maximizing generative adversarial nets”. In: *Advances in Neural Information Processing systems*. 2016, pp. 2172–2180.

[30] Y. Choi et al. “Stargan: Unified generative adversarial networks for multi-domain image-to-image translation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8789–8797.

[31] L. Chongxuan et al. “Triple generative adversarial nets”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 4088–4098.

[32] C. Chu, A. Zhmoginov, and M. Sandler. “Cyclegan, a master of steganography”. In: *arXiv preprint arXiv:1712.02950* (2017).

[33] D. Clevert, T. Unterthiner, and S. Hochreiter. “Fast and accurate deep network learning by exponential linear units (elus)”. In: *arXiv preprint arXiv:1511.07289* (2015).

[34] I. Cordón et al. “Imbalance: oversampling algorithms for imbalanced classification in R”. In: *Knowledge-Based Systems* 161 (2018), pp. 329–341.

[35] A. Creswell et al. “Generative adversarial networks: An overview”. In: *IEEE Signal Processing Magazine* 35.1 (2018), pp. 53–65.

[36] W. Dai et al. “SCAN: Structure correcting adversarial network for organ segmentation in chest X-rays”. In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, 2018, pp. 263–273.

[37] A. Dal Pozzolo. “Adaptive machine learning for credit card fraud detection”. In: (2015).

[38] B. Das, N.C. Krishnan, and D.J. Cook. “RACOG and wRACOG: Two probabilistic oversampling techniques”. In: *IEEE transactions on knowledge and data engineering* 27.1 (2015), pp. 222–234.

[39] A. Dash et al. “Tac-gan-text conditioned auxiliary classifier generative adversarial network”. In: *arXiv preprint arXiv:1703.06412* (2017).

[40] E.L. Denton et al. “Unsupervised learning of disentangled representations from video”. In: *Advances in neural information processing systems*. 2017, pp. 4414–4423.

[41] X. Di, V.A. Sindagi, and V.M. Patel. “Gp-gan: Gender preserving gan for synthesizing faces from landmarks”. In: *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE. 2018, pp. 1079–1084.

[42] C. Donahue, B. Li, and R. Prabhavalkar. “Exploring speech enhancement with generative adversarial networks for robust speech recognition”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, pp. 5024–5028.

[43] C. Donahue, J. McAuley, and M. Puckette. “Synthesizing audio with generative adversarial networks”. In: *arXiv preprint arXiv:1802.04208* (2018).

[44] H. Dong et al. "Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[45] G. Douzas and F. Bacao. "Effective data generation for imbalanced learning using conditional generative adversarial networks". In: *Expert Systems with applications* 91 (2018), pp. 464–471.

[46] T. Dozat. "Incorporating nesterov momentum into adam". In: *ICLR Workshop*. Vol. 1. 2013. 2016, p. 2016.

[47] J. Duchi, E. Hazan, and Y. Singer. "Adaptive subgradient methods for online learning and stochastic optimization". In: *Journal of Machine Learning Research* 12.Jul (2011), pp. 2121–2159.

[48] J. Engel et al. "Gansynth: Adversarial neural audio synthesis". In: *arXiv preprint arXiv:1902.08710* (2019).

[49] C. Esteban, S.L. Hyland, and G. Rätsch. "Real-valued (medical) time series generation with recurrent conditional gans". In: *arXiv preprint arXiv:1706.02633* (2017).

[50] M. Ester et al. "A density-based algorithm for discovering clusters in large spatial databases with noise". In: *Kdd*. Vol. 96. 34. 1996, pp. 226–231.

[51] I.W. Evett and E.J. Spiehler. "Rule induction in forensic science". In: *KBS in Goverment* (1987), pp. 107–118.

[52] A. Fernández et al. "Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary". In: *Journal of Artificial Intelligence Research* 61 (2018), pp. 863–905.

[53] U. Fiore et al. "Using Generative Adversarial Networks for improving classification effectiveness in Credit card fraud detection". In: *Information Sciences* (2017).

[54] Python Software Foundation. "Python Language Reference (Version 3.6. 3.)" In: (2017).

[55] C. François. *keras*. https://github.com/fchollet/keras. 2015.

[56] B.J. Frey, G.E. Hinton, and P. Dayan. "Does the wake-sleep algorithm produce good density estimators?" In: *Advances in Neural Information Processing Systems*. 1996, pp. 661–667.

[57] J. Friedman, T. Hastie, and R. Tibshirani. *The Elements Of Statistical Learning*. Vol. 1. Springer series in statistics New York, 2001.

[58] M. Friedman. "The use of ranks to avoid the assumption of normality implicit in the analysis of variance". In: *Journal of the American Statistical Association* 32.200 (1937), pp. 675–701.

[59] R. Fu et al. "Time Series Simulation by Conditional Generative Adversarial Net". In: *arXiv preprint arXiv:1904.11419* (2019).

[60] Z. Gan et al. "Triangle generative adversarial networks". In: *Advances in Neural Information Processing Systems*. 2017, pp. 5247–5256.

[61] V. Ganganwar. "An overview of classification algorithms for imbalanced datasets". In: *International Journal of Emerging Technology and Advanced Engineering* 2.4 (2012), pp. 42–47.

[62] M. Gao et al. "PDFOS: PDF estimation based over-sampling for imbalanced two-class problems". In: *Neurocomputing* 138 (2014), pp. 248–259.

[63] J. Gauthier. "Conditional generative adversarial nets for convolutional face generation". In: *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester* 2014.5 (2014), p. 2.

[64] C.J. Geyer. "Practical markov chain monte carlo". In: *Statistical science* (1992), pp. 473–483.

[65] X. Glorot and Y. Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, pp. 249–256.

[66] X. Glorot, A. Bordes, and Y. Bengio. "Deep sparse rectifier neural networks". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 2011, pp. 315–323.

[67] I. Goodfellow. "NIPS 2016 tutorial: Generative adversarial networks". In: *arXiv preprint arXiv:1701.00160* (2016).

[68] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. Vol. 1. MIT press Cambridge, 2016.

[69] I. Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. 2014, pp. 2672–2680.

[70] J. Gui et al. "A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications". In: *arXiv preprint arXiv:2001.06937* (2020).

[71] G.L. Guimaraes et al. "Objective-reinforced generative adversarial networks (organ) for sequence generation models". In: ().

[72] I. Gulrajani et al. "Improved training of Wasserstein gans". In: *Advances in Neural Information Processing Systems*. 2017, pp. 5767–5777.

[73] H. Han, W. Wang, and B. Mao. "Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning". In: *International Conference on Intelligent Computing*. Springer. 2005, pp. 878–887.

[74] J.A. Hanley and B.J. McNeil. "The meaning and use of the area under a receiver operating characteristic (ROC) curve". In: *Radiology* 143.1 (1982), pp. 29–36.

[75] H. He and E.A. Garcia. "Learning from imbalanced data". In: *IEEE Transactions on Knowledge & Data Engineering* 9 (2008), pp. 1263–1284.

[76] K. He et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[77] Z. He et al. "Attgan: Facial attribute editing by only changing what you want". In: *IEEE Transactions on Image Processing* 28.11 (2019), pp. 5464–5478.

[78] G.E. Hinton. "Training products of experts by minimizing contrastive divergence". In: *Neural computation* 14.8 (2002), pp. 1771–1800.

[79] G.E Hinton, S. Osindero, and Y. Teh. "A fast learning algorithm for deep belief nets". In: *Neural Computation* 18.7 (2006), pp. 1527–1554.

[80] G.E. Hinton and R.R. Salakhutdinov. "Reducing the dimensionality of data with neural networks". In: *science* 313.5786 (2006), pp. 504–507.

[81] S. Hitawala. "Comparative study on generative adversarial networks". In: *arXiv preprint arXiv:1801.04271* (2018).

[82] S. Hochreiter and journal=Neural computation volume=9 number=8 pages=1735–1780 year=1997 publisher=MIT Press Schmidhuber J. "Long short-term memory". In: ().

[83] J. Hoffman et al. “Cycada: Cycle-consistent adversarial domain adaptation”. In: *arXiv preprint arXiv:1711.03213* (2017).

[84] W. Hong et al. “Conditional generative adversarial network for structured domain adaptation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1335–1344.

[85] Y. Hong et al. “How Generative Adversarial Networks and Their Variants Work: An Overview”. In: *ACM Computing Surveys (CSUR)* 52.1 (2019), p. 10.

[86] C. Hsu et al. “Voice conversion from unaligned corpora using variational autoencoding wasserstein generative adversarial networks”. In: *arXiv preprint arXiv:1704.00849* (2017).

[87] U. Hwang, D. Jung, and S. Yoon. “HexaGAN: Generative adversarial nets for real world classification”. In: *arXiv preprint arXiv:1902.09913* (2019).

[88] S. Ioffe and C. Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015).

[89] N. Japkowicz and S. Stephen. “The class imbalance problem: A systematic study”. In: *Intelligent Data Analysis* 6.5 (2002), pp. 429–449.

[90] N. Jetchev, U. Bergmann, and R. Vollgraf. “Texture synthesis with spatial generative adversarial networks”. In: *arXiv preprint arXiv:1611.08207* (2016).

[91] Y. Jin et al. “Towards the automatic anime characters creation with generative adversarial networks”. In: *arXiv preprint arXiv:1708.05509* (2017).

[92] A. Jolicoeur-Martineau. “The relativistic discriminator: a key element missing from standard GAN”. In: *arXiv preprint arXiv:1807.00734* (2018).

[93] T. Karras, S. Laine, and T. Aila. “A style-based generator architecture for generative adversarial networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4401–4410.

[94] T. Karras et al. “Progressive growing of gans for improved quality, stability, and variation”. In: *arXiv preprint arXiv:1710.10196* (2017).

[95] S. Kazeminia et al. “GANs for medical image analysis”. In: *arXiv preprint arXiv:1809.06222* (2018).

[96] N. Killoran et al. “Generating and designing DNA with deep generative models”. In: *arXiv preprint arXiv:1712.06148* (2017).

[97] T. Kim et al. “Learning to discover cross-domain relations with generative adversarial networks”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1857–1865.

[98] D.P. Kingma and J. Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[99] D.P. Kingma and M. Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[100] D. Korkinof et al. “High-resolution mammogram synthesis using progressive generative adversarial networks”. In: *arXiv preprint arXiv:1807.03401* (2018).

[101] D. Korkinof et al. “MammoGAN: High-Resolution Synthesis of Realistic Mammograms”. In: (2019).

[102] S. Kullback. *Information theory and statistics*. Courier Corporation, 1997.

[103] S. Kullback and R.A. Leibler. “On information and sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86.

[104] A. Kumar, A. Biswas, and S. Sanyal. "ecommercegan: A generative adversarial network for e-commerce". In: *arXiv preprint arXiv:1801.03244* (2018).

[105] A.B.L. Larsen et al. "Autoencoding beyond pixels using a learned similarity metric". In: *arXiv preprint arXiv:1512.09300* (2015).

[106] C. Ledig et al. "Photo-realistic single image super-resolution using a generative adversarial network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4681–4690.

[107] K.J. Lee and J.B. Carlin. "Multiple imputation for missing data: fully conditional specification versus multivariate normal imputation". In: *American Journal of Epidemiology* 171.5 (2010), pp. 624–632.

[108] S. Lee et al. "A seqgan for polyphonic music generation". In: *arXiv preprint arXiv:1710.11418* (2017).

[109] J. Li et al. "Perceptual generative adversarial networks for small object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1222–1230.

[110] S.C. Li, B. Jiang, and B. Marlin. "Misgan: Learning from incomplete data with generative adversarial networks". In: *arXiv preprint arXiv:1902.09599* (2019).

[111] Y. Li, K. Swersky, and R. Zemel. "Generative moment matching networks". In: *International Conference on Machine Learning*. 2015, pp. 1718–1727.

[112] J.H. Lim and J.C. Ye. "Geometric gan". In: *arXiv preprint arXiv:1705.02894* (2017).

[113] S.K. Lim et al. "DOPING: Generative data augmentation for unsupervised anomaly detection with GAN". In: *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2018, pp. 1122–1127.

[114] J. Lin. "Divergence measures based on the Shannon entropy". In: *IEEE Transactions on Information theory* 37.1 (1991), pp. 145–151.

[115] K. Lin et al. "Adversarial ranking for language generation". In: *Advances in Neural Information Processing Systems*. 2017, pp. 3155–3165.

[116] L. Liu et al. "On the variance of the adaptive learning rate and beyond". In: *arXiv preprint arXiv:1908.03265* (2019).

[117] M. Liu and O. Tuzel. "Coupled generative adversarial networks". In: *Advances in Neural Information Processing Systems*. 2016, pp. 469–477.

[118] Z. Liu, J. Wang, and Z. Liang. "CatGAN: Category-aware Generative Adversarial Networks with Hierarchical Evolutionary Learning for Category Text Generation". In: *arXiv preprint arXiv:1911.06641* (2019).

[119] V. López et al. "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics". In: *Information sciences* 250 (2013), pp. 113–141.

[120] M. Lucic et al. "Are gans created equal? a large-scale study". In: *Advances in Neural Information Processing systems*. 2018, pp. 700–709.

[121] A.L. Maas, A.Y. Hannun, and A.Y. Ng. "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml*. Vol. 30. 1. 2013, p. 3.

[122] L. Maaten and G. Hinton. "Visualizing data using t-SNE". In: *Journal of Machine Learning Research* 9.Nov (2008), pp. 2579–2605.

[123] A. Makhzani et al. "Adversarial autoencoders". In: *arXiv preprint arXiv:1511.05644* (2015).

[124] P. Manisha and S. Gujar. "Generative Adversarial Networks (GANs): What it can generate and What it cannot?" In: *arXiv preprint arXiv:1804.00140* (2018).

[125] X. Mao et al. "Least Squares Generative Adversarial Networks". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2794–2802.

[126] G. Mariani et al. "Bagan: Data Augmentation with Balancing GAN". In: *arXiv preprint arXiv:1803.09655* (2018).

[127] J. Mathew et al. "Kernel-based SMOTE for SVM classification of imbalanced datasets". In: *IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society*. IEEE. 2015, pp. 1127–1132.

[128] P. McCullagh. "Generalized linear models". In: *European Journal of Operational Research* 16.3 (1984), pp. 285–292.

[129] M. Mirza and S. Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[130] D. Misra. "Mish: A Self Regularized Non-Monotonic Neural Activation Function". In: *arXiv preprint arXiv:1908.08681* (2019).

[131] T.M. Mitchell. *The Discipline of Machine Learning*. Vol. 9. Carnegie Mellon University, School of Computer Science, Machine Learning Department, 2006.

[132] T. Miyato et al. "Virtual adversarial training: a regularization method for supervised and semi-supervised learning". In: *IEEE transactions on pattern analysis and machine intelligence* 41.8 (2018), pp. 1979–1993.

[133] O. Mogren. "C-RNN-GAN: Continuous recurrent neural networks with adversarial training". In: *arXiv preprint arXiv:1611.09904* (2016).

[134] A. Mottini, A. Lheritier, and R. Acuna-Agost. "Airline passenger name record generation using generative adversarial networks". In: *arXiv preprint arXiv:1807.06657* (2018).

[135] A. Müller. "Integral probability metrics and their generating classes of functions". In: *Advances in Applied Probability* 29.2 (1997), pp. 429–443.

[136] S.S. Mullick, S. Datta, and S. Das. "Generative Adversarial Minority Oversampling". In: *arXiv preprint arXiv:1903.09730* (2019).

[137] V. Nair and G.E. Hinton. "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 807–814.

[138] P. Nemenyi. "Distribution-free multiple comparisons". In: *Biometrics*. Vol. 18. 2. Princeton University. 1962, p. 263.

[139] Y. Nesterov. "A method for unconstrained convex minimization problem with the rate of convergence O (1/kˆ 2)". In: *Doklady AN USSR*. Vol. 269. 1983, pp. 543–547.

[140] W. Nie, N. Narodytska, and A. Patel. "Relgan: Relational generative adversarial networks for text generation". In: *International Conference on Learning Representations* (2019).

[141] S. Nowozin, B. Cseke, and R. Tomioka. "f-gan: Training generative neural samplers using variational divergence minimization". In: *Advances in Neural Information Processing systems*. 2016, pp. 271–279.

[142] A. Odena. "Semi-supervised learning with generative adversarial networks". In: *arXiv preprint arXiv:1606.01583* (2016).

[143] A. Odena, C. Olah, and J. Shlens. "Conditional image synthesis with auxiliary classifier gans". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 2642–2651.

[144] A. Van den Oord et al. "Conditional image generation with pixelcnn decoders". In: *Advances in neural information processing systems*. 2016, pp. 4790–4798.

[145] N. Park et al. "Data synthesis based on generative adversarial networks". In: *Proceedings of the Very Large Data Bases Endowment* 11.10 (2018), pp. 1071–1083.

[146] T. Park et al. "Semantic image synthesis with spatially-adaptive normalization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2337–2346.

[147] S. Pascual, A. Bonafonte, and J. Serra. "SEGAN: Speech enhancement generative adversarial network". In: *arXiv preprint arXiv:1703.09452* (2017).

[148] M. Pérez-Ortiz, P.A. Gutiérrez, and C. Hervás-Martínez. "Borderline kernel based over-sampling". In: *International Conference on Hybrid Artificial Intelligence Systems*. Springer. 2013, pp. 472–481.

[149] T. Pohlert. "The pairwise multiple comparison of mean ranks package (PMCMR)". In: *R package* 27 (2014).

[150] G. Qi. "Loss-sensitive generative adversarial networks on lipschitz densities". In: *arXiv preprint arXiv:1701.06264* (2017).

[151] N. Qian. "On the momentum term in gradient descent learning algorithms". In: *Neural networks* 12.1 (1999), pp. 145–151.

[152] T. Qiao et al. "Mirrorgan: Learning text-to-image generation by redescription". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1505–1514.

[153] A. Radford, L. Metz, and S. Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[154] P. Ramachandran, B. Zoph, and Q.V. Le. "Searching for activation functions". In: *arXiv preprint arXiv:1710.05941* (2017).

[155] S.J. Reddi, S. Kale, and S. Kumar. "On the convergence of adam and beyond". In: *arXiv preprint arXiv:1904.09237* (2019).

[156] S. Reed et al. "Generative adversarial text to image synthesis". In: *arXiv preprint arXiv:1605.05396* (2016).

[157] S. Reed et al. "Generative adversarial text to image synthesis". In: *arXiv preprint arXiv:1605.05396* (2016).

[158] D.J. Rezende, S. Mohamed, and D. Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *arXiv preprint arXiv:1401.4082* (2014).

[159] D.B. Rubin. *Multiple imputation for nonresponse in surveys*. Vol. 81. John Wiley & Sons, 2004.

[160] Y. Rubner, C. Tomasi, and L.J. Guibas. "The earth mover's distance as a metric for image retrieval". In: *International Journal of Computer Vision* 40.2 (2000), pp. 99–121.

[161] S. Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv preprint arXiv:1609.04747* (2016).

[162] D.E Rumelhart, G.E. Hinton, and R.J. Williams. "Learning Representations by Back-propagating Errors". In: *Nature* 323.6088 (1986), p. 533.

[163] T. Saito and M. Rehmsmeier. "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets". In: *PloS one* 10.3 (2015), e0118432.

[164] Y. Saito, S. Takamichi, and H. Saruwatari. "Statistical parametric speech synthesis incorporating generative adversarial networks". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.1 (2017), pp. 84–96.

[165] R. Salakhutdinov and G. Hinton. "Deep boltzmann machines". In: *Artificial Intelligence and Statistics*. 2009, pp. 448–455.

[166] T. Salimans et al. "Improved techniques for training GANs". In: *Advances in Neural Information Processing Systems*. 2016, pp. 2234–2242.

[167] J.L. Schafer and M.K. Olsen. "Multiple imputation for multivariate missing-data problems: A data analyst's perspective". In: *Multivariate behavioral research* 33.4 (1998), pp. 545–571.

[168] T. Schlegl et al. "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery". In: *International Conference on Information Processing in Medical Imaging*. Springer. 2017, pp. 146–157.

[169] C. Shang et al. "VIGAN: Missing view imputation with generative adversarial networks". In: *2017 IEEE International Conference on Big Data (Big Data)*. IEEE. 2017, pp. 766–775.

[170] H. Shin et al. "Continual learning with deep generative replay". In: *Advances in Neural Information Processing Systems*. 2017, pp. 2990–2999.

[171] J.W. Smith et al. "Using the ADAP learning algorithm to forecast the onset of diabetes mellitus". In: *Proceedings of the Annual Symposium on Computer Application in Medical Care*. American Medical Informatics Association. 1988, pp. 261–265. URL: \url{https:www.kaggle.com/uciml/pima-indians-diabetes-database}.

[172] J. Song et al. "Multi-agent generative adversarial imitation learning". In: *Advances in Neural Information Processing Systems*. 2018, pp. 7461–7472.

[173] K. Sricharan et al. "Semi-supervised conditional gans". In: *arXiv preprint arXiv:1708.05789* (2017).

[174] N. Srivastava et al. "Dropout: a simple way to prevent neural networks from overfitting". In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[175] D.J. Stekhoven and P. Bühlmann. "MissForest—non-parametric missing value imputation for mixed-type data". In: *Bioinformatics* 28.1 (2012), pp. 112–118.

[176] W.N. Street, W.H. Wolberg, and O.L. Mangasarian. "Nuclear feature extraction for breast tumor diagnosis". In: *Biomedical image processing and biomedical visualization*. Vol. 1905. International Society for Optics and Photonics. 1993, pp. 861–870.

[177] H. Tang et al. "Cycle in cycle generative adversarial networks for keypoint-guided image generation". In: *Proceedings of the 27th ACM International Conference on Multimedia*. 2019, pp. 2052–2060.

[178] TensorFlow Team. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from http://www.tensorflow.org. 2015. URL: http://www.tensorflow.org.

[179] R. Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), pp. 267–288.

[180] L. Tran, X. Yin, and X. Liu. "Representation learning by rotating your faces". In: *IEEE transactions on pattern analysis and machine intelligence* 41.12 (2018), pp. 3007–3021.

[181] S. Tulyakov et al. "Mocogan: Decomposing motion and content for video generation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1526–1535.

[182] E. Tzeng et al. "Adversarial discriminative domain adaptation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 7167–7176.

[183] D. Volkhonskiy, I. Nazarov, and E. Burnaev. "Steganographic generative adversarial networks". In: *Twelfth International Conference on Machine Vision (ICMV 2019)*. Vol. 11433. International Society for Optics and Photonics. 2020, p. 114333M.

[184] C. Vondrick, H. Pirsiavash, and A. Torralba. "Generating videos with scene dynamics". In: *Advances in Neural Information Processing Systems*. 2016, pp. 613–621.

[185] E.M. Voorhees. "Implementing agglomerative hierarchic clustering algorithms for use in document retrieval". In: *Information Processing & Management* 22.6 (1986), pp. 465–476.

[186] J. Walker et al. "The pose knows: Video forecasting by generating pose futures". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 3332–3341.

[187] J. Wang et al. "Irgan: A minimax game for unifying generative and discriminative information retrieval models". In: *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 2017, pp. 515–524.

[188] K. Wang et al. "Generative adversarial networks: introduction and outlook". In: *IEEE/CAA Journal of Automatica Sinica* 4.4 (2017), pp. 588–598.

[189] T. Wang et al. "Video-to-video synthesis". In: *arXiv preprint arXiv:1808.06601* (2018).

[190] X. Wang et al. "Esrgan: Enhanced super-resolution generative adversarial networks". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 0–0.

[191] Y. Xue et al. "Segan: Adversarial network with multi-scale l 1 loss for medical image segmentation". In: *Neuroinformatics* 16.3-4 (2018), pp. 383–392.

[192] L. Yang, S. Chou, and Y. Yang. "MidiNet: A convolutional generative adversarial network for symbolic-domain music generation". In: *arXiv preprint arXiv:1703.10847* (2017).

[193] Z. Yi et al. "Dualgan: Unsupervised dual learning for image-to-image translation". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2849–2857.

[194] G. Yildirim et al. "Generating High-Resolution Fashion Model Images Wearing Custom Outfits". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2019, pp. 0–0.

[195] D. Yoo et al. "Pixel-level domain transfer". In: *European Conference on Computer Vision*. Springer. 2016, pp. 517–532.

[196] J. Yoon, J. Jordon, and M. Van Der Schaar. "Gain: Missing data imputation using generative adversarial nets". In: *arXiv preprint arXiv:1806.02920* (2018).

[197] L. Yu et al. "Seqgan: Sequence generative adversarial nets with policy gradient". In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.

[198] M.D. Zeiler. "ADADELTA: an adaptive learning rate method". In: *arXiv preprint arXiv:1212.5701* (2012).

[199] H. Zhang and M. Li. "RWO-Sampling: A Random Walk Over-Sampling Approach to Imbalanced Data Classification". In: 20 (Nov. 2014).

[200] H. Zhang et al. "Self-attention generative adversarial networks". In: *arXiv preprint arXiv:1805.08318* (2018).

[201] H. Zhang et al. "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 5907–5915.

[202] J. Zhang et al. "Privbayes: Private data release via bayesian networks". In: *ACM Transactions on Database Systems (TODS)* 42.4 (2017), p. 25.

[203] M.R. Zhang et al. "Lookahead Optimizer: k steps forward, 1 step back". In: *arXiv preprint arXiv:1907.08610* (2019).

[204] J. Zhao, M. Mathieu, and Y. LeCun. "Energy-based generative adversarial network". In: *arXiv preprint arXiv:1609.03126* (2016).

[205] S. Zhou et al. "Genegan: Learning object transfiguration and attribute subspace from unpaired data". In: *arXiv preprint arXiv:1705.04932* (2017).

[206] X. Zhou et al. "Stock market prediction on high-frequency data using generative adversarial nets". In: *Mathematical Problems in Engineering* 2018 (2018).

[207] J. Zhu et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232.