Generative versus Sampling-Based Approaches to Variability of Class Imbalance in Visual Anomaly Detection

by

Nasik Muhammad Nafi

B.Sc., Bangladesh University of Engineering and Technology, 2015

A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computer Science
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2019

Approved by:

Major Professor
Dr. William H. Hsu

Copyright

Abstract

Data sets for visual anomaly detection are often stratified such that every stratum or batch in the data set suffers from imbalance of different magnitude. A common approach to this detection task is to use supervised inductive learning from labeled or partially labeled image data to simultaneously solve the task of segmenting the anomaly and classifying it. Many representations and algorithms for these learning tasks exhibit some preference (inductive bias) towards balanced data from each class and thus perform better with balanced data sets than imbalanced. Such representations and algorithms are sensitive to not only the aggregate degree of class imbalance but its within-stratum variation. This includes learning representations such as deep learning for intermediate visual features.

Several oversampling-based techniques have been proposed to mitigate the skewness of the data. However, most of the synthetic oversampling techniques such as Synthetic Minority Over-sampling Technique (SMOTE) or Adaptive Synthetic Sampling (ADASYN) are suitable only for the low dimensional data which limits their application in visual anomaly detection. Recently, deep generative models such as Variational Autoencoders (VAE) or Generative Adversarial Networks (GAN) have been established as effective approaches to augment high-dimensional image data. However, the literature lacks a detailed study of the learning process in a data set augmented to cope with variable imbalance across strata.

We carried out an experiment to analyze the training phase and the final classifier performance when the more imbalanced batch is augmented using different approaches to achieve the same data ratio as the less imbalanced batch. We identified the classification on merged batches as baseline and compared the performance of the classifier on data sets augmented by simple oversampling, an adaptation of SMOTE, and a GAN-based generative model. Our results indicate that the GAN-based augmentation is capable of avoiding overfitting and leads to better performance.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First of all I would like to thank my supervisor, Dr. William H. Hsu, for his continuous supervision, guidance, and valuable advice throughout the entire research work. I thank him for his constant criticism, which has given me a positive outlook towards the various problems faced during my research and strengthened my determination and confidence to complete this thesis. I am especially grateful to him for allowing me to choose my course of actions with freedom, and for enlightening me during the moments of frustration.

I would like to express my gratitude to Dr. Mitchell Neilsen and Dr. Torben Amtoft for taking time to serve as my committee members. Opinions they provided meant a lot to me.

I would also like to thank my friends, labmates, and classmates for their support, encouragement, and critics. I specially thank Robert Stewart and Majed Alsadhan for letting me to use their machines for my simulations.

I take this opportunity to thank my wife, Nazmun Akter Pia, for being so lovable, supporting and kind. Also, I am grateful to my parents, elder sister, younger brother, in-laws and other family members who provided continuous support throughout my life.

Chapter 1

Introduction

1.1 Overview

In the real world, one is often faced with the problem of anomaly detection from image data. With advances in computational processing of high-dimensional data such as image, visual anomaly detection tasks become feasible from a computational perspective. However, due to the infrequent occurrence of anomalous events, data sets available for anomaly detection are inherently imbalanced. Sometimes these data come in batches from different sources. Due to variations in provenance, every batch may have a different distribution and commensurate degree of imbalance.

Nowadays, in visual anomaly detection, it is usual to artificially generate additional anomalous data to reduce the imbalance. The quality of the new data is dependent on these data synthesis techniques, which are the central topic of this thesis. Adding new data generated by different techniques have different effect on the classifier. Thus, data augmentation techniques, and specifically how learning of the classifier is influenced by the addition of new synthetic data, are active area of research.

1.2 Motivation

Anomaly detection is the task of identifying unusual items, events or observations which raise doubts by significantly varying from the majority of the data [55]. As these anomalous data points can be linked to some sort of problem or abnormal events such as electricity pilferage, fraudulent transactions, rare diseases, malfunctioning equipment, etc., identification of those events are of particular interest. Specifically, visual anomaly detection problems also arise in a broad range of real-life application fields where irregular events such as cracks in bridges or other constructions, product defects, human or plant disease, irregularities in galaxy images etc. need to be detected from images [6]. Due to the prevalence of imbalanced data in various visual anomaly detection, it is worthy to explore the effective imbalanced learning methods.

Our main motivation came from a project of the Kansas State University Laboratory for Knowledge Discovery in Databases (KSU KDD Lab) which is titled as "Detecting damaged insulator from electric tower images using deep learning" and sponsored by Black & Veatch. Inc [23]. In this project we were required to classify damaged and undamaged insulator from tower images. Data was collected from different lines by a KDD Lab affiliate at Kansas State University Polytechnic using remote-piloted unmanned aerial systems (UAS), or drones. Furthermore, due to the variability of weather and maintenance facilities the lines had different ratios of damaged and undamaged insulators. There was a scarcity of damaged insulator images in some lines which made the classification task more critical. Our ultimate target was to increase the classifier performance over all samples available. However, these raised the following four questions:

- How we can address the variability of both classes in different lines?
- Does this variation of imbalance across batches really matter?
- Is there any problem if we simply merge all the batches to get a single data set?
- If we want to balance every batches how could that be achieved? Which balance ratio we should try to achieve?

The simplest approach to identify irregularities in data is to flag the data points that deviate from common statistical properties of a distribution. Popular machine learning-based techniques for anomaly detections are decision trees, k-nearest neighbors (k-NN), k-means clustering, and support vector machine (SVM)-based clustering. However, in the presence of imbalance data these algorithms tend to treat minority samples as noise and hence produce a strong bias towards the majority class. Skewed distributions also lead to failure in learning the true distribution of the minority class owing to lack of enough representative.

Two types of approaches were proposed by the researchers to improve the classification performance of imbalanced data sets. One approach is to hit the problem from algorithmic perspective, and another is to look at the problem from data level. In the first approach, the classifier itself is altered at the algorithm level to bias towards the minority class, while keeping the original data unchanged. For example: cost-sensitive learning and recognition-based learning [31]. Cost sensitive learning puts emphasis to the cost of different kinds of misclassification. The aim of this type of learning is to limit the total cost at minimum level [38]. At data level, oversampling and under-sampling techniques are applied to create or delete samples to accomplish a balanced data distribution. We are particularly interested at the data level techniques.

1.3 Problem Statement

A very naive approach to achieve certain balance in a data set is to simply oversample the data point of minority class. However, this does not help much to learn the underrepresented features from minority class. Thus, how can we create new but different samples representing that minority class is one of the main questions to answer in case of addressing class imbalance. When dealing with imbalanced data, the classic synthetic oversampling techniques accomplished the state-of-the-art performance. However, these methods are solely designed for low dimensional feature space. Therefore, difficult to adapt with high-dimensional data samples, including images. Recent deep generative models such as VAE and GAN have already gained success in generating a variety of complex data, such as handwritten digits,

faces, road signs, bedroom scenes, and CIFAR images. However, there is a lack of literature comparing the common old oversampling-based techniques with most recent GAN-based techniques.

1.4 Objectives and Significance of the Thesis

1.4.1 Objectives

The complexity of the visual anomaly detection problems increases with the increase in variation of imbalance among batches and also with the increase in the skewness of data within batches. Data augmentation-based techniques are becoming more popular in solving visual anomaly detection problems. Thus, our thesis objective is to evaluate data augmentation techniques with respect to image quality, training time metrics, and test accuracy of the classifier trained on the augmented data set. Choosing appropriate data generation policy, balancing the data in imbalanced batches, and ensuring accurate prediction of anomalous class are the main focus. We also intend to study the behavior of different evaluation parameters like training accuracy, training loss, validation accuracy, validation loss etc. during the training process and their effect on final classification accuracy.

1.4.2 Significance

In this thesis, we compared data generation policy from different category and discussed in detail about how to merge batches in case of variation of class imbalance. We have done extensive experiments in accordance with our selected models and methods. In addition, we do not have any study which reveals the progression of loss function and accuracy of training and validation set during training in the above-mentioned scenario. However, this information can greatly help to understand the learning process particularly to validate underfitting and overfitting as well as to get an idea of bias-variance trade-off and generalization error. Therefore, we also analyzed the training time metrics.

Chapter 2

Background

2.1 Random Sampling

Among data enhancement based approaches, the very obvious choice to overcome class imbalance is to repeat same data multiple times. Studies have shown that an improved overall performance have been achieved with a balanced data compared with an imbalanced one for several base classifiers [10] [32]. To achieve a better balance in data two types of sampling can be done. One is to oversampling the instance of minority class and another is to under-sampling the majority class.

While oversampling methods adjust the data set by expanding the quantity of minority samples, undersampling methods try to remove some majority samples to keep balance [28]. Random oversampling increase the number of minority samples through arbitrarily replicating existing minority samples. Random oversampling is capable of improving the overall performance of learning process to some degree. However, it fails to provide any extra information to the training set. Moreover, random oversampling method has the risk of overfitting of machine learning models. In comparison to random oversampling, random undersampling methods may remove some important training information which may negatively affect the learning process [4]. Fig. 2.1 [51] and Fig. 2.2 [36] depicts the concept of random oversampling and undersampling respectively.

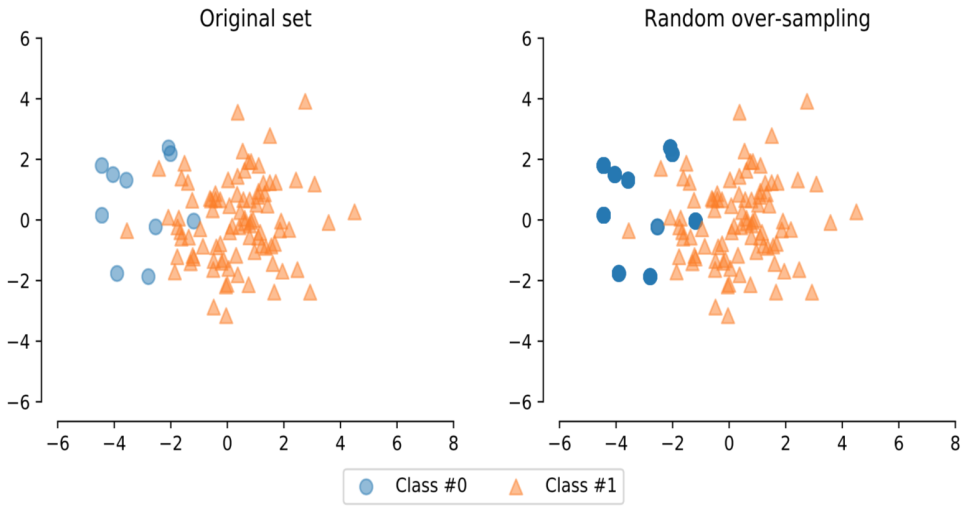


Figure 2.1: Random Oversampling (Zhang, 2018) [51]

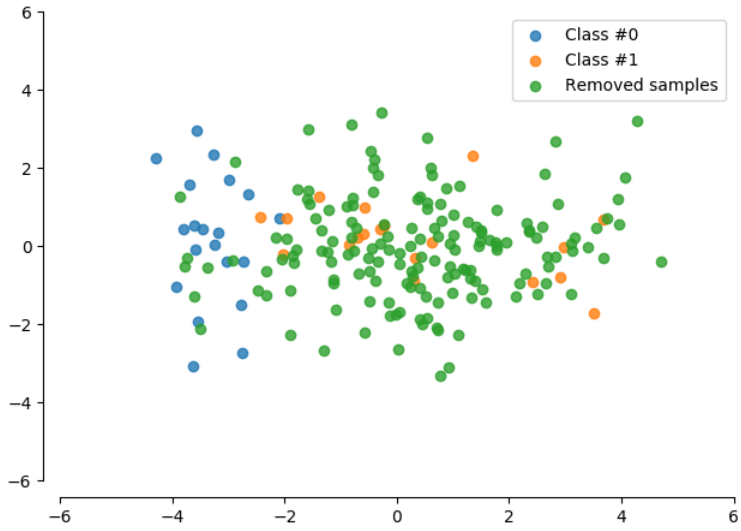


Figure 2.2: Random Undersampling (Lemaitre, 2016) [36]

2.2 Synthetic Minority Over Sampling Technique

When one class in the training set dominates the other, synthetic oversampling methods generate new data samples to restore balance in the data set. In this way, better learning performance is also achieved. Synthetic minority over sampling technique (SMOTE) is one of the most popular oversampling methods for dealing with imbalance data set. The aim of

SMOTE is to over-sample the minority class by creating "synthetic" data based on original samples rather than adding replicated data to the minority group [5].

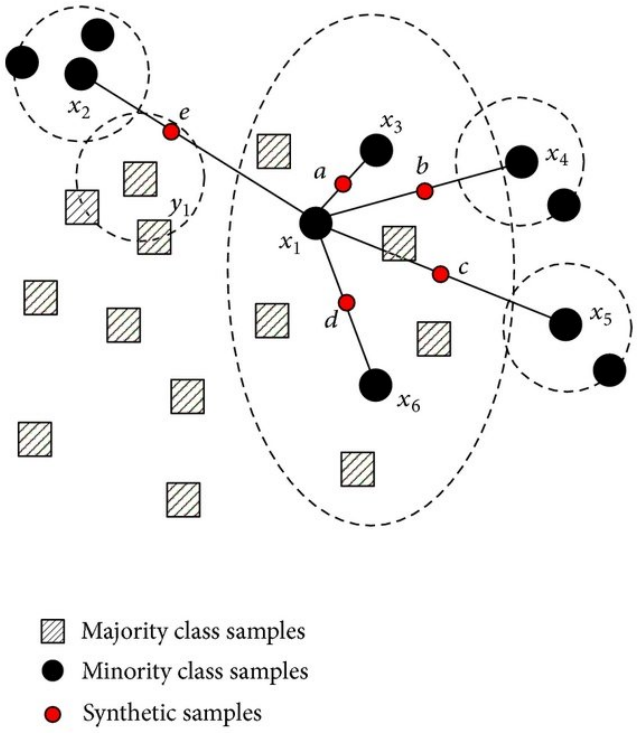


Figure 2.3: Synthetic data generation in SMOTE (Hu & Li, 2013) [24]

The SMOTE process is illustrated in Fig. 2.3 [24]. In contrast to random oversampling, SMOTE expands the minority samples in a way that benefits the learning process and it also minimize the overfitting problem to some degree. However, SMOTE still have to overcome various disadvantages such as generalization and variance issues [21].

The oversampling is done by taking each minority class sample and introducing corresponding synthetic examples. To generate synthetic samples, SMOTE focuses on the similarities among existing minority instances. For specific feature sample x_i in a feature set S, SMOTE find the K-nearest neighbors of x_i in the feature space. One of these K-nearest neighbors of x_i is randomly chosen to generate a synthetic sample and the euclidean distance between the feature vector and random neighbor is calculated, then multiplied by a random

number between 0 and 1. The result is added to the original feature vector [7].

$$x_{new} = x_i + (\hat{x}_i - x_i) * \delta \tag{2.1}$$

Going one step ahead from general SMOTE, Han et al. proposed `borderline-SMOTE` methods to generate synthetic samples on the borderline between two classes. On the other hand, SMOTE generates new synthetic samples along the line between the minority example and its selected nearest neighbors irrespective of their position in feature space. As the samples lying on the border are crucial to learn the class boundary, `borderline-SMOTE` outperform general SMOTE.

Suppose the whole training set is T, the minority class set is P and the majority class set is N. Set P has pn number of samples and set N has nn number of samples. In `borderline-SMOTE1`, for every member of set P, m nearest neighbors are calculated from the training set T. $m\prime$ denotes the number of majority sample among the m nearest neighbor where $0 \leq m\prime \leq m$. If all the m nearest neighbors of a minority sample are majority samples that means $m\prime = m$, then it can be considered as noise and discarded from the process of synthetics sample generation. If most of the m nearest neighbors are majority sample such as $m/2 \leq m\prime < m$ then this minority sample is a crucial example for classification as it can easily be misclassified. These crucial points are stored in a set called $DANGER$. Consider the element of $DANGER$ set as p_i. Now for every p_i, k-nearest neighbors are calculated from set P. Depending on the number of synthetic data needs to be generated, s number of neighbors are randomly selected for every p_i. Finally the differences dif_j between p_i and its s nearest member should be calculated. Then s new samples can be generated using the following formula.

$$synthetic_j = p_i + r_j * dif_j, j = 1, 2, ..., s \tag{2.2}$$

where r_j is a random number between 0 and 1. The following figures illustrate the selection of candidate minority sample for new synthetic data generation in `borderline-SMOTE1`.

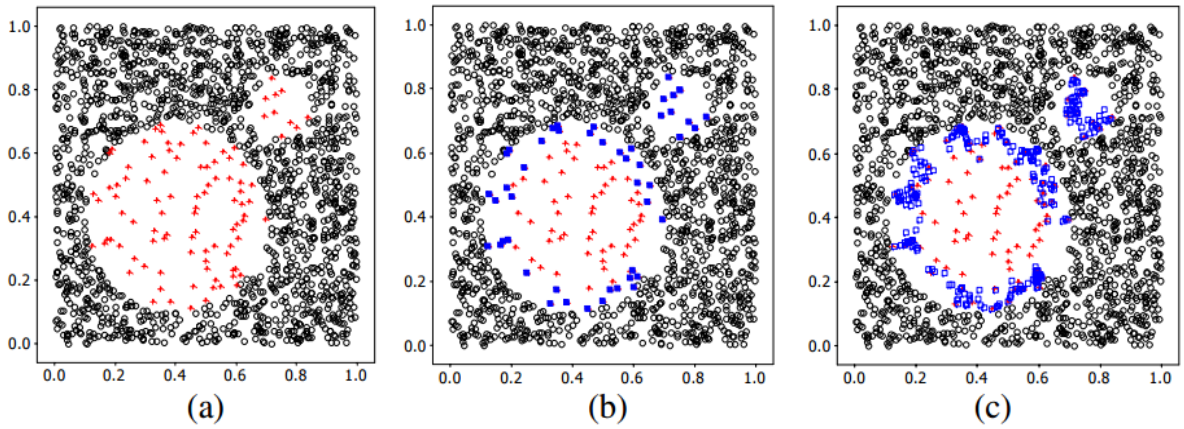


Figure 2.4: Candidate Selection and Synthetic data generation in `borderline-SMOTE1` (Han, Wang, & Mao et al., 2005) [19]

Besides generating synthetic samples that lie between the element of $DANGER$ set and their nearest neighbors from set P, `borderline-SMOTE2` creates new samples from nearest majority samples from set N. However, this time the value of r_j is selected randomly between 0 and 0.5 which ensures new generated samples are closer to the minority class.

2.3 Adaptive Synthetic Sampling

Adaptive Synthetic Sampling (ADASYN) incorporates an adaptive strategy to generate new data. When new data is going to be generated around minority class samples, ADASYN apply a weighted distribution for different minority class samples according to their level of difficulty in learning [20]. The key concept behind this is to generate more synthetic data for minority class samples that are harder to learn compared to those minority samples that are easier to learn. Difficulty of learning is determined by the number of majority class neighbors in the K-nearest neighbor. More majority class neighbor means more difficult to learn because it can be easily misclassified as majority class. Besides, reducing the bias introduced by the class imbalance ADASYN helps greatly to learn the decision boundary in the critical regions by adaptively shifting the classification decision boundary toward the difficult examples. The data generation policy is illustrated in Fig. 2.5 [35].

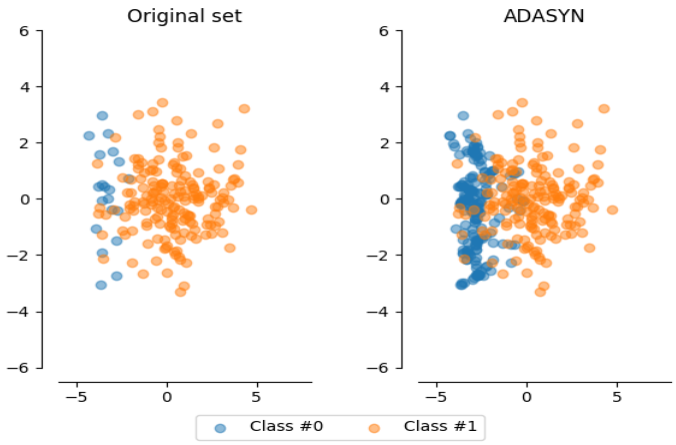


Figure 2.5: Synthetic data generation using ADASYN (Lemaitre, 2016) [35]

To understand the details of the procedure of ADASYN, consider the training data set T with m samples where the number of minority class samples is m_s and the number of majority class samples is m_l. Therefore, we have $m_s \leq m_l$ and $m_s + m_l = m$. Therefore, the degree of class imbalance will be

$$d = m_s/m_l$$

where $d \in (0, 1]$. ADASYN generates new data if the degree of class imbalance is less than a preset threshold, d_{th} denoting the maximum tolerated degree of class imbalance ratio. Thus, ADASYN allows data generation if

$$d < d_{th}$$

Synthetic data generation procedure of ADASYN can be described as follows:

1. The first step is to determine the number of synthetic samples required to generate for the minority class to achieve a certain level of balance. If G is the number of synthetic samples we want to generate then

$$G = (m_l - m_s) \times \beta$$

where $\beta \in [0, 1]$. The parameter β denotes the target balance level after synthetic data generation. When $\beta = 1$ means we want to make a fully balanced data set by making

m_l and m_s equal.

2. The next step is to find the K nearest neighbor for every sample x_i belongs to the minority class according to the Euclidean distance in n dimensional space. Determine the ratio α_i as follows:

$$\alpha_i = \delta_i / K, i - 1, ..., m_s$$

where δ_i represents the number of majority class samples found in the K nearest neighbor, therefore $\alpha_i \in [0, 1]$.

3. Normalize α_i as follows to make $\hat{\alpha}_i$ a density distribution $\sum_i \hat{\alpha}_i = 1$:

$$\hat{\alpha}_i = \alpha_i / \sum_{i=1}^{m_s} \alpha_i$$

4. In ADASYN the number of samples generated from a minority class instance will be different. Therefore, the next step is to determine that number for every minority sample x_i:

$$g_i = \hat{\alpha}_i \times G$$

where G is the same number defined in step 1 as the total number of synthetic sample we want to synthesize.

5. To generate g_i samples for every x_i belongs to the minority class ADASYN uses the same steps as SMOTE.

   Do the LOOP for 1 to g_i:

   (a) Randomly select one minority sample, x_{zi}, from the K-nearest neighbors for sample x_i.

   (b) Generate new sample:

$$s_i = x_i + (x_{zi} - x_i) \times \lambda$$

   where λ is a random number between $[0, 1]$ and $(x_{zi} - x_i)$ is the difference vector.

End LOOP

2.4 Variational Autoencoder

VAE is one of the most popular deep generative model to learn the complicated multi-dimensional data distribution such as images. It uses neural networks as function approximator and learns in an unsupervised fashion [9]. It can be considered as a probabilistic graphical model based on Bayesian inference where the model aims to learn the underlying probability distribution of the training data so that it can sample new data from that learned distribution. One of the important benefit of VAE is that it makes weak assumption on the data while the traditional generative models either require strong assumptions about the structure of the data or rely on computationally expensive inference procedures.

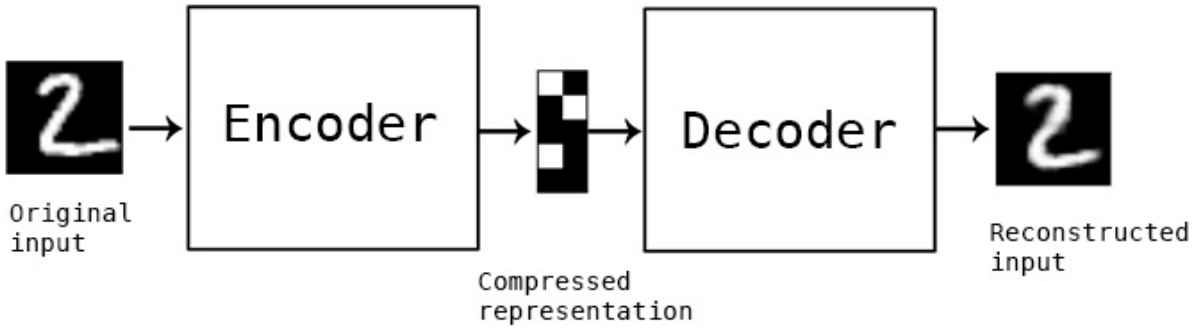


Figure 2.6: Vanilla Autoencoder Architecture (Chollet, 2016) [7]

VAE is an improved version of the vanilla autoencoder. An autoencoder network is basically a pair of two connected networks, an encoder and a decoder. An encoder network takes in an input, and converts it into a smaller, dense representation, which the decoder network can use to convert it back to the original input. The encoder produces an encoding that means a much smaller representation of the original data that contains enough information to characterize the data in hand. Generally, the encoder is trained together with additional parts of the network to produce encoding specifically useful for the required task.

The optimization during training is achieved using back-propagation. Encoding is nothing but a feature extraction process. Thus, it has a wide range of usage in CNN-based classifiers where nearly a couple of hundreds dimensional encoding are extracted from given input so that theyre particularly useful for classification. Encoder is bound to discard information as the encoding has far less dimension than the input. However, in the limited encoding the encoder tries to preserve as much of the relevant information as possible, and intelligently discard irrelevant portions. Autoencoders use encoder to generate encodings that are specifically useful for reconstruction of the input. The decoder takes the encoding as input and learns to properly reconstruct a full image just from that encoding. Together the encoder and the decoder form an autoencoder. The entire network - combination of encoder and decoder- is usually trained as a whole. The loss function is usually either the mean-squared error or cross-entropy between the output and the input, known as the reconstruction loss. As it is evident from the name, reconstruction loss penalizes the network for constructing outputs different from the input.

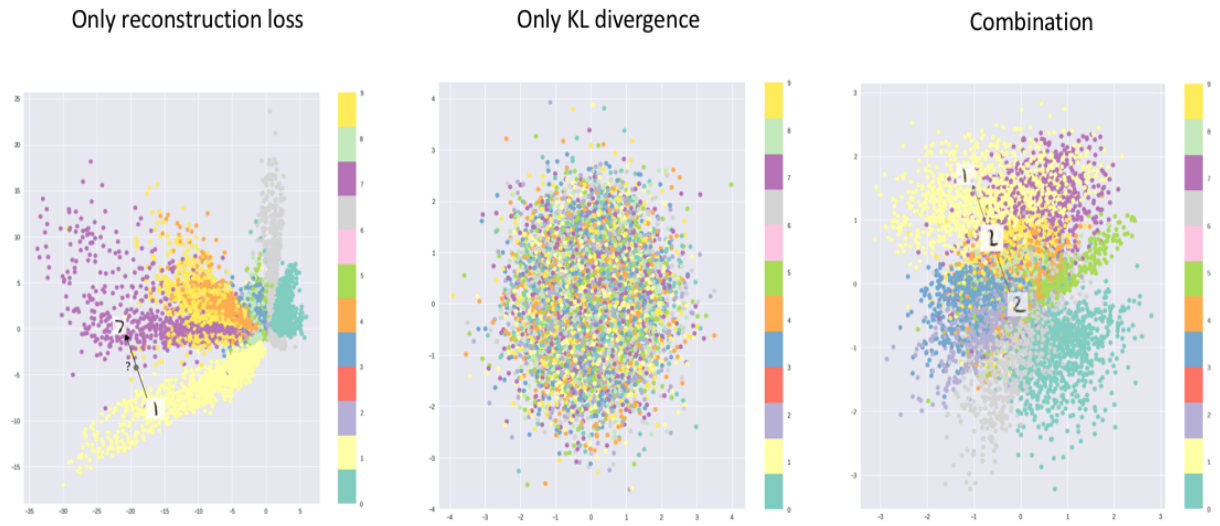


Figure 2.7: Visualization of encoding after training an autoencoder on the MNIST dataset (Shafkat, 2018) [42]

Formal assumption is that the original training data is generated from a low-dimensional latent representation which we called latent variable. These latent variables can hold mean-

ingful information about the output that model needs to generate. Problem with autoencoder is that the latent space where the encoded vectors lie may not be continuous. For example, the visualization of the encoding of MNIST data set in a 2D latent space shows that the encoding form distinct clusters [42]. Distinct encoding cluster for every class makes it easier to the decoder to reconstruct the actual image. This is particularly useful when the target is to replicate only the existing images. However, VAE fails when the target is to generate variation of images from random samples of a continuous latent space. This happens due the lack of knowledge about the specific latent space from where the latent variable is sampled.

From probabilistic perspective, VAE model defines a joint probability distribution over data x and latent variable z as

$$p(x, z) = p(x \mid z)p(z)$$

where the latent variables are drawn from a prior $p(z)$ and the data x have a likelihood $p(x \mid z)$ that is conditioned on latent variables z. Therefore, it can be considered as a decomposition of likelihood and prior. Under this generative process, aim is to maximize the probability of each data in x which is given as

$$p(x) = \int p(x \mid z)p(z)dz \tag{2.3}$$

According to Eq. 2.3, we need to find the distribution of latent variables and how to integrate Eq. 2.3 over all the dimensions of z. The integration needs to be carried over all the dimensions of z and is therefore intractable. The way out is to approximately maximize $P(x)$ in equation 2.1. To do that $P(z)$ is infer using $P(z|x)$ which is not known. $P(z|x)$ is inferred using variational inference. $P(z|x)$ is modeled using simpler distribution $Q(z|x)$ which is easy to find and the difference between $P(z|x)$ and $Q(z|x)$ is minimized using Kull-backLeibler(KL) divergence metric approach so that our hypothesis is close to the true

distribution. Then, the marginal likelihood of each sample in x can be written as

$$logP(x) = D_{KL}(Q(z \mid x) \mid\mid P(z \mid x)) + E_{z\sim Q}[logP(x \mid z) - logQ(z \mid x)] \tag{2.4}$$

The second term in Eq. 2.4 is the variational lower bound on the marginal likelihood of sample x. Eq. 2.4 can be rewritten as

$$logP(x) - D_{KL}(Q(z \mid x) \mid\mid P(z \mid x)) = -D_{KL}(Q(z \mid x) \mid\mid P(z)) + E_{z\sim Q}[logP(x \mid z)] \tag{2.5}$$

In this equation our target is to maximize the left hand side and the right hand side is convenient for optimization using gradient descent. Intuitively we are maximizing the marginal likelihood $logP(x)$ and minimizing the KL divergence between approximate posterior and the prior distribution. The approximate posterior is often chosen as a multivariate Gaussian with parameters $\mu(x)$ and $\sum(x)$ [47]. Fig. 2.8 shows a VAE architecture [42].

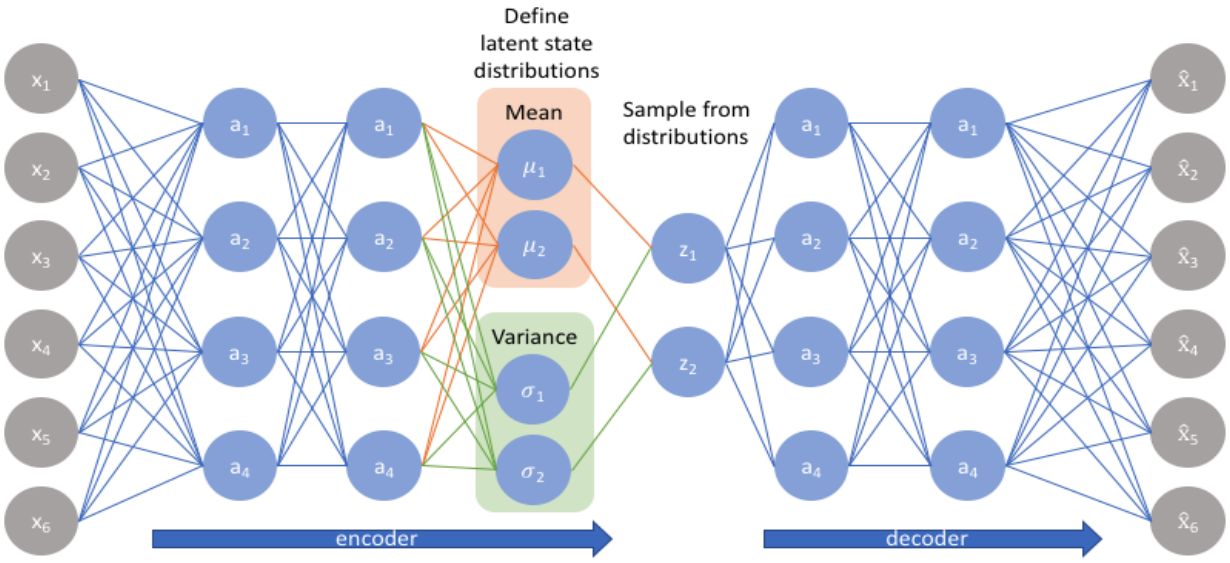


Figure 2.8: VAE Architecture (Shafkat, 2018) [42]

2.5 Generative Adversarial Network

In recent years, Generative Adversarial Network (GAN) proposed by Goodfellow et al. has brought an breakthrough in the generative model domain [17]. Like other generative model GAN also tries to learn the underlying distribution of the training data. A GAN is a combination of two networks: a generator network G and a discriminator network D. G generates new data instances by acquiring the knowledge of data distribution. Network D distinguish the real images from generated images. That means if an instance from true data set is presented to the discriminator, it has to recognize the instance as real and if an instance from fake or generated set is presented it has to recognize the instance as fake. Fig. 2.9 illustrates the basic structure of a GAN [45].

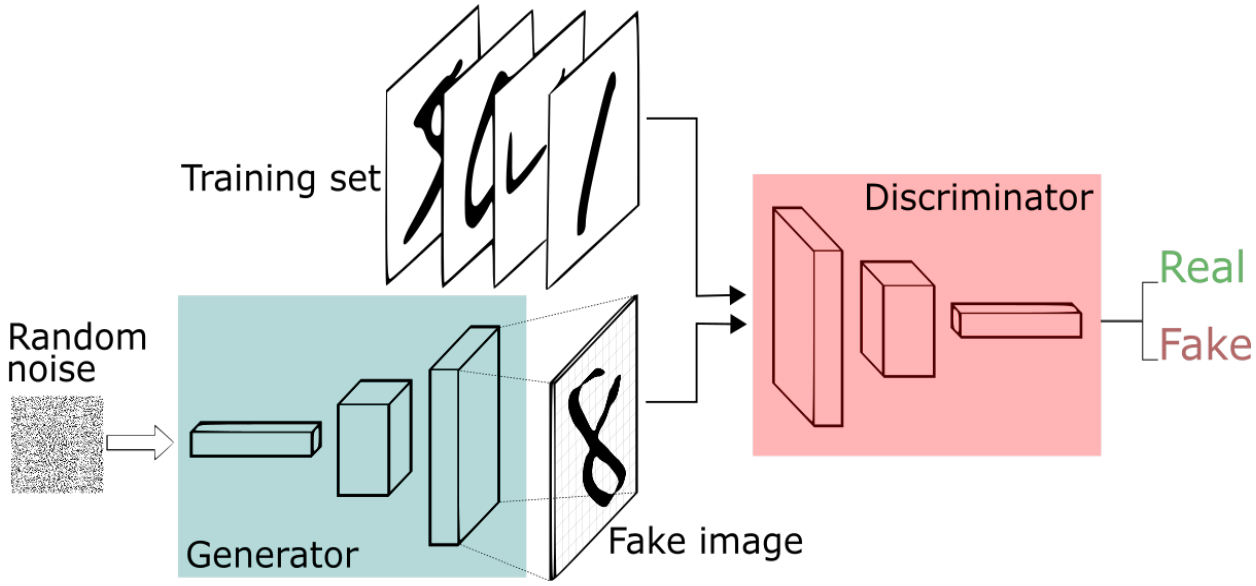


Figure 2.9: GAN Architecture (Silva, 2017) [45]

Thus, the steps a GAN takes can be described as follows without delving into the underlying mathematics (for which the interested reader is referred to [17]):

- The generator takes a random vector and return an image.
- The generated image is fed into the discriminator along with real images from true data set.

- The discriminator takes in both real and fake images and returns prediction as probabilities where 1 represents the image is from true set and 0 represents the image is from generated set.

Formally, the distribution of the generator P_G over data x is modeled as a differentiable function $G(z;\theta_g)$. The function $G(z;\theta_g)$ can be approximated by a neural network with parameters θ_g where the input is noise vector z. The discriminator function $D(x;\theta_d)$ is a function of sample x and outputs a single scalar representing the probability that x came from the real data distribution P_{data} rather than P_G. The function $D(x;\theta_d)$ can also be implemented by a neural network with parameters θ_d. The ultimate training target of GAN is to learn a generator distribution $P_G(x)$ that matches with $P_{data}(x)$ [16].

A double feedback loop controls the training of the two network. Network D is in a feedback loop with the true data while the generator is in a feedback loop with the discriminator. The objective function of the GAN is defined as

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{data}}[logD(x)] + \mathbb{E}_{x\sim noise}[log(1-D(G(z)))] \tag{2.6}$$

During training, the discriminator tries to make $D(G(z))$ equal to 0 and $D(x)$ equal to 1 which will maximize the associated probability of making correct decision by the discriminator. At the same time, the generator tries to make D(G(z)) equal to 1 to maximize the probability of discriminator making a mistake. Thus, essentially it is a mini-max two player game where both networks are trying to optimize a different and opposing objective function [17]. Also it is an example of an actor-critic model.

If the training process proceed batch wise and the batch size is m, then the parameters of the discriminator network need to be updated by ascending its stochastic gradient:

$$\nabla_{\theta_d}\frac{1}{m}\sum_{i=1}^{m}[logD(x^{(i)}) + log(1-D(G(z^{(i)})))] \tag{2.7}$$

The generator parameters need to be updated by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} [log(1 - D(G(z^{(i)})))] \tag{2.8}$$

During training one network tries to maximize some loss while the other tries to minimize the loss. As two networks try to optimize opposing functions, it is preferred to train against a static adversary. Therefore, while training the discriminator the generator values should be held constant; and while training the generator the discriminator value should be held constant. This provides a better read on the gradient that expresses the generator's preferences (utility and rewards). Also, at the start of the game, when the generator has not learned anything, the gradient is usually very small. Thus, pre-training the discriminator on true data before training the generator establishes a clearer gradient.

Another aspect that is important in GAN training is to set the learning rate of the two network such that both networks exhibit same skill level. Otherwise if the generator becomes too good, it will persistently exploit weaknesses in the discriminator that lead to false negatives. If the discriminator is too good, it will return values so close to 0 or 1 that the generator will struggle to read the gradient.

Recent GANs incorporate deep convolutional neural network architecture for both the generator and the discriminator, which has led to outstanding performance. Here are some examples of faces generated by DCGAN in Fig. 2.10 [41].

Figure 2.10: Images generated by DCGAN (Radford, Metz & Chintala et al., 2016) [41]

Chapter 3

Related Research

3.1 Previous Work On Data Sets Grouped in Batches

There are several existing approaches to reduce the negative effect when the data set is divided into batches. When there is available information about every batch, per batch characteristics are used to feature engineer. In case of lack of information about the batches, literature prefers to apply normalization techniques at the feature level to take benefit from every batch while avoiding bias towards any particular batch. When every batch has a balance, it is easier, at the same time fruitful to merge all of them.

In the SKICAT system [14], sky images were grouped into batches. Each set of images came from a single large digitized photographic plates. Therefore, every batch had different source plate from where the image regions were selected. The task was to classify the detected objects as stars, galaxies, or instrumental artifacts in sky images. Image processing routines were used to identify sky objects and to extract a set of basic features for each object. After that normalization was applied to some of the original features to reduce the effect of batch wise distribution of features.

Another approach to handle the batch issue is to view the individual batches in the training set as coming from a different context and use a context-sensitive learning algorithm [46] [49].

Kubat et al. also tried to address variation in batches where they had 9 different batches of oil spill images [30]. The images all come from the same satellite and the same general geographical location (strait of Juan de Fuca between Vancouver Island and the northern tip of Washington state) but the times when they were obtained are different thus one can expect that the images contain oil spills of different origin and of different types. The particular method they used is leave one batch out (LOBO). This is essentially the same as the traditional leave one out methodology except that one whole batch is left out on each iteration rather than just one example. There are also several approaches such as those of Burl et al. (1998) [3], Ezawa et al. (1996) [11], and Fawcett & Provost (1997) [13] that applied separation of batches used for training from those for testing.

3.2 Previous Work On Class Imbalance

The issue of class imbalance can be tackled on the level of the classifier. In such case, the learning algorithms are modified, e.g. by introducing different weights to misclassification of examples from different classes [52] or explicitly adjusting prior class probabilities [33]. As discussed in Section 1.2, we are not particularly interested in algorithm level solutions, rather our aim is to work with the data level methods that operate on training set and change its class distribution.

At the data level the most straightforward and common approach is the use of sampling methods. The basic version of it is called random minority oversampling, which simply replicates randomly selected samples from minority classes. It has been shown that oversampling is effective and robust [37], yet it can lead to over-fitting [5][48]. In case of image data set there is also some variation of random oversampling which replicates random images adding slight variations such as rotation, translation, blur, center cropping, contrast, sharpening etc. This strategy has been used for a while in a wide range of applications like plant leave classification [50], concealed cargo inspection [26], human disease detection [27].

A more advanced sampling method that aims to overcome this issue is SMOTE [5]. It augments artificial examples created by interpolating neighboring data points. Some ex-

tensions of this technique were proposed, for example focusing only on examples near the boundary between classes [19]. Another type of oversampling approach uses data preprocessing to perform more informed oversampling. DataBoost-IM, on the other hand, identifies difficult examples with boosting preprocessing and uses them to generate synthetic data [18]. ADASYN tries to sample more new data around difficult sample than simple minority samples [20]. Cluster-based oversampling first clusters the data set and then oversamples each cluster separately [29]. This way it reduces both between-class and within-class imbalance. An oversampling approach specific to neural networks optimized with stochastic gradient descent is class-aware sampling [43]. The main idea is to ensure uniform class distribution of each mini-batch and control the selection of examples from each class.

In 2014, Goodfellow et al. proposed generative adversarial networks (GAN) to generate high-dimensional data like images [17]. After its successful appearance, in the last couple of years a good number of different architectures were proposed for different type of image generation. They have already been used in generation of medical images [15][44], acoustic scenes [40], plant leaves [54] etc.

In recent years, some GANs have been proposed specifically to reduce class imbalance. Radford et al. (2016) [41] proposed and evaluated a set of constraints on the architectural topology of Convolutional GANs that make them stable to train in most settings. They named this class of architectures Deep Convolutional GANs (DCGAN). Zhu et al. (2017) [53] proposed CycleGAN which aims to create images with some particular emotions because in emotion classification some classes of emotions like disgusted are relatively rare comparing to other labels like happy or sad. Fanny & Cenggoro (2018) [12] proposed class expert generative adversarial network (CE-GAN) as the solution for imbalance data classification. They combined class specific data generation as an early step of a classifier where the classifier can be trained end-to-end given the imbalance data set. Data Augmentation Generative Adversarial Network (DAGAN) does not depend on the classes themselves and it can be applied to generate novel unseen classes of data [1]. As a result, DAGAN outperforms general GANs in the fewshot learning scenario. Mariani et al. tried to couple autoencoding and GAN [39]. Here the generative model learns useful features from majority classes and

uses these to generate images for minority classes. They applied class conditioning in the latent space to drive the generation process towards a target class. The generator in the GAN is initialized with the decoder of an autoencoder that enables to learn an close class conditioning in the latent space and to start the adversarial training from a more stable point helping to mitigate convergence problems arising with traditional GANs.

All the previous work generally compared their works with baseline or other works of the same category. For instance, oversampling-based techniques have been compared with other oversampling-based techniques while generative models are evaluated based on whether they can outperform the vanilla classifier trained on data set without augmentation. Until recently, there was no literature comparing the common old oversampling-based techniques with most recent GAN-based techniques. Zhang (2018) [51] attempted to compare sampling techniques with generative models with their proposed Extended Nearest Neighbor (ENN) based selection process to add the most relevant samples to the original imbalanced database. However, the study lacks in depth analysis of the training phase, encountered with sampling techniques and generative models.

Chapter 4

Methodology

4.1 Addressing Batchwise Class Imbalance

The first issue in our problem formulation is how to learn and experiment with batched examples. The critical feature is that, examples are naturally grouped in batches. Whenever data is collected in batches there is a possibility that the batches systematically differ from one another or that there is a much greater similarity of examples within a batch than between batches. The classier will be learned from one set of images and it will be applied on images that were not part of this set.

Among the approaches described in Section 3.1 normalization of features will not works best standing in the era of deep learning, where we do not do any hand engineering on features rather we learn them using deep convolutional neural networks. Also, context-sensitive learning does not fit perfectly as in most of the cases we are not aware of the context. Specially when there is a scarcity of the data or imbalance, even simple contextual techniques will not be successful. We do not always know what are the contextual parameters. Even when we know that in reality there is a contextual variable that influences the classifier e.g. the amount of lightening in a area, which can have impact on both case of the insulator texture or plant leaf texture, often we have no way to compute the value of this variable. Therefore, we decided not to pursue context-sensitive learning.

An alternative approach is to combine all the examples into one large data set in the hope that the learning algorithm will be able to detect systematic differences between batches and react by creating a disjunctive definition with a disjunct say for each batch. However, if the individual batches have a relatively small number of positive examples such a system will be prone to the problem of small disjuncts. Moreover, the batches that have many examples will dominate those that have few examples and also true for the within batch ratios.

We have to model both the within batch characteristics and the across batch characteristics. Also we simply do not have enough data or batches to do this with any certainty. Therefore, to make sure that our learning algorithm is not bias to any particular batch the best approach will be to make all the batches to have same level of balance and then merge all the batches to build a large data set. Thus our work will follow this approach.

The next thing that we need to decide is what will be that 'same level' of balance. While little systematic analysis of imbalance and methods to deal with are available for deep learning, researchers employ some methods that might be addressing the problem likely based on intuition. That is why we will take a qualitative approach rather than quantitative approach. Every variation of imbalance in every batch is natural. Therefore, we can select the highest ratio present in our observed reality as an standard for the more imbalanced set unless the highest ratio is too imbalance.

4.2 Selection of Data Augmentation Techniques

As a certain level of balance has to be achieved, we need to find a better way to augment data. Based on our review of the literature, the method most commonly applied in deep learning is oversampling. However, we have identified three overall category for augmentation - random oversampling, synthetic oversampling, and generative models. In our experiment we wished to select one method from each of the category to come to a final judgment at the end. To this end, we selected random oversampling, SMOTE, and GAN. As synthetic oversampling-based approaches are not easily applicable to higher dimension of data we will take some measure to use it for image data.

4.2.1 Random Oversampling

The Random oversampling method operates by replicating the randomly selected set of examples from the minority class, so that the majority class does not have overbearing presence during the training process. We chose random image every time from the original data set until we reach the required number of images to achieve the desired balance. I used the `RandomOverSampler` from the `imbalanced-learn` package of `scikit-learn` [34].

4.2.2 Adaptation of SMOTE

SMOTE algorithm creates artificial data based on the feature space similarities between existing minority class by introducing non-replicated minority class. It has been noted that although SMOTE seems to work well with low dimensional data, the effectiveness in the case of high-dimensional data is less impressive. This is due to the fact that SMOTE is not able to manage the bias in the majority class for the classifier where the data is high-dimensional.

Also in the deep learning setting where we learn the feature vector of an image data inside the classifier, this method is not particularly suitable. Even if we can extract the feature vector by any means from the image data and find the feature vector for a synthetic image using SMOTE, how can we construct the image from that feature vector without help of any generative model?

One way to circumvent this issue is to use the whole image as its feature vector. Thus, if the width of the image is w pixel, height of the image is h pixel, and there is 3 channels, then the size of the feature vector will be $c \times w \times h$. From here in the text, when we will mention SMOTE that will denote this particular adaptation of SMOTE.

4.2.3 Wasserstein GAN

As there is a good number of available GAN models, it is hard to choose one. In 2017, Arjovsky et al. introduced a form of GAN called Wasserstein-GAN, WGAN in short, an alternative to traditional GAN training [2]. This new model can improve the stability of learning, get rid of problems like mode collapse, and provide meaningful learning curves. In

their work, they concentrate on the various ways to measure how close the model distribution and the real distribution are, or equivalently, on the various ways to define a distance or divergence $\rho(P_\theta, P_r)$. The most fundamental difference between such distances is their impact on the convergence of sequences of probability distributions. WGAN minimizes a reasonable and efficient approximation of the Earth Mover (EM) distance which they named Wasserstein-1. Training WGANs does not require maintaining a careful balance in training of the discriminator and the generator, and does not require a careful design of the network architecture either.

In WGAN, they incorporated DCGAN with their Wasserstein-1 loss function. DCGAN is one of the best Deep Convolutional Generative Adversarial Network [41]. It takes as input 100 random numbers drawn from a uniform distribution and outputs an image of desired shape. The network consists of many convolutional, deconvolutional and fully connected layers. The network uses many deconvolutional layers to map the input noise to the desired output image. Batch Normalization is used to stabilize the training of the network. ReLU activation is used in generator for all layers except the output layer which uses tanh layer and Leaky ReLU is used for all layers in the Discriminator. Fig. 4.1 and Fig. 4.2 represent the DCGAN discriminator and generator network respectively [41].

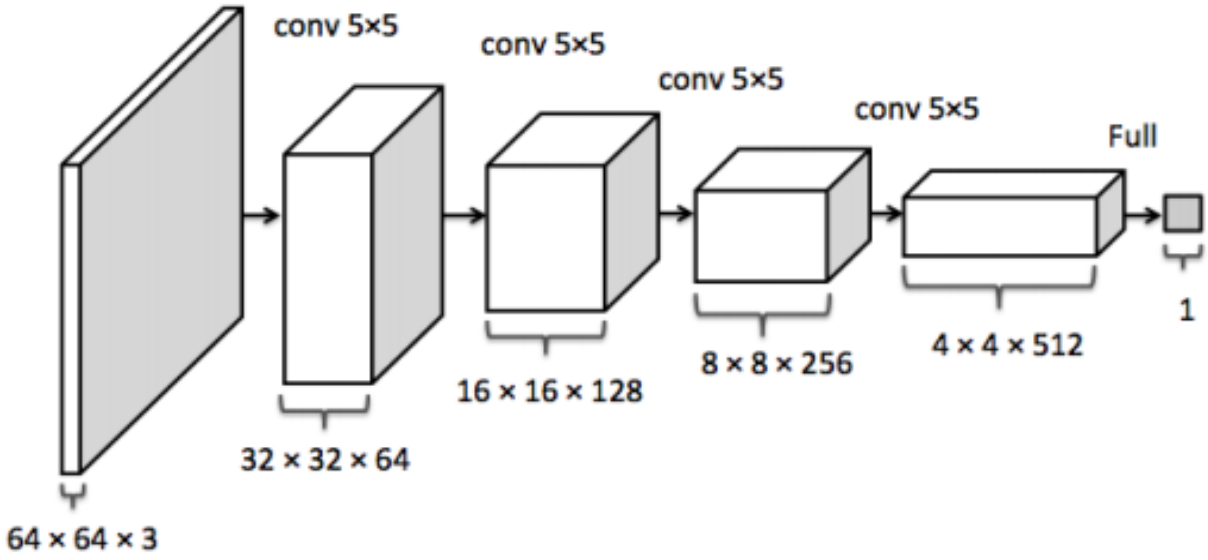


Figure 4.1: Architechture of DCGAN Discriminator (Radford, Metz & Chintala et al., 2016) [41]

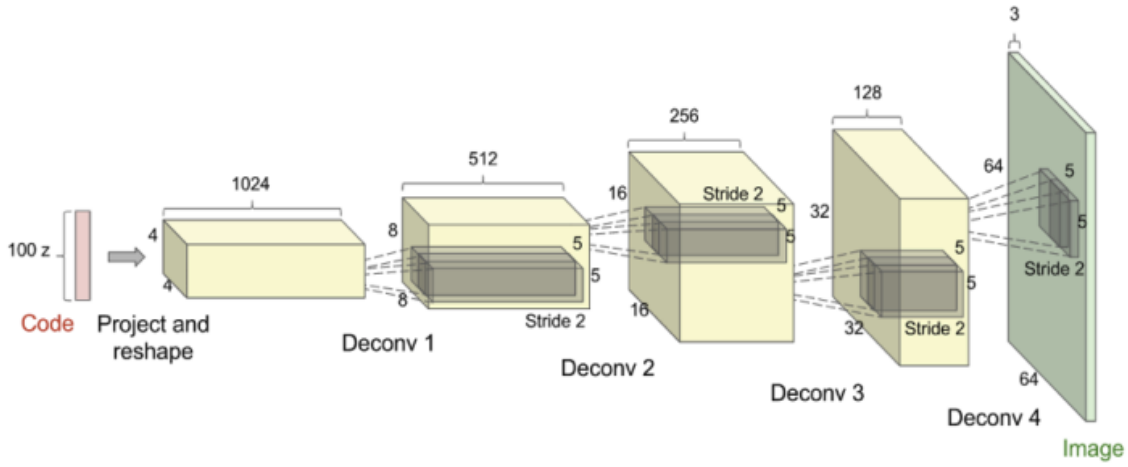


Figure 4.2: Architechture of DCGAN Generator (Radford, Metz & Chintala et al., 2016) [41]

4.3 Experimental Design

We designed experiments to evaluate the performance of our selected algorithms. We compared every result to the results of the baseline approach and to other selected approaches. We also performed an qualitative analysis on the image generated.

4.3.1 Data Set Selection

We tried to choose a data set that is analogous to the task of classifying damaged and undamaged insulator. In the insulator images, the task is to identify damaged insulator which has discoloration in the texture due to lightening or other effects. Fig. 4.3 shows a sample of electric tower image. From the image we need to isolate the insulators first to apply any classifier which adds an extra layer of difficulty not relevant to our main goal.

Keeping similarity with the insulator identification task, we picked up the detection task of infected tomato plant leaves. We selected the disease *Mosaic Virus* because it causes a damage similar to discoloration. Infected leaves have a yellowish texture on the original

green color. The *PlantVillage* data set exhibits imbalance across mosaic virus classes [25].

Figure 4.3: Example image from damaged insulator detection task (T. D. Witt, personal communication, 2019)

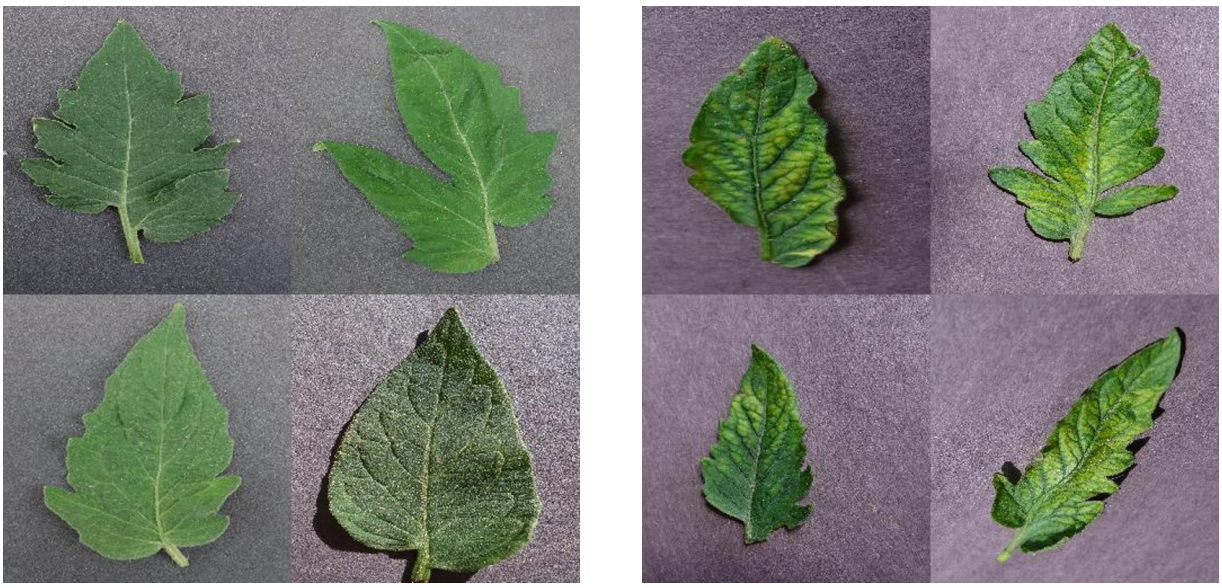
4.3.2 Data Set Preparation

In our tomato plant data set we have total 1590 healthy leaves and 370 infected leaves. Therefore, we considered healthy leaves as majority class and infected leaves as minority class. We artificially created two batches of same size having different level of imbalance. The more imbalance data set has approximately 5% infected leaves and 95% healthy leaves while the less imbalance batch has 33% infected leaves and 67% healthy leaves.

Table 4.1: Number of images in each batch

	Healthy	Infected	Total
More Balanced Set	660	320	980
Imbalanced Set	930	50	980

No preprocessing was required as the data is already clean. All the leaf images were captured against a similar grayish background. Fig. 4.4 presents some samples from both classes.



(a) Healthy Leaves (b) Infected Leaves

Figure 4.4: Examples of healthy and infected leaves (Hughes & Salathe, 2015) [25]

We used 20% of the data for testing. Also rest of the data was divided again into 80:20 ratio for training and validation set respectively. During cross-validation each division of data maintained the ratio mentioned in Table 4.1 for every batch.

4.3.3 Classifier Network

What type of classier should we use is one of the crucial decision to make. In a traditional classification task, a logistic regression works well when the data is linearly separable but fails to understand the non-linear relationship. SVM employs kernel tricks and maximal margin concepts to perform better in non-linear and high-dimensional tasks. Even a powerful SVM model, most of the times, benefit from the proper feature selection and feature extraction techniques. A single layer perceptron did not perform up to the expectations as it could only capture limited linear patterns, stacking two or more neural layers improved the performance significantly. A Convolutional Neural Network (CNN, or ConvNet) are a special kind of

multi-layer neural networks, designed to recognize visual patterns directly from pixel images with minimal preprocessing. Nowadays, for image classification and object recognition tasks, CNN-based approaches gained much popularity than any other approaches.

The ImageNet project, a large visual database designed for use in visual object recognition, runs an annual software contest, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), where proposed approaches compete to correctly classify and detect objects and scenes. According to the result of the ILSVRC, ResNet model outperforms all the existing models.

Table 4.2: Comparisons of top architectures from ILSVRC

Architecture	Num of Parameters	Top-1 Accuracy	Top-5 Accuracy	Year
AlexNet	61M	57.1	80.2	2012
VGG	138M	70.5	91.2	2013
Inception-V1	7M	69.8	89.3	2013
ResNet-50	25.5M	75.2	93	2015

ResNet incorporates residual module, as depicted in Fig. 4.5 which allows training of very deep networks without vanishing gradient and degradation problem [22]. It also features heavy batch normalization. ResNet were able to train a NN with even 152 layers while still having lower complexity than VGGNet. For our problem we have chosen ResNet50. This variant of ResNet has 50 layers. It achieves a top-5 error rate of 3.57%. Fig. 4.6 shows the details of the ResNet Architecture [8].

4.3.4 Experimental Setup

In K-fold cross-validation technique the data set is randomly divided into k subsets. The model is trained with k-1 subsets and tested on the remaining 1 subset. This process is repeated k number of times each time considering a different subset for testing. We used 5-fold cross-validation in our experiment.

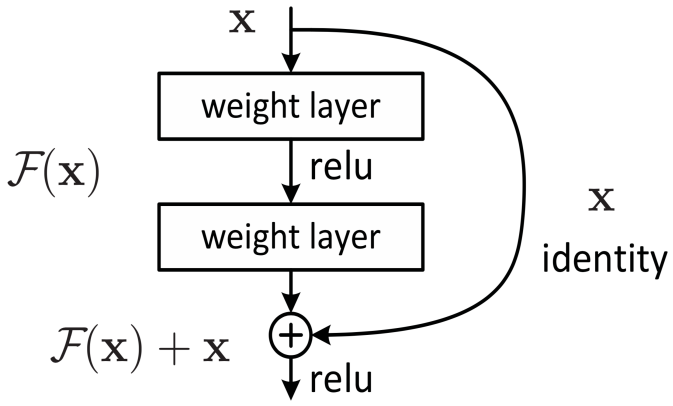


Figure 4.5: Residual Module (He, 2016) [22]

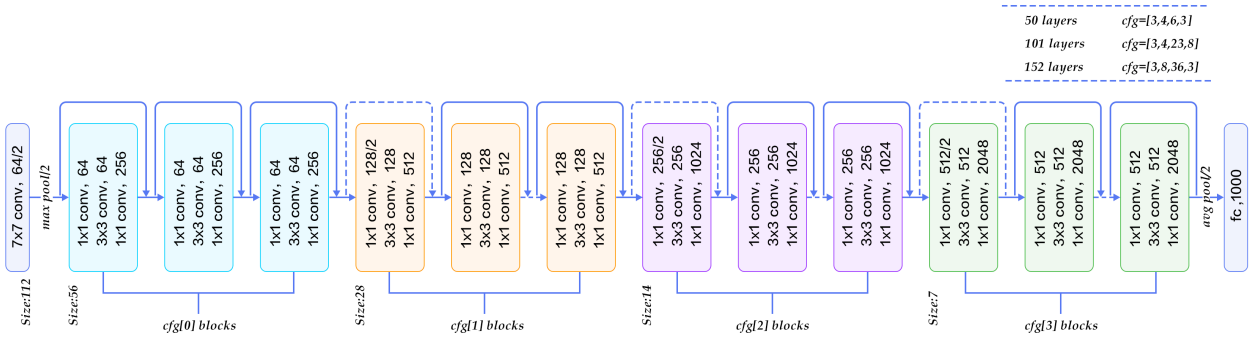


Figure 4.6: ResNet Architecture (Das, 2017) [8]

The other major parameters of the experiment are outlined as follows:

- Size of the images used: 64×64
- Number of channel in the image: 3
- Number of epoch for GAN training: 200,000
- Number of epoch for classifier training: 3,000
- Dropout Rate in classifier network: 0.2
- Learning Rate for the optimizer: 0.001

Chapter 5

Results and Experimental Studies

This chapter includes a brief description of the evaluation metrics. Besides presenting the values of the evaluation metrics, it illustrates the differences in the performance for each approach.

5.1 Evaluation Metrics

Measuring the performance of a classifier applied on imbalanced data using traditional metrics such as accuracy is difficult since it does not take into account the lower number of instances in the minority class. Metrics such as Precision and Recall have been used frequently for assessing the performance of a classifier in such cases. A combination of these measures, such as F-measure, are single class focus metrics that use different combinations of specificity and sensitivity of the classifiers to give a better indication of performance. Ranking order metrics such AUC measure assess a classifiers performance over all imbalance ratios and hence provide a summary of the entire range.

True Positives (TP): They are the correctly predicted positive values which means the classes that are "positive" are predicted to be "positive".

True Negatives (TN): They are the correctly predicted negative values which means the classes that are "negative" are predicted to be "negative".

False Positives (FP): They are the incorrectly predicted positive values which means the classes that are "negative" are predicted to be "positive".

False Negatives (FN): They are the incorrectly predicted negative values which means the classes which are "positive" are predicted to be "negative"

Accuracy:

$$Precision = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

Precision:

$$Precision = \frac{TP}{TP + FP} \tag{5.2}$$

Recall:

$$Recall = \frac{TP}{TP + FN} \tag{5.3}$$

F-measure or F-score:

$$Precision = \frac{(1 + \beta^2)\ Precision.Recall}{\beta^2.\ Recall + Precision} \tag{5.4}$$

where β is a weight coefficient to adjust the significance of recall (usually $\beta = 1$). When $\beta = 1$, it is generally named as F1 score.

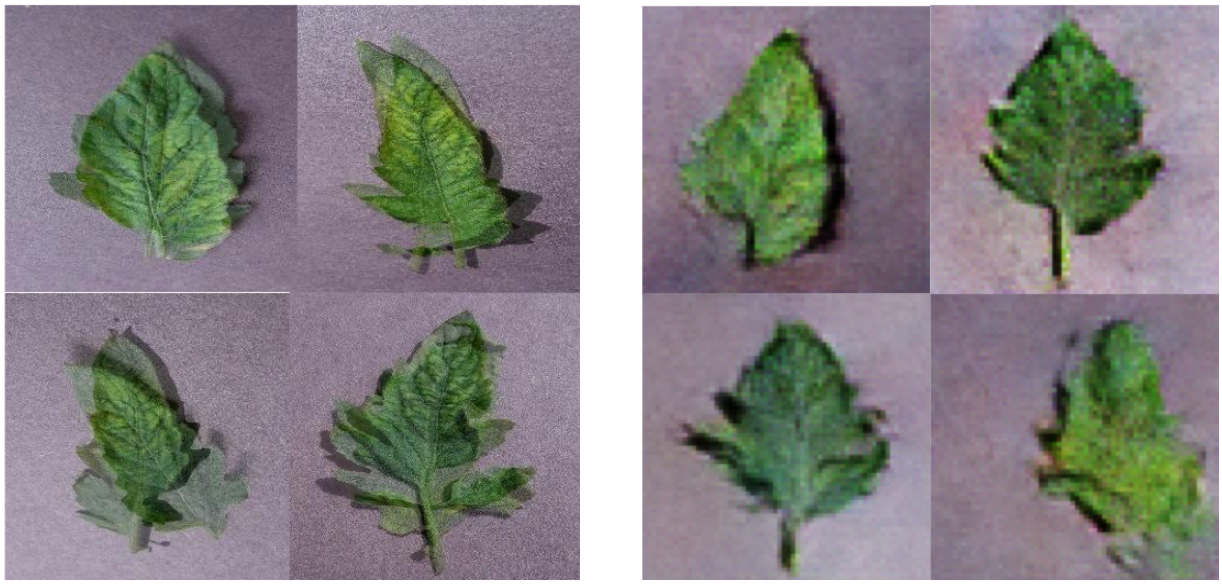
Area Under the Curve (AUC) Area Under the Curve (AUC) is an evaluation method independent of selected threshold and prior probabilities. It measures the probability of the classifier assigning a higher rank to a randomly chosen positive example than a randomly chosen negative example and represents the performance of a classifier averaged over all possible cost ratios. Some limitations of this measure may be noted with different classifiers, for comparative purposes due to the skew-ratio distribution and interpretability.

5.2 Quality of Augmented Data

The quality of the image will have a significant effect on the performance of the classifier. There is no way to quantitative judge the quality rather than to do that qualitatively. One can think of the following issues while determining the quality of the generated images.

- Generated images must represent the desired class.
- Generated images must not be repetitive.
- Generated images must be different from the real ones already available in the training set.

The images generated by the SMOTE are appeared to be little blurry in the edges. As this algorithms pick an value between two similar data point, the images looks like overlapping of two image. On the other hand, images generated by GAN are more clear and sharp. The shape of the leaves are perfect in most of the times. They seem to have a lot of variation. Fig. 5.1 shows the quality of the generated images.



(a) Generated by SMOTE (b) Generated by GAN

Figure 5.1: Examples of generated leaf image

5.3 Performance on Test Data

Table 5.1 shows the value of the performance metrics for different approaches. These results are based on 5-fold cross-validation and the high-lighted results are the best in each metrics. It illustrates that the GAN-based method outperforms all other methods evaluated by different metrics. These results also show that the GAN-based method eliminates the influence of the skewed data distribution.

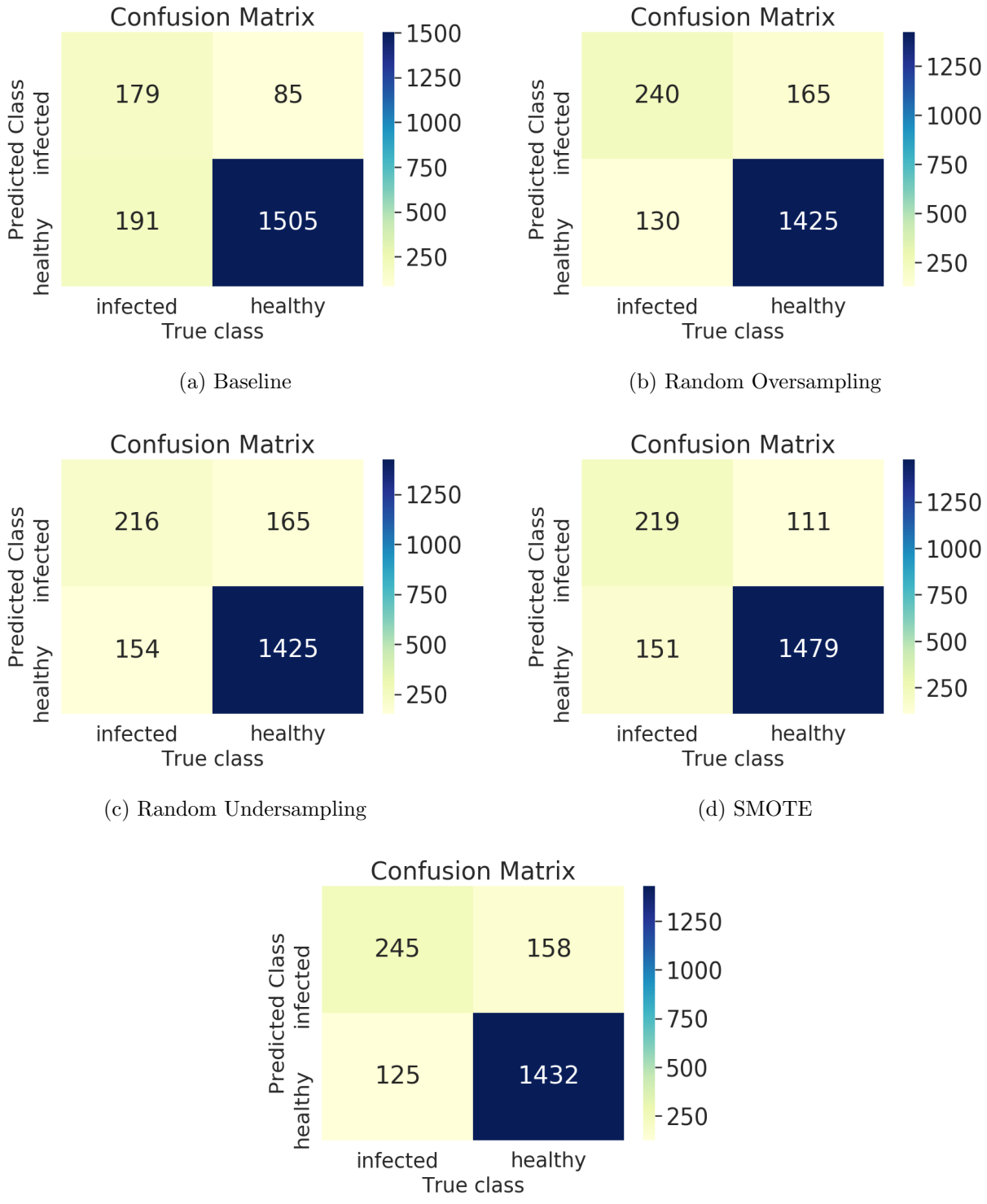
To demonstrate the credibility of our experimental results, we ran 5-fold cross-validation, and conduct the Students t-test to show significant performance difference exists between the GAN-based approach and the baseline. Table 5.2 presents all the p values determined for each metric. The p-values for Recall and F1-score are less than 0.05, which suggests that the means are significantly different at the 95% level of confidence.

Table 5.1: Evaluation metrics for each approach (Average for 5-fold cross-validation)

Approach Name	Accuracy	Precision	Recall	F1	AUC
Baseline	0.859184	**0.695476**	0.483784	0.562106	0.883044
Random Oversampling	0.849490	0.606722	0.648649	0.620203	0.866607
Random Undersampling	0.837245	0.598592	0.583784	0.582286	0.855682
SMOTE	**0.866327**	0.664705	0.591892	0.625911	0.887723
GAN-based	0.855612	0.613628	**0.662162**	**0.632928**	**0.892504**

Table 5.2: Significance test for baseline and GAN-based method for 5-fold cross-validation with 95% confidence level

	Accuracy	Precision	Recall	F1	AUC
P-value	0.35	0.11	**0.02**	**0.01**	0.25



(a) Baseline

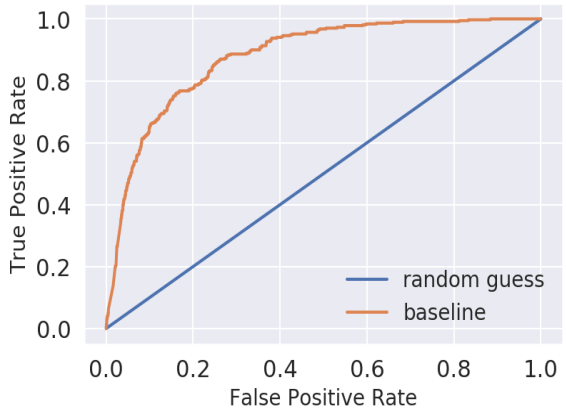
(b) Random Oversampling

(c) Random Undersampling

(d) SMOTE

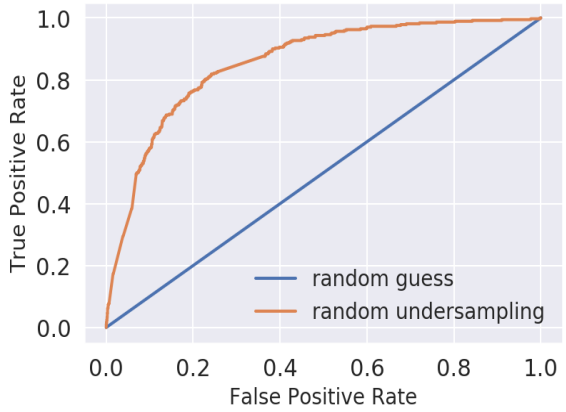
(e) GAN-based

Figure 5.2: Aggregated Confusion Matrix for each approach after 5-fold cross-validation

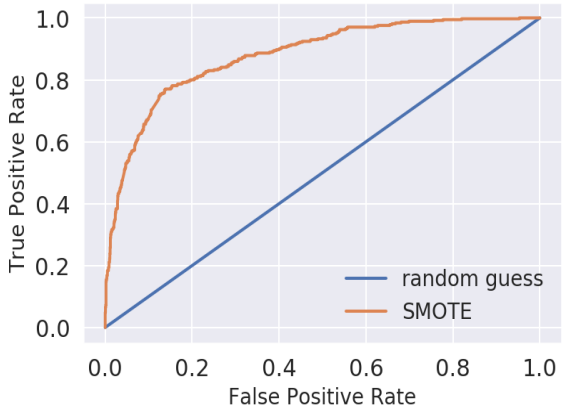


(a) Baseline

(b) Random Oversampling



(c) Random Undersampling



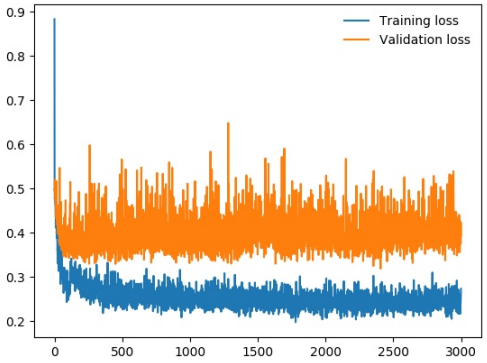
(d) SMOTE

(e) GAN-based

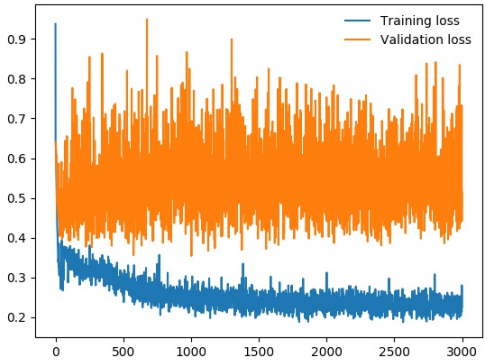
Figure 5.3: Aggregated ROC curve for each approach after 5-fold cross-validation

5.4 Comparative Analysis

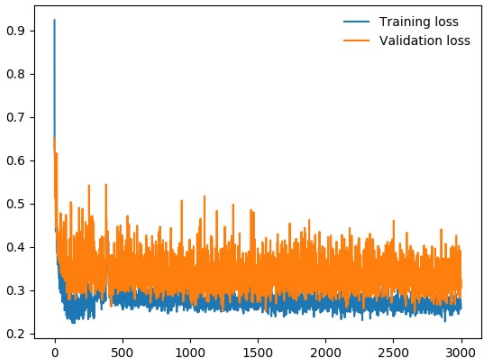
From Table 5.1 it is clear that SMOTE achieves the best accuracy. However, as discussed earlier, accuracy is not a better metric for imbalanced data set. The baseline wins over all in terms of precision. Therefore, the rate of misclassifying negative examples are less for baseline. As the baseline has more negative samples it is reasonable to do so. In terms of other three metric - recall, f1-score, and AUC - the GAN-based approach is performing well compared to any other approach. Random undersampling failed to perform any good in terms of all of the metrics as it lost some information due to the undersampling.



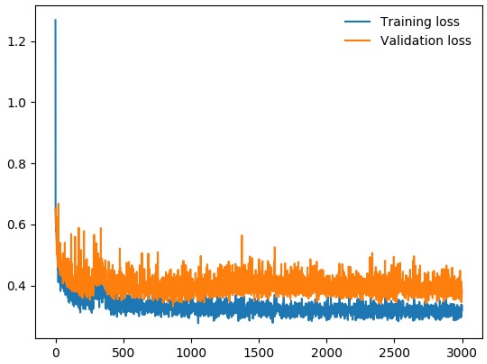
(a) Baseline



(b) Random Oversampling



(c) SMOTE



(d) GAN-based

Figure 5.4: Loss function plot for different approaches

The recall for GAN-based approach is significantly higher than the baseline and slightly

better than random oversampling-based approach. Therefore, we can say GAN-based approach is good at detecting the positive class which is our prime objective. Also, the F1-score and AUC is greater for the GAN-based approach which means it maintains a good balance in detecting both class.

Also we noticed that GAN-based approach helps to avoid overfitting. Overfitting means the model got more biased towards the training data set. This can be determine by looking at the values of loss function for both the training and validation set at at the training time. If the training loss is much lower than the validation loss we can conclude that the model is getting a good hand on classifying the training set but failing to apply that knowledge on the validation set. SMOTE-based approaches are also good at avoiding overfitting as they also adds different sample than what exists in the original data set. Random oversampling-based approach is the one that is more prone to overfitting. Fig. 5.4 presents the training and validation loss function for different approaches.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

We performed a series of experiments to investigate the performance of different data augmentation techniques. Our study shows that a GAN-based generative approach can produce more clear and meaningful image samples. Based on our analysis from Chapter 5, we can conclude that deep generative models, such as GAN, could be implemented as image generator to compensate the skewed data distribution, which can capture the distribution of original data more accurately. The variation in the samples introduced by a GAN-based approach make it easier for classifier to find classification boundaries. Thus, a GAN-based approach outperforms other traditional random oversampling, random undersampling, and feature space oversampling methods. Previous work on generative models demonstrated the superiority of GAN-based approach for class imbalance on simple structured image data sets, such as MNIST, NIST19 and Fashion MNIST but presented a doubt in case of complicated and critical data domain. Our work invalidates the doubt and clearly indicates the possibility of using generative models as a data augmentation tool for critical classification tasks like visual anomaly detection.

6.2 Future Work

One major topic of future work is to further investigate the GAN-based approach on other critical visual anomaly detection tasks. In future, We wish to compare the GAN-based approach to other generative approaches such as VAE. During our experiment we noticed there is a significant effect of learning rate on learning from imbalanced data set. Slight increase in learning rate can cause drastic performance reduction which is not usual in case of balanced data set. This effect can be due to the particular data set or can be a general issue. Therefore, our future work will include exploring further by using different learning rate on diverse data set. Last but not least, further work on developing more effective and computationally more efficient algorithms for data augmentation is highly desirable.

Bibliography

[1] Antoniou, A., Storkey, A., & Edwards, H. (2018). Data augmentation generative adverdarial networks. Retrieved from https://arxiv.org/abs/1711.04340.

[2] Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein gan. Retrieved from https://arxiv.org/abs/1701.07875.

[3] Burl, M. C., Asker, L., Smyth, P., Fayyad, U., Perona, P., Crumpler, L., & Aubele, J. (1998). Learning to recognize volcanoes on venus. *Machine Learning*, 30(2-3), 165–194.

[4] Chawla, N. V. (2009). Data mining for imbalanced datasets: An overview. *Data mining and knowledge discovery handbook, Springer*, 875–886.

[5] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 321–357.

[6] Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, 6(1), 1–6.

[7] Chollet, F. (2016). Building autoencoders in keras. Retrieved from https://blog.keras.io/building-autoencoders-in-keras.html.

[8] Das, S. (2017). Cnn architectures: Lenet, alexnet, vgg, googlenet, resnet and more . Retrieved from https://medium.com/@sidereal/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5.

[9] Doersch, C. (2016). Tutorial on variational autoencoders. Retrieved from https://arxiv.org/abs/1606.05908.

[10] Estabrooks, A., Jo, T., & Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20(1), 18–36.

[11] Ezawa, K. J., Singh, M., & Norton, S. W. (1996). Learning goal oriented bayesian networks for telecommunications management. *Thirteenth International Conference on Machine Learning*, 139–147.

[12] Fanny & Cenggoro, T. W. (2018). Deep learning for imbalance data classification using class expert generative adversarial network. *Procedia Computer Science*, 135, 60–67. doi:https://doi.org/10.1016/j.procs.2018.08.150.

[13] Fawcett, T. & Provost, F. (1997). Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1(3), 291–316.

[14] Fayyad, U. M., Weir, N., & Djorgovski, S. (1993). Skicat: A machine learning system for automated cataloging of large scale sky surveys. *Tenth International Conference on Machine Learning*, 112–119.

[15] Frid-Adar, M., Klang, E., Amitai, M., Goldberger, J., & Greenspan, H. (2018). Synthetic data augmentation using gan for improved liver lesion classification. *IEEE 15th International Symposium on Biomedical Imaging*, 289–293.

[16] Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. Retrieved from https://arxiv.org/abs/1701.00160.

[17] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, 2672–2680.

[18] Guo, H. & Viktor, H. L. (2004). Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *ACM Sigkdd Explorations Newslette*, 6(1), 30–39.

[19] Han, H., Wang, W.-Y., & Mao, B.-H. (2005). Borderline-smote: a new over-sampling method in imbalanced data sets learning. *Advances in Intelligent Computing*, 878–887.

[20] He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. *IEEE International Joint Conference on Neural Networks*, 1322–1328. doi:10.1109/IJCNN.2008.4633969.

[21] He, H. & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21, 1263–1284.

[22] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

[23] Hsu, W. & Carraway, K. (2018). Image analysis and data science: Aerial survey of street insulators. *Black & Veatch Corporation*.

[24] Hu, F. & Li, H. (2013). A novel boundary oversampling algorithm based on neighborhood rough set model: Nrsboundary-smote. Retrieved from https://www.researchgate.net/publication/287601878_A_Novel_Boundary_Oversampling_Algorithm_Based_on_Neighborhood_Rough_Set_Model_NRSBoundary-SMOTE/figures.

[25] Hughes, D. P. & Salathe, M. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. Retrieved from https://arxiv.org/abs/1511.08060.

[26] Jaccard, N., Rogers, T. W., Morton, E. J., & Griffin, L. D. (2017). Detection of concealed cars in complex cargo x-ray imagery using deep learning. *Journal of X-Ray Science and Technology*, 25(3), 323–339.

[27] Janowczyk, A. & Madabhushi, A. (2016). Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases. *Journal of Pathology Informatics*, 7, 29.

[28] Japkowicz, N. & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5), 429–449.

[29] Jo, T. & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter*, 6(1), 40–49.

[30] Kubat, M., Holte, R. C., & Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar image. *Machine Learning*, 30, 195–215.

[31] Kukar, M. & Kononenko, I. (1998). Cost-sensitive learning with neural networks. *13th European Conference on Artificial Intelligence (ECAI)*, 445–449.

[32] Laurikkala, J. (2001). Improving identification of difficult small classes by balancing class distribution. *Artificial Intelligence in Medicine (AIME). Lecture Notes in Computer Sciencee, Springer*, 2101, 63–66.

[33] Lawrence, S., Burns, I., Back, A., Tsoi, A. C., , & Giles, C. L. (1998). Neural network classification and prior class probabilities neural networks: Tricks of the trade. *Springer*, 299–313.

[34] Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17), 1–5.

[35] Lemaitre, G., Nogueira, F., Oliveira, D., & Aridas, C. (2016a). Adasyn. an illustration of the adaptive synthetic sampling approach for imbalanced learning adasyn method. Retrieved from http://glemaitre.github.io/imbalanced-learn/auto_examples/over-sampling/plot_adasyn.html.

[36] Lemaitre, G., Nogueira, F., Oliveira, D., & Aridas, C. (2016b). Random under-sampling. an illustration of the random under-sampling method. Retrieved from http://glemaitre.github.io/imbalanced-learn/auto_examples/under-sampling/plot_random_under_sampler.html.

[37] Ling, C. X. & Li, C. (1998). Data mining for direct marketing: Problems and solutions. *KDD*, 98, 73–79.

[38] Ling, C. X. & Sheng, V. S. (2008). Cost-sensitive learning and the class imbalance problem. *Encyclopedia of Machine Learning, Springer*.

[39] Mariania, G., Scheidegger, F., Istrate, R., Bekas, C., & Malossi, C. (2018). Bagan: Data augmentation with balancing gan. Retrieved from https://arxiv.org/abs/1803.09655.

[40] Mun, S., Park, S., Han, D. K., & Ko, H. (2017). Generative adversarial network based acoustic scene training set augmentation and selection using svm hyper-plane. *Detection and Classification of Acoustic Scenes and Events*, 93–97.

[41] Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. Retrieved from https://arxiv.org/abs/1511.06434.

[42] Shafkat, I. (2018). Intuitively understanding variational autoencoders. Retrieved from https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf.

[43] Shen, L., Lin, Z., & Huang, Q. (2016). Relay backpropagation for effective learning of deep convolutional neural networks. *European Conference on Computer Vision. Lecture Notes in Computer Science, Springer*, 9911, 467–482.

[44] Shin, H.-C., Tenenholtz, N. A., Rogers, J. K., Schwarz, C. G., Senjem, M. L., Gunter, J. L., Andriole, K. P., & Michalski, M. (2018). Medical image synthesis for data augmentation and anonymization using generative adversarial networks. *Simulation and Synthesis in Medical Imaging*, 11037, 1–11.

[45] Silva, T. (2017). A short introduction to generative adversarial networks. Retrieved from https://sthalles.github.io/intro-to-gans.

[46] Turney, P. D. (1993). Exploiting context when learning to classify. *Proceedings of the European Conference on Machine Learning*, 402–407.

[47] Wan, Z., Zhang, Y., & He, H. (2017). Variational autoencoder based synthetic data generation for imbalanced learning. *IEEE Symposium Series on Computational Intelligence (SSCI)*, 1–7.

[48] Wang, K.-J., Makond, B., Chen, K.-H., & Wang, K.-M. (2014). A hybrid classifier combining smote with pso to estimate 5-year survivability of breast cancer patients. *Applied Soft Computing*, 20, 15–24.

[49] Widmer, G. & Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1), 69–101.

[50] Zhang, C., Zhou, P., Li, C., & Liu, L. (2015). A convolutional neural network for leaves recognition using data augmentation. *IEEE International Conference on Computer and Information Technology*, 2143–2150.

[51] Zhang, Y. (2018). Deep generative model for multi-class imbalanced learning. *Open Access Master's Theses. Paper 1277.*

[52] Zhou, Z.-H. & Liu, X.-Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18, 63–77.

[53] Zhu, X., Liu, Y., Qin, Z., & Li, J. (2017). Emotion classification with data augmentation using generative adversarial networks. Retrieved from https://arxiv.org/abs/1711.00648.

[54] Zhu, Y., Aoun, M., Krijn, M., & Vanschoren, J. (2018). Data augmentation using conditional generative adversarial networks for leaf counting in arabidopsis plants. *Computer Vision Problems in Plant Phenotyping.*

[55] Zimek, A. & Schubert, E. (2017). Outlier detection. *Encyclopedia of Database Systems, Springer New York*, 1–5. doi:10.1007/978-1-4899-7993-3_80719-1.