Learning from small and imbalanced dataset of images using generative adversarial neural networks.

ALI-GOMBE, A.

2019

Learning from Small and Imbalanced Dataset of Images using Generative Adversarial Neural Networks

Adamu Ali-Gombe

A report submitted as part of the requirements for the degree
of PhD
at the School of Computing
Robert Gordon University
Aberdeen, Scotland

December 2019

Supervisor Dr. Eyad Elyan

Abstract

The performance of deep learning models is unmatched by any other approach in supervised computer vision tasks such as image classification. However, training these models require a lot of labeled data which are not always available. Labelling a massive dataset is largely a manual and very demanding process. Thus, this problem has led to the development of techniques that bypass the need for labelling at scale. Despite this, existing techniques such as transfer learning, data augmentation and semi-supervised learning have not lived up to expectations. Some of these techniques do not account for other classification challenges such as class-imbalance problem. Thus, mostly underperforming than fully supervised approaches.

In this thesis, we propose new methods to train a deep model on image classification with limited number of labeled examples. This was achieved by extending state-of-the-art generative adversarial networks with multiple fake classes and network switchers. These new features enabled us to train a classifier using large unlabeled data while generating class specific samples. The proposed model is label agnostic and is suitable for different classification scenarios ranging from weakly supervised to fully supervised settings. This was used to address classification challenges with limited labeled data and class-imbalance problem.

Extensive experiments were carried out on different benchmark datasets. Firstly, the proposed approach was used to train a classification model, and our findings indicated that the proposed approach achieved better classification accuracies, especially when the number of labeled samples is small. Secondly, the proposed approach was able to generate high-quality samples from class-imbalance datasets. The samples quality is evident in improved classification performances when generated samples were used in neutralising class-imbalance. The results are thoroughly analyzed and overall, our method showed superior performances over popular resampling technique and AC-GAN model. Finally, we successfully applied the proposed approach as a new augmentation technique to two challenging real-world problems, namely, face with attributes and

legacy engineering drawings. The results obtained demonstrate that the proposed approach is effective even in extreme cases.

keywords: Deep Learning, Image Classification,Generative Adversarial Networks (GAN).

List of Abbreviations

AC-GAN Auxiliary Classifier GAN

ALI-GAN Adversarially Learned Inference GAN

BAGAN Balancing GAN

BI-GAN Bi-directional GAN

BEGAN Boundary Equilibrium GAN

CCGAN Context Conditional GAN

CDBN Convolutional Deep Belief Networks

CGAN Conditional GAN

CLR Class Rectification Loss

CNN Convolutional Neural Network

COGAN coupled GAN

DAGAN Data Augmentation GAN

DBN Deep Belief Networks

DCGAN Deep Convolution GAN

EBGAN Energy Based GAN

EGAN Evolutionary GAN

FID Freschet Inception Distance

FM-GAN Fine-grained Multi-attribute GAN

FSC-GAN Few Shot Classifier GAN

GAN Generative Adversarial Networks

GMM Gaussian Mixture Model

IS Inception Score

IPM Integral Probability Metric

JSD Jensen Shannon Divergence

KL-Divergence Kullback-Leibler Divergence

LFW Labeled Faces in Wild

LapGAN Laplacian GAN

LMLE Large Margin Local Embedding

LSGAN Least Square GAN

LSTM Long Short Term Memory

MFC-GAN Multiple Fake Class GAN

MLP Multi-Layer Perceptron

MMD Maximum Mean Discripancy

NCE Noise Contrastive Estimation

NN Neural Network

PCA Principal Component Analysis

PandID Process and Instrumentation Diagram

ResNet Residual Network

SAGAN Self-Attention GAN

SGAN Semi-supervised GAN

SGD Stochastic Gradient Descend

SIFT Scale Invariant Feature Transform

SMOTE Synthetic Minority Oversampling Technique

SURF speed Up Robust Features

SSIM Structured Similarity Index Metric

S^2**GAN** Style and Structure GAN

SVHN Street View House Number

SVM Support Vector Machines

TAC-GAN Tex Auxiliary Classifier GAN

VAC-GAN Versatile Auxiliary Classifier GAN

VAE Variational Auto-Encoder

VGG Visual Geometry Group

WGAN Wasserstein GAN

Acknowledgements

All praise is due to Allah (SWT), for this achievement, and His continuous blessings and favours upon me. My deepest appreciation goes to my principal supervisor, Dr Eyad Elyan, for his patience and guidance throughout my PhD. I also want to thank Prof. Chrisina Jayne for her support in making this thesis a success. I wish to express my greatest gratitude to my colleague Pamela Johnson, for her support during the writing-up stage. I will also like to appreciate Dr. Sadiq Sani, Dr. Laud Charles and all my colleagues in N426 and N436 for the invaluable discussions we had during this research. I would also like to thank my family members for their unending support over the years. I want to say a big thank you to my brother Dr Abubakar Ali-Gombe, for his fatherly advice and backing, and my uncles, Dr Isa Jalo and Musa Jalo for believing in me. My sincere appreciation goes to my wife Hauwa and my son Imam, for their love, support, patience and understanding throughout the journey. Finally, I would like to acknowledge the National Information Technology Development Agency (NITDA) for funding my PhD.

Declaration

I confirm that the work contained in this PhD project report has been composed solely by myself and has not been accepted in any previous application for a degree. All sources of information have been specifically acknowledged and all verbatim extracts are distinguished by quotation marks.

Signed ..... Date .....

Adamu Ali-Gombe

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Overview

Image classification is the process of identifying the object in a given image and is an essential area in computer vision. It has witnessed significant progress over the last decade. Image classification is used in different computer vision domains such as: self-driving cars to isolate road lanes from pavements [202]; in face recognition systems to identify individual faces [141]; and in the medical domain to predict the presence or absence of disease from images such as CT-scans and X-rays images [160].

Traditional image processing methods combine hand-crafted features and supervised machine learning algorithm to classify images. Input images are converted into representations (or features) from pixel values and a classifier such as Support Vector Machine (SVM) is then trained. Examples of these representations include Scale Invariant Feature Transform (SIFT) [119], Speed Up Robust Features (SURF) [16] and Haar features [179]. These approaches are known to be quite efficient in dealing with some image classification challenges. For instance, SIFT keys have demonstrated to be resistance to image scale, orientation and substantial occlusion. The shortfall of these approaches is the reliance on domain knowledge to code representations. Consequently, model performances depend on how efficient these representations are. Moreover, the freedom and generalisation of these approaches is limited.

Deep learning is a branch of machine learning which relies on artificial neural networks to make predictions from data. It is an emerging field that has improved the accuracy of image classification by a wide margin. The accuracy of these models is unmatched by traditional approaches. Deep learning uses raw pixel values with no feature engineering required. These models learn hierarchical patterns during training from simple features

like edges and colour to more complex features like an object's part that makes up the image. Researchers have embraced deep learning models because they generalize better to samples not seen in training. However, massive data is required to achieve the best performances.

1.2 Background

Different domains have benefited from advancements in deep learning such as speech recognition, text generation and video synthesis. In specific domains such as image classification, large volumes of labeled data are required to ensure good results. To address this challenge, large datasets have been made available in the public domain such as ImageNet[1], MSCOCO[2], PASCALVOC[3], and others. These datasets provide a great resource for developing complex and accurate models. Such datasets have also opened doors for other approaches such as transfer learning, where a model trained on a huge and diverse dataset is used to address other problems.

That said, obtaining sufficient labeled examples for a dedicated dataset is crucial in image classification. However, curating a massive dataset has proved to be a daunting task. Abundant unlabeled images are readily available, but the process of annotating and labelling a dataset is labour intensive, time-consuming and costly. One way around this is to reach out to many people through out-sourcing mediums such as Amazon Mechanical Turk[4] to speed up things at some extra cost. Sometimes, the process requires an expert's knowledge to correctly identify the object of interest, such as in the medical domain. Moreover, experts are not always available, and this may add to the overall cost of preparing the data. In some domains like video tempering, collecting tampered videos is challenging [87]; hence, sufficient labeled data can be difficult to acquire. Generally, the problem of limited labeled data occur when no labels are available at all or when the dataset is partially labeled or when the dataset is fully labeled, but few examples exist in the class of interest.

These scenarios led to the proliferation of methods that have been successfully applied where labeled data is scarce. These methods include, unsupervised [63], semi-supervised [92], one-shot [178] or few-shot learning [189]. Data augmentation [50] and synthesis [51] are common practices that have been applied to improve model performances. However, models trained only on synthesized data largely under-perform as

[1] http://image-net.org/
[2] http://cocodataset.org/home
[3] http://host.robots.ox.ac.uk/pascal/VOC/
[4] https://www.mturk.com/

shown by Bousmalis *et al.* [25] and Shrivastava *et al.* [162]. Therefore, it can be argued that acquiring plausible images and well-labeled training data will continue to be crucial in image classification.

In recent years, generative models have demonstrated to be powerful frameworks to obtain plausible training data. This can be very important in cases where labeled samples are hard to obtain. Recently, Generative Adversarial Networks (GAN) have been widely recognized as the state-of-the-art in image generation. For example, GAN models can produce high-resolution images and also diverse categories as shown by Karras *et al.* [89] and Odena *et al.* [140]. This class of generative models are trained using gradient-based back-propagation and allow single feed-forward sampling. The key advantage of GANs is that they do not apply mean squared error (or similar estimates) as correctness measure. They rather apply distance-based similarity of distributions using Kullback-Leibler divergence [63]. This feature improves reliability while producing the finest images. Since their creation in 2014 [63], different GAN frameworks have been proposed and successfully applied to image generation [140], classification [138] and representation learning [52].

1.3 Motivation

Applying deep models requires collecting a large number of images. As seen in Figure 1.1, these images are manually annotated with bounding boxes so that labels are provided to isolate the objects of interest. Bounding box co-ordinates of objects from the annotated images are retrieved when training a model to recognize the object of interest. In some instances, images may require cleaning and processing such as cropping, de-blurring and resizing before labelling. However, annotating images is largely a manual and time consuming task for deep learning practitioners. Furthermore, data collection in some domains such as astrophysical observation [156] or images of the eye movements [162] is very challenging, hence no enough data exist to apply deep models.

Large labeled data are central to obtaining the desired performances from deep models [99]. Obtaining labels for a dataset manually is an expensive task. And automatically annotating and labelling a new dataset still remains an open problem in computer vision. Although, semi-automatic approaches exist but may require a full or partial supervision [196].

Techniques such as semi-supervised learning [92] have demonstrated to be useful in cases where a fully labeled dataset cannot be obtained. The key strength of these approaches is their ability to use publicly available unlabeled data with small labeled data to carry

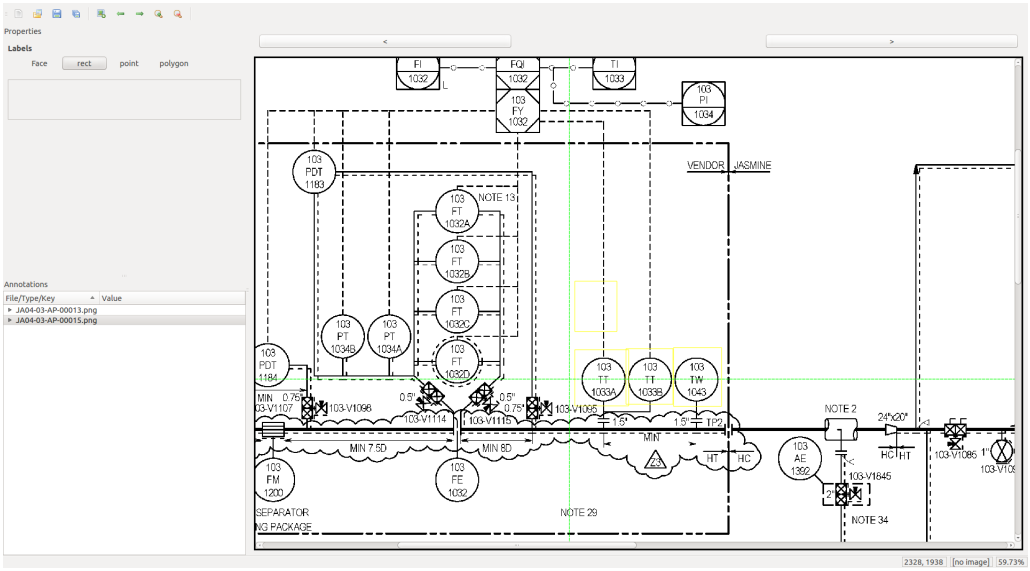


Figure 1.1: *Image annotation and labelling by manually drawing bounding boxes around the object of interest (using sloth annotation tool[a]).*

[a]https://sloth.readthedocs.io/en/latest/

out a task. Nonetheless, these approaches are lagging in terms of accuracy compared to fully supervised approaches. Again, these are ineffective in areas where substantial data cannot be obtained. This has lead to the proliferation of techniques such as transfer learning. Transfer learning applies a model trained on a task with massive data on a second problem with fewer examples. This was successfully applied to many problems and works best when the two tasks are closely related. However, transfer learning creates other issues such as catastrophic forgetting in neural networks [94], and the number of fine-tuning steps or the layers to fix is open to experimentation.

Sometimes labeled data may be available, but the number of samples in the class of interest is small. This is a common occurrence in the real-world. For instance, in fraud detection, majority of the transactions recorded are legitimate, but only an insignificant percentage of the transactions are illegal. This scenario creates a bias in the data categories leading to an imbalanced dataset. A model trained on this type of skewed data performs poorly on the class of interest [183]. Class-imbalance can occur in binary classification, multi-class and multi-label label classification problems. Resampling is a common approach used to rebalance classes; however, simple resampling in a multi-class or multi-label classification will affect other classes [97].

Data augmentation strategies are used in deep learning to increase variance in samples. This technique applies a set of predefined transformation functions such as image translation, noise distortion and affine transforms to existing samples. Since models

Figure 1.2: *Traditional augmentation in road traffic signs. The first row are the initial images and second row are the augmented samples using simple mirror.*

are generalizable, it is assumed that a small distortion in the input should not produce extraneous results. In spite of that, augmentation uses a finite set of functions that may be applied before a reasonable limit is reached [50]. The pitfall of using traditional augmentation in a tightly similar dataset could be disastrous in critical applications such as a self-driving car. For instance, if we apply a simple translation to the German Road Traffic Signs (GRTSB), left turn becomes right turn or speed limit changes from 60 to 90 and so on as shown in Figure 1.2. Again, data augmentation in extreme imbalance cases may not be sufficient. According to Vinyal *et al.* [178] data augmentation by itself does not remedy the lack of data but only prevents over-fitting.

When sufficient data is not available, deep learning practitioners may generate artificial samples through data synthesis. These can be very helpful in instances where data could not be reasonably collected. Conversely, models trained on synthesized data under-perform when tested on real examples [162]. Therefore, a massive well-labeled dataset is necessary in obtaining the desired output from deep models.

1.4 Objectives

The main aim of this research is to develop a method that supports the training of deep models with limited labeled data. The objectives are:

- A literature review of existing methods that addresses the lack of labeled data in deep learning.
- Evaluate the performances of deep learning models on datasets that are poorly annotated and labeled.

- To create a novel method to train deep models using minimal number of labeled examples. This will mitigate the problem of insufficient labeled data in image classification.
- To develop a framework that generates realistic class-specific samples in the absence of enough labeled instances.
- To address class-imbalance in real-world problems with under-represented class instances using the methods and techniques developed.

1.5 Thesis Contributions

The contributions in this thesis are in the domain of image generation and classification. These are as follows.

- An experimental framework to analyze the performance of deep convolutional networks. The experiments were designed to evaluate the performances of the models on poorly annotated samples. We also analyzed existing approaches to improve model performances such as transfer learning and traditional data augmentation approaches. The results highlighted limitations of these approaches and some of the classification challenges when sufficient labeled data is not available. The findings were presented at the 18^{th} International Conference on Engineering Applications of Neural Networks 2017 (EANN) in Athens, Greece [6].
- A novel generative adversarial network was developed. The new model extended the existing framework with multiple fake classes and a network switcher. We also developed a new method to train a classifier using a large amount of unlabeled data along with adversarial training. Multiple fake classes improve classification performances in a fine-grain classification, and the network switcher enables training on unlabeled data while reducing model parts redundancy. The results appeared as part of the proceeding in IEEE World Congress on Computation Intelligence (WCCI), International Joint Conference on Neural Networks in July 2018 at Rio De Janeiro, Brazil [3].
- Synthesized high-quality class-specific samples using the proposed model from different class-imbalance datasets. This was used to address the limitations of traditional augmentation approaches in extreme and multi-class class imbalance scenarios. The generated minority samples were used as a source of data for augmentation. Our generated samples showed better diversity and quality when compared to popular resampling and state-of-the-art GAN models. This was

published in July 2019 edition of Elsevier NeuroComputing Journal [5].

- Applied the new frameworks in challenging real-world domains, namely, facial images and engineering drawings. The model was used to generate face images from extreme under-represented attribute and symbols from multiple minority classes. We demonstrated the usefulness of these samples by improving classification performances in the minority classes. The results from this work is part of the proceeding at the International Joint Conference on Neural Networks in July 2019 at Budapest [2].

A list of the publications resulting from these contributions are available in appendix A.

1.6 Thesis Structure

The remainder of this thesis is organised as follows.

Chapter 2 Literature Review. This chapter reviews existing approaches to learning when labeled data is limited. In this chapter, we highlight some of the strengths and weaknesses of these existing techniques.

Chapter 3 Generative Adversarial Networks. This chapter introduces Generative Adversarial Networks and some basic theoretical background around adversarial training which forms the basis of the methods proposed. The chapter explores what GANs can generate, where they can be applied and identify some of the limitations of the framework. The chapter also focuses on the recent advances in this area and categorizes them accordingly.

Chapter 4 Deep Learning and labeling Dilemma. In this chapter, we evaluate the performance of deep learning models on a poorly annotated dataset. We also analyze the performance of traditional augmentation techniques and transfer learning in small labeled dataset. The experiments carried out, demonstrated that without sufficient data the model performs poorly.

Chapter 5 Few Shot Classifier GAN. This chapter introduces a novel GAN model to learn from unlabeled data. This model is used to generate images and train a classifier. The new model uses multiple fake classes whilst combining both unsupervised and supervised learning in training using network switchers. The trained classifier achieved good classification performance even in the presence of substantial unlabeled data.

Chapter 6 Image Generation and Classification from Class-imbalanced Datasets. This chapter presents a new GAN framework that is capable of generating data from a

class-imbalanced dataset. In this chapter, the proposed model was also used as an augmentation approach to neutralize class imbalanced datasets. We compared the proposed framework with existing resampling and state-of-the-art GAN framework. The samples from the proposed model had better diversity which is indicated by the performances over different metrics.

Chapter 7 Application Domains. This chapter applies the proposed model on two real-world application domain where labeled training data is scarce. The model is applied to face generation from under-represented attributes and symbols generation from P&ID diagram. In both cases, the model demonstrated significant improvements over the baseline, particularly in extreme conditions.

Chapter 8 Conclusion. The conclusions of the thesis are presented in this chapter. It summarises the overall findings from the overall body of work. These findings are discussed and we also provide an insight into some possible future research areas.

Chapter 2

Literature Review

This chapter reviews traditional image classification and deep learning approaches. We explore the recent successes of deep learning and the reliance on huge labeled data. In particular, the chapter surveys established approaches in addressing learning when sufficient labeled data is not available. The chapter also identifies some of the successes and limitations of these techniques.

2.1 Traditional Approaches

Traditional machine learning approaches have been used to detect and classify objects, and over the years, they have become popular in computer vision. These approaches require features engineering which are hard coded. The traditional approach to image classification follows the pipeline of preprocessing, feature extraction and classification.

2.1.1 Preprocessing

Preprocessing is a cleaning step that reduces the number of unwanted elements within an image. This may also be applied to further isolate regions of interest from an image. Real data such as pictures and videos can be very noisy, complex and redundant. Part of preparing a dataset for other tasks may be labelling and annotating them. Preprocessing may also include some random cropping [99], resizing [164], image rotation [6], whitening [164], thresholding [186], morphological operations [42], binarization [144] and other techniques favouring the classification algorithm. These are valuable techniques that prepare the image by isolating further the region of interest for easy feature extraction. These techniques are independent of the classification algorithm chosen.

Figure 2.1: *SIFT key proposals drawn across the image by applying the SIFT algorithm. The center of the circles represents the key points and the circle is the neighbourhood considered.*

2.1.2 Feature Extraction

Features in an image can be local when they are specific to a region or global when they spans the entire image. Image features may include texture, contours, edges, lines, intensity values, colours or even objects. A feature within an image is a region of interest which can be a segment of an image with a common characteristics. Features may be identified by sudden change in intensity values or other common characteristics of an image region. Features may also be defined by complex relationships such as neighbourhoods or transformation functions to bring out the desired results. The process of identifying these features in an image is known as features extraction. Using simple features from images, researchers have developed representations that are invariant to challenges faced in image classification. For instance, Lowe [119] developed Scale Invariant Features Transform (SIFT) to address the issue of scale, illumination and rotation (upto 60°) in image classification. SIFT keys are generated from image pyramids obtained by systematically applying Gaussian function to estimate maxima and minima of a pixel by comparing it to its neighbours. SIFT key representations are also resistant to image deformation and occlusion to a certain degree. Figure 2.1 shows SIFT keys identified as circles in the figure.

Scale invariance in an important characteristic of feature representations. However, the size of key representations, the speed of generation and classification is very important

in image applications. In order to realize true scale invariance in image detection, Viola and Jones [179] introduced a new method to generate image features. This technique generates Haar-like features from pixels. A key contribution of this approach is the generation of *Integral Image* based on pixel neighbourhood regions. An integral image is an image whose pixel values are the summation of the values before it (that is left and above it). Features are collected from 2, 3 and 4 rectangular regions by the applications of Haar basis function. Haar functions are an orthogonal family of switch rectangular waveforms whose value can differ from one function to another [4]. An interesting attribute of this process is that all this is done in constant time. This simplified object detection which made tasks such as facial detection applicable and deployable to facial databases, teleconferencing and portable devices.

Speed Up Robust features (SURF) [16] were introduced as a way to perform interest point detection and description faster, with more efficiency than previously known methods. This technique makes use of the idea of *Integral Images* to generate a robust features vector. It is invariant to both scale and in-plane rotation. Integral images allow for fast box-type convolution filters (i.e different filters are applied to image segments at ones). The speed of detection is increased by the application of Hessian matrix which is used to detect blobs. The Hessian is a second derivative and second order cross partial of a scalar function taken at a point [113]. Feature detection is made visible with gradient change. These are stored in a blob response map on a different scale. Contrast types are determined by the sign of the Laplacian that allows for faster matching as well. SURF keys perform well in classification task even in difficult real-life applications like 3D-reconstruction, camera calibration and mosaicing human retina.

2.1.3 Image Classification

Classification associate data with an existing class and could be between two (binary) or more (multiple) classes. In some cases, more than one class may be considered as appropriate (multi-label classification). Classes are encoded using labels usually in one-hot encoding that represent the target value at the end of the task. Classification plays an important role in areas such as image recognition [141], scene parsing [31], autonomous driving [202] and automatic segmentation [118]. Images are represented as a matrix of pixel values in the computer memory. This representation is subject to other standards available such as RGB, YUV, HSI and others. Image classification is a supervised task that groups similar images into a predefined category. While humans consider image classification a very simple task, image classification is a challenging task to computers. Some of these challenges include scale, occlusion, illumination, noise and lack of labeled data. In a general sense, classifying an image may require

some pre-processing, feature extraction and a classification algorithm.

Machine learning is used in classification task in various domains such as Natural Language Processing (NLP), Case Based Reasoning (CBR) and computer vision. Early successes in image classification were recorded with machine learning by relying on feature representations to understand the pattern in underlying data. The features extracted are used to train a model such as logistic regression, Nearest Neighbour (K-NN), Support Vector Machines (SVM) and Decision Tree (DT). For instance, Setitra and Larabi [159] performed object classification and retrieval from images by training a K-NN classifier using SIFT features in a bag of features representation. Closely related to this, Hossain *et al.* [78] classified fish species using SVM with Pyramid Histogram of Words (PHW) which is a variant of SIFT in a Bag Of Words (BOW) representation. Local Logistic Regression (LLR) is a variant of logistic regression that was used by Yang *et al.* [191] to classify images of hand written digits. Although these techniques work, their accuracy is dependent on the hard coded representations chosen.

2.2 Deep Learning

Valpolka in [177] defined deep learning as an art of estimating deep features in a hierarchical manner. For example in an image problem, these features could start from simpler ones such as colour, edges and lines in lower layers to more complex ones such as object parts in upper layers. Deep learning algorithms are representation learning algorithms that apply non-linear transformation sequentially to data and create a more abstract representation from the data [62]. Deep learning has been effective in challenging problems where it is very difficult to formulate a direct solution.

Most deep learning approaches today implement some form of a Neural Networks(NN) model. There are different kinds of neural networks such as a fully connected network or Multi Layer Perceptron (MLP), a partially connected network such as a Convolutional Neural Network (CNN), a Recuurent Neural Network (RNN), a Bayesian network and others. Generally, a neural network contains a collection of neurons which represent small computational units. A neuron maybe seen as a single logistic regression unit. NN usually has an input, hidden and output layers composed of one or more units. Figure 2.2 shows the structure of a neural network. Deep neural networks are neural networks that have several layers of hidden units. NN has been around since the inception of artificial intelligence [62]. Its resurgence is due to the increase in computational power and the availability of data [164]. Neural networks are popularly trained by applying back propagation technique [104] to allow gradient flow across different layers

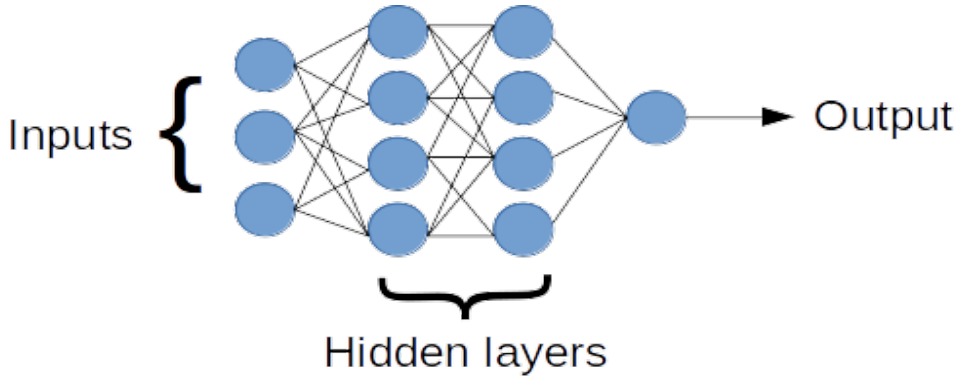


Figure 2.2: *Structure of a typical neural network.*

in the model. Back-propagation ensures that the loss from the output layer is disseminated to previous layers through the model. Modern NN employ algorithms such as Adam Optimizers [91] and Stochastic Gradient Descent [23] in optimizing the model.

Deep learning models work on raw pixel values; thus, no features engineering is required. These models have gained popularity in image classification domain because of their distinctive accuracy, especially in the presence of substantial training data. The number of layers in deep neural networks have increased significantly over the years, from Alex-Net [99] with less than ten layers to Dense-Nets [80] with over 200 layers. Efficient parameter tuning, over-fitting handling, vanishing gradient solutions and better training techniques have also proliferated. Deeper models have higher accuracies than shallow ones; however, data requirement increases as the network size increases.

More recently, with the advent of large-scale labeled datasets such as Imagenet [44] and MSCOCO [112], deep learning algorithms have provided state-of-the-art solutions in challenging computer vision problems like object detection and classification. For instance, one of the breakthroughs in this area was after the invention of Imagenet dataset and competition. Although the labelling process was manual, the dataset provided very large, cleaned and fully labeled data with 1000 classes. When Krizhevsky *et al.* in 2012 applied a deep Convolution Neural Network (CNN) on Imagenet recognition challenge, their error rate was half that of the runner-up in the same competition [99]. Since then, the winning models has been deep learning algorithms with error rates continually approaching zero.

Semi-automatic [45] and weakly supervised [30] labelling approaches exist. However, crowd sourcing remains the most popular and effective choice for labelling a dataset.

Annotators use manual annotation tools such as Sloth[1], LEAR[2], VGG Image Annotator[3] and FastAnnoation[4] as standalone tools for image labelling. Annotation tools such as LabelMe [153] are available online. For online tools, there may be privacy issues with data sources and that might not be an option. Some of the challenges confronting these tools are: what to label, complexity, validation and the label text itself [153]. The major downside of these tools is that users manually annotate the images. Hence, accuracy and quality of annotations is a bottleneck problem. In order to reduce this effect, researchers have resorted to automatic and manual selective validations [21, 45]. Others leave this open to the annotator to decide what to label and how to segment it if required [153].

2.3 Lack of Sufficient labeled Examples

Access to large datasets has been one of the key factors that has pushed further the boundaries of image recognition and deep learning research as a whole [164]. However, only a small set of data in the public domain is labeled. Consequently, several approaches have been proposed to mitigate the lack of labeled data in deep learning such as transfer learning, unsupervised learning, few-shot learning and generative modelling.

2.3.1 Unsupervised Learning

Unsupervised learning has been used to support classification through representation learning over unlabeled data which are readily available. A common way to learn good features is to use an auto-encoder which have been around for a while. An auto-encoder as an input, a hidden (bottleneck layer) and an output layer. This set-up is used to compress representations in the input and regenerate them in output layer. In this way, by reconstructing the inputs correctly, the network will learn the most important attribute of the input data and describe the latent attributes of the input data. The auto-encoders proposed by Hinton [77] were used as a dimensionality reduction tool in images using non-linear transformations. In this approach, training starts with a layer-wise pre-training of Restricted Boltzmann Machine (RBM) layers followed by a fine-tuning step using image reconstruction loss. However, scaling up is a significant limitation for this model [77]. In a similar approach, Lee *et al.* [107] stack up RBMs using convolutional layers similar to a deep belief set up called Convolutional

[1]http://sloth.readthedocs.io/en/latest/
[2]https://lear.inrialpes.fr/people/klaeser/software-image-annotation
[3]http://www.robots.ox.ac.uk/ vgg/software/via/
[4]https://github.com/christopher5106/FastAnnotationTool

Deep Belief Networks or CDBN. Images used for training were large, which is aimed at a more realistic scenario. The authors also introduced a novel pooling based on probabilistic max-pooling. CDBN were evaluated using CALTECH-101 and MNIST datasets and results obtained are comparable to the state-of-the-art. CDBN also proved to be effective when tested on unlabeled data to extract representation from images that the network has not seen.

Raina *et al.* [147] demonstrated how unlabeled data from different categories than the target classes could be used to improve classification accuracies. They formalised the problem as a self-taught learning problem where the model learns basic elements from unlabeled data by applying sparse coding. In their experiments, images were pre-processed using Principal Component Analysis (PCA) to reduce dimension before learning algorithm is applied. An SVM and Gaussian Discriminant Analysis (GDA) were used for supervised classification by comparing raw pixels, PCA and sparse coding features. Sparse coding achieved the best performances on Caltech101 images with an accuracy of 46.6% using this approach, which was significantly better than the first published supervised approach on the dataset (16%).

Deep models such as Convolutional Neural Networks (CNN) also learn representations irrespective of the task they perform. These representations are independent of classes. Thus, deep learning researchers have applied them in performing classification without the need for labeled data during training. This can be seen in the work of Dosovitskiy *et al.* [50] were they trained a CNN using few data and no labels in an unsupervised fashion. Training samples were generated by randomly selecting a fixed number of images from a dataset, in this situation STL-10, Caltech-10 and CIFAR-10. These samples are passed through a set of random transformation functions that generate a set of surrogates. The surrogates are used to train a CNN nick-named Exemplar-CNN. Exemplar-CNN had three convolution layers and one fully connected layer. An SVM was used as a classifier that feeds on the features learned by the CNN. Results obtained were better than the state-of-the-art models on STL-10 dataset in unsupervised learning. However, training accuracy plateaued when surrogates reached 8000 that is, increasing the number of surrogates beyond 8000 did not improve the model accuracies.

In a similar direction to [50], Gidaris *et al.* [60] used geometric transformations to learn semantic representations from unlabeled data. The authors trained the CNN to recognise 2D rotation angle of images through classification. Learning these simple transformations proved significant in learning powerful semantic structures better than state-of-the-art on different benchmark datasets and significantly closed the gap on supervised learning. Training on the pretext task provided alternative features that have

improved different vision tasks such as detection, classification and segmentation. Experiments were conducted on CIFAR-10, Imagenet, Places and PASCAL VOC datasets on different computer vision tasks.

2.3.2 Semi-supervised Learning

Semi-supervised learning addresses part of the label availability by integrating unlabeled data in the training process. Because both the labeled and unlabeled data come from the same domain, the unlabeled data can be used to improve supervised tasks like classification by learning generic tasks. These generic tasks include learning representations/features [3], predicting image orientation [60] and pseudo labelling of samples [106]. Semi-supervised learning is particularly justifiable because of the vast amount of unlabeled data in the public domain; hence, it has attracted a lot of interest recently. It has been successfully applied to discriminative as well as generative models where the number of labeled examples are small [69].

Lee [106] carried out semi-supervised learning by performing supervised learning with pseudo-labels for unlabeled data. Pseudo-labels were obtained from the classifier by chosen the class with the highest activation during classification and these were treated as if they were true labels of the samples. Because of the varying number of labeled to unlabeled examples, the author used a supervised loss function with a hyper-parameter on unlabeled sample losses. Hence, the neural network was trained using pseudo-labels as entropy regularization. Experiments were run using a neural network with one hidden layer and 100, 600 and 3000 labeled examples from MNIST dataset. The proposed methods achieved state-of-the-art performance on MNIST dataset. Apart from classification, another task that has benefited from semi-supervised learning is segmentation. For instance, semi-supervised segmentation was achieved by [167] using weakly annotated training data, unlabeled data and a set of generated images as extra training data. Similarly, self-supervision techniques were also used to improve image generation in [33]. Kuzniestsov *et al.* [101] also used semi-supervised learning to predict depth maps from monocular images using auto-encoders.

2.3.3 Transfer Learning

Transfer learning is a technique that takes a model trained on a large dataset (source dataset) and apply it to a different problem domain (target dataset). The idea behind this is that since the same hierarchical representations are learned by different converged networks during training, representations and weights learned should be generalizable

across different datasets. By intuition, a significant amount of training time can be saved by providing this knowledge from an already converged network. The common practice is to replace the top layer(s) of the model, usually the classification layer or in some cases the fully connected layers as well to accommodate the new problem set [6].

Transfer learning may require further training, also known as fine-tuning, to suit the target. Learned parameters from state-of-the-art models such as Alex-Net [99], VGG-Nets [164], Inception model [174] and Res-Net [72] are all available in the public domain. These have been successfully applied as an alternative to training a model from scratch, especially when the two domains are similar. Starting from a pre-trained network promotes faster convergence and better accuracies than training from scratch. For instance, Long *et al.* [118] trained a fully connected network for semantic segmentation by using already trained Alex-Net, Google-Net and VGG-16 weights. Pre-trained weights were also applied in [132] for classification with deep models and to perform domain adaptation. However, transfer learning is restricted by model architecture and may fail if the two domains are very different [6]. Also, the amount of labeled data required to fine-tune the model may be large. Moreover, transfer learning does not address other classification challenges like a class-imbalance problem.

2.3.4 Class-Imbalance and Cost-Sensitive Methods

The class-imbalance problem in binary classification is an active research area which has witnessed the development of well-established techniques. However, little attention is given to the class-imbalance problem in multi-classification [97]. Imbalanced classes in a multi-classification problem may require new sampling strategies and data pre-processing steps [97] other than those used in binary classification. Existing methods for handling such problem include multi-class decomposition [56], Class Rectification Loss (CRL) [49] and mean squared false error [183]. Resampling methods such as oversampling and undersampling are widely used in this area. However, oversampling is prone to over-fitting and undersampling may discard essential data points [27].

Buda *et al.* [27] showed in an experimental study how the performance of CNN drops significantly when the data is imbalanced. Wang *et al.* [183] modified the learning algorithm to account for class-imbalance by penalising the misclassification of minority class instances (i.e., cost-sensitive methods). However, applying such methods require careful consideration of the cost matrix settings, which can be tricky in a real-life problem [97].

In deep models such as CNN for example, Class Rectification Loss (CRL) [49] was

used to handle class-imbalance. CRL algorithm performs hard mining of the minority class in each batch forcing the model to create a boundary for each minority class with a hard positive and negative threshold. Similarly, Large Margin Local Embedding (LMLE) [79] employs clustering among classes to maintain the structure of the minority data. However, these techniques can be computationally expensive in large data domains [49].

2.3.5 Data Augmentation

Data augmentation techniques are increasingly becoming an integral part of deep model approaches for classification. Exemplar CNN [50] was based on systematic augmentation of data and achieved state-of-the-art results on CIFAR-10 dataset. Data augmentation is a widely used technique to handle class-imbalanced datasets. Ali *et al.* [6] used affine transformation and noise distortion across classes to generate more samples and reduce the impact of class-imbalance. However, trivial augmentation may not suffice for extreme class-imbalanced data or when sufficient data is not available. Besides, orientation-related features in some domain may limit the application of simple augmentation approaches [122]. Thus, more sophisticated augmentation techniques such as image pairing [84] and mixup [199] have been proposed.

In recent years, generative models were successfully used to generate samples. GANs proved to be state-of-the-art in generating and capturing data [89]. In an imbalanced dataset, the aim is to generate class-specific samples, therefore supervised GAN models such as Conditional GAN (C-GAN) [126] offer potential solution for such a problem. However, these models and other established GAN frameworks such as vanilla GAN [63] and Auxiliary Classifier GAN (AC-GAN) [140] have performed poorly on class-imbalanced datasets by failing to generate the required minority samples [122, 181]. Recently, good performance was reported by [57] using a Deep Convolutional GAN (DCGAN) [146] to synthesize artificial liver lesion images. This was achieved by using traditional augmentation techniques to oversample the training set. Similarly, Baur *et al.* [15] generated high-resolution skin lesion images using MelanoGAN (a variant of DCGAN + Laplacian GAN [47]) from a small dataset of 2k samples. The model was used to synthesize more skin lesion samples to reduce the effect of class-imbalanced data in training a ResNet-50 [73] for classification. These examples show that trivial data augmentation techniques can be successful in handling class imbalance related problems ([57], [15]). However, it should be noted that these examples were applied to binary datasets with no orientation dependent features or fuzzy class boundaries.

Other approaches combine GAN with other generative processes such as auto-encoder training. Features learned by the auto-encoder are then used to initialize the generator and discriminator of the GAN model. This may require a second training step [122] or joint training [181] to perform conditional adversarial training. Data Augmentation Generative Adversarial Networks (DAGAN) [9], Balancing GAN (BAGAN) [122] and Fine-grained Multi-attribute GAN (FM-GAN) [181] used a similar strategy to synthesize more samples. Image refinement is another technique used which preserves the image class while producing diverse synthetic samples. Zhu *et al.* [204] applied image translation to generate minority samples using a reference sample in an emotion recognition task. However, this approach was evaluated using two closely-related classes (i.e translate a face to another face image). Other approaches re-parametrise the adversarial training by adding extra losses or stricter conditions during generation. This enforces learning and generation of minority samples such as in DeliGAN [67]. The latent space in DeliGAN is parametrized by a Gaussian Mixture Model (GMM) whose parameters are learned alongside the GAN parameters.

2.3.6 Few-shot Learning

One-shot learning was introduced by Fei-Fei *et al.* [55] as a way to train models with one or few labeled samples available. The model is simply formulated as a Bayesian framework, $P(\theta|\mathcal{X}, \mathcal{A})$, where θ is the model parameters, $\mathcal{X}$ are the features locations and $\mathcal{A}$ is the appearance. This is advantageous in learning classes with one instance from previously learned classes with many instances. It is useful in applications such as face recognition [175] where only one image of a person is available. Such applications are required to cater for newly enrolled users after the system is deployed. Deepface face recognition system [175] is an example of such an idea. Deepface employs a CNN that was trained using labeled Faces in Wild (LFW) [81] to learn face representations. At test time, Chi-squared distance (χ^2) was used in the classifier to compare two facial representations. The authors achieved a 97.3% accuracy on the test set which was better than all existing approaches. In the same direction, a triplet loss function was used by Schroff *et al.*[157] to train their face recognition CNN and achieved a 99.63% on LFW and 95.12% on YouTube Faces Database [187].

Few-shot learning is an extension of one-shot learning to situations with few labeled more training examples. The argument is that a classifier is expected to generalize well to new classes not seen in training given small number of examples. For instance, Xu *et al.* [189] applied a novel few-shot model to retrieval/recognition task and reported better accuracies than state-of-the-art-models on the benchmark datasets. Few-shot

classification was also used by Mottias *et al.* [132] to perform supervised domain adaptation using a deep model. The target dataset was chosen to be weekly labeled and the approach combined adversarial learning with few-shot classification to obtain state-of-the-art results on the benchmark datasets. Common deep models used in few-shot learning include Siamese networks [175] Memory Networks [172, 178], and Prototypical Networks [165].

Attention mechanism in matching networks proposed by Vinyals *et al.* [178] was applied to learn label embeddings from few examples in order to classify unlabeled samples. The authors used a non-parametric approach to few-shot classification. The concept of neural network with memory allowed the framework to train by matching images of unlabeled samples to labeled data within the label space. This may be explained as a weighted nearest neighbour classification. The approach showed improved performances when applied to computer vision problem in Omniglot [102] and *mini*Imagenet [178] datasets. Using a different approach, Ravi and Larochelle proposed learning a meta-model in a few-shot regime. The model uses a Long Short Term Memory (LSTM) module to control the updates to a classifier and the model trains specific classifier versions for every episode. The results obtained was comparable to other metric learners such as matching networks.

Prototypical networks [165] use a linear classifier that is based on Euclidean distance to perform few-shot classification. Prototypical networks are similar to matching networks; however, a prototype was used as a support. The prototype is the mean of a class in embedding space. The intuition behind choosing the mean as support because similar samples tend to cluster around the class mean. The model also employed episodic training proposed in [148]. The authors experimented on CUB-200 [180] and achieved state-of-the-art performances on CUB-200 dataset in zero-shot learning.

2.3.7 Classification using Generative Models

Kingma *et al.* [92] demonstrated how deep generative models can be useful in learning a good classifier in the absence of a fully labeled dataset. Generative Adversarial Networks (GAN) is one approach which has been receiving attention lately. It is an efficient way to combine other learning techniques such as semi-supervised to address the availability of labeled data. This was demonstrated in [143] where the authors predict wealth distribution from a huge dataset of satellite images using Wasserstein GAN (WGAN) with only 5% of the data being labeled.

GAN was also used to improve classification on synthetic samples in [162]. The authors showed that models trained on refined images perform better than those trained

on synthetic images. GAN models are useful tools in handling scarce labeled data. For instance, conditional GAN was used in [51] to generate a completely new set of data to solve the problem of imbalanced training data. Adversially Learned Inference GAN (ALI-GAN) [52] was tested on SVHN and CIFAR-10 dataset for semi-supervised learning and it outperformed all GAN models in image classification. However, the results obtained were still below the supervised and the best semi-supervised approach.

An extension to ALI-GAN [52] was proposed by Donahue *et al.* [48]. The model was used to learn the inverse mapping from image space into latent space. The proposed GAN model was trained on pairs of latent representations and images. An encoder network was added alongside the generator and was used to project real images into latent space. The encoder and generator did not share weights but are made to work together in fooling the discriminator network. Experiments were run on MNIST, Imagenet and PASCAL VOC. Features learned were tested on classification, image generation and segmentation tasks.

The latent spaces in GAN generator have also proved to be useful in classification. Kumar *et al.* [100] modified the objective of Bi-directional GAN (BIGAN) [48] to include a term that approximates the tangent space of data manifold. The tangent spaces is a directional derivative operator along the transformation function (linear map) in a manifold. In this context, is simply a set of vectors that define salient features in the data manifold. The Tangents learned were directly injected into the classifier in order to improve classification invariance. The proposed approach produced better accuracies on SVHN than the benchmark but performed poorly on CIFAR-10. However, their approach was trained on far fewer epochs than the original BIGAN.

2.4 Conclusions

Deep learning models are state-of-art in image classification [80], object detection [88], semantic segmentation [88] and other related problems on several benchmark datasets. However, these approaches require massive labeled datasets. The first perspective of the problem could be to obtain automatic labelling tools and techniques. However, this remains an open problem in the research community. OpenImages [96] is one of few datasets that make use of automatic labeling at large scale (9 million images). Apart from the fact that its validation and test sets were annotated by humans using crowdsourcing, a large number of its labels comes from Inception model which was trained on Imagenet [20]. Another attribute of the labels in this setup is the confidence score, which is a source of concern for users. With humans, the score is binary (either 0

or 1), but it can assume a range of floating-point values from 0.5 to 1.0 with machine-generated labels.

The second perspective to the problem is to develop a technique that can handle datasets with small labeled data. Semi-supervised learning is one such technique that complements a small labeled data with large unlabeled data to improve performance. Significant progress has been made in semi-supervised learning domain but substantial labeled samples are required to achieve optimum results. Moreover, the unlabeled data employed is usually from the same classes as the small labeled examples and this could be challenging for domains where data collection is difficult. Few-shot learning is a promising approach that has been successfully applied to small and less challenging datasets such as Omniglot. But few-shot learning still falls short when compared to fully supervised approaches on complex datasets. For instance, the few-shot learning error on *mini*Imagenet (subset of Imagenet) [178] is far greater than the best-supervised approach [80] on the *full*Imagenet dataset. This suggests that as the complexity grows, the model is unable to scale up as required. Furthermore, few-shot learning is label agnostic in training but still requires label information to prepare batches and uses a fully supervised episodic training.

Generative models use distance metrics to estimate the probability distributions over data. Particularly, the use of Kullback-Leibler divergence as a metric has proved to be a good estimate in producing better generative models. Despite the great successes of generative models, their contribution in classification is minimal when compared to their discriminative counter parts. Existing research has shown that using generative models such as GANs, to solve limited labels problem can be achieved in two ways. It can be used to train with partly labeled data by complementing with unlabeled data such as [143] in a semi-supervised learning set up, but may require some reasonable amount of labeled data. Alternatively, GANs can be used to generate more samples to augment the training set [51]. That said, the advancements in both directions are minimal and far from competing with other classification approaches [52]. GANs have great potentials but are limited by other factors surrounding their usage such as training complexity, the fidelity of generated samples and the lack of a clear evaluation criterion. However, their flexibility to be combined with different learning approaches makes them strong candidates in this domain.

Supervised GANs require large labeled data and research has also shown that these models are sensitive to class-imbalance. Subsequently, we explore how these limitations could be mitigated and the possibility of using adversarial training and GAN generated samples to improve classification performances. The next chapter delves in details into the recent trends in GAN models and further investigate the limitations identified.

Chapter 3

Generative Adversarial Networks

This chapter discusses in details the structure of Generative Adversarial Networks (GAN). According to Yann Lecun, one of the pioneers of deep learning, *"GANs and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion"*[1]. In the following sections, we discuss in detail the training procedure of GANs, how to optimize these models, and with a particular interest in supervised GANs. This chapter also looks at the recent progress in GANs concerning the architecture, objective function and training techniques. We also present limitations of GANs, some evaluation techniques and examples of successful application areas.

3.1 Introduction

Generative Adversarial Networks (GAN) was proposed by Goodfellow *et al.* [63]. The GAN framework was used to estimate generative models through adversarial training. A generative adversarial network is a deep generative model (also a structured probabilistic model) that has two-component models, a generator G and a discriminator D. Both D and G are implemented as either neural networks (most of the time) or other differentiable models. The Generator randomly samples inputs z, from a noise distribution $p(z)$. The noise input is used to generate image samples that are passed to the discriminator. The discriminator is also trained using existing training data. The task of the discriminator is to distinguish between samples that are generated from noise and the existing data. The discriminator is sometimes referred to as as *critic*. Figure 3.1 shows the structure of the GAN model.

The output in Figure 3.1 *real/fake*, represents the probability of an image being real or

[1] https://www.analyticsindiamag.com/gans-biggest-breakthrough-in-ai/

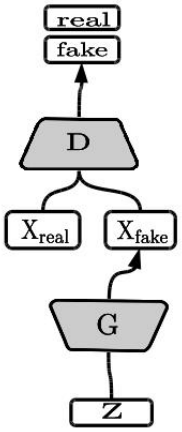


Figure 3.1: *Structure of the original GAN Model.*

fake, X_{real} and X_{fake} represent the samples from the training set (real data) and the generated samples (fake data) respectively. The generator tries to maximize the probability of the discriminator mistaking its samples for real data while the discriminator tries to minimize that. The two models are adversaries in their game objective, and the value function is shown in equation 3.1.

$$\min_D \max_G V(D,G) = E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim p_z(z)}[log(1-D(G(z)))] \text{ [63]} \tag{3.1}$$

Where $p_{data}(x)$ is the probability distribution over the real data, x is a sample from the real training data, p_z is the probability distribution over the noise vector z, and $G(z)$ is the output from the generator function G (or generated images). The value function in Equation 3.1 converges at a saddle point in between the min-max of the two equilibriums under non-extreme conditions. In practice, the training is done interchangeably using gradient descent over the parameters of D and G, as shown in equations 3.2 and 3.3 for some m mini-batches. Again, G is trained to maximize $logD(G(z))$ to avoid saturation. That is to ensure gradient flow under different conditions.

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} [logD(x^{(i)}) + log(1-D(G(z^{(i)}))] \tag{3.2}$$

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} log(1-D(G(z^{(i)}))) \tag{3.3}$$

Where θ_d are the parameters of the discriminator d, θ_g are the parameters of the

generator g, $z^{(i)}, x^{(i)}$ are a single batch of generated and training data respectively. The above equations are gradient updates that could be implemented using Stochastic Gradient Descent algorithm (SGD). The initial experiments showed that the framework requires some k steps update on the discriminator, followed by a single-step update of the generator. This is to keep the discriminator close to optimum and better or ahead of the generator network. This is because the discriminator model is used to guide the learning process. The aim in a GAN set-up is to end up with a G that can generate samples that are within the actual data distribution. However, there is no guaranty that the value function $V(D, G)$ will converge to the saddle point at every run hence, it is mainly unstable in training.

Global Optimality : The framework reaches optimum when the probability distribution over the generated samples (P_{gen}) are closely indistinguishable from the probability distribution over the original data (P_{data}). At this stage, it is expected that the two distributions P_{data} and P_{gen} would be equal. This was demonstrated using Jenson Shannon (JS) divergence [63].

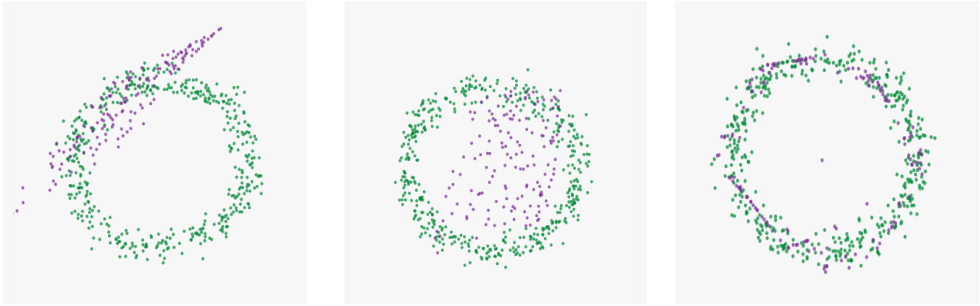
Optimal Discriminator : An optimal discriminator D^* for some fixed G, is given by

$$D^*{}_G(x) = \frac{p_{data}(x)}{p_{data}(x) + p_gen(x)} \quad [63] \tag{3.4}$$

An optimum D does not allow for the correct G to be learned because there is no adequate gradient flow for G updates. Again, there is no guaranty that the model will reach a global optimum or converge during training. Arjovsky *et al.* [11] studied the instability problem and vanishing gradients of GAN networks. The authors suggested that selection and dimensionality of the sample space greatly affect the stability of the generator. Also, in their experiment using the cost functions (equations 3.2 and 3.3), a perfect discriminator contributed to unpredictability of models. Hence, the generator may not learn anything with an optimal discriminator. A vanishing gradient on the generator was caused by the optimal discriminator as shown in [11] when observing the gradient change in the discriminator of a Deep Convolution GAN (DCGAN). Similarly, the gradient in the alternate cost function $-logD(G(z))$ for the generator has high variance, which leads to instability in training. They argued that using a divergence measure such as Jenson Shannon or Kullback Leibler divergence as a metric is not appropriate.

Nonetheless, remarkable results are obtained even with the original GAN objective. Using GANLab[2] simulation, Figure 3.2 shows a simulated example of a GAN learning a simple 2D mixture of Gaussian Noise datasets. The GAN generator used has one

[2]https://poloclub.github.io/ganlab/



(a) 0 iterations (b) 1k iterations (c) 10k Iterations

Figure 3.2: *The green dots are real data points and the purple dots are the generated data points. After 10k iterations, the generator model has almost perfectly matched the data distribution.*

hidden layer with ten neurons and a discriminator with two layers and nine neurons. The model was trained using the original GAN objective, and it can be seen from the figure that even with a simple model, the GAN framework was able to capture the distribution.

Other generative frameworks such as Deep Boltzmann Machine (DBM) [107], Noise Contrastive Estimation (NCE) [68] and Variational Auto-Encoder (VAE) [93] all share some similarities with GAN. All these approaches learn some form of density distribution over a set of data. Generative models have been unsuccessful in some ways because they rely on maximum likelihood estimation of probability density distributions which can sometimes be very costly to compute. Training complexities had also limited their ability to scale up to larger problem sets.

GANs have been successful in areas where other generative models have failed, such as high-resolution and less blurry image synthesis. Goodfellow *et al.* [63] used an end to end differentiable model to estimate loglikelihood without Markov chain approximations. GAN models receive a significant boost when Radford *et al.* [146] discovered how deep convolutional neural network can be used to obtain high-resolution images, and when Mirza and Osindero [126] proposed labels for plausible image generation. These formed the basis for many variants of supervised GAN models. GAN since its inception have been used in image re-construction [156], semi-supervised classification [154], image correction, text to image generation [150], natural image generation [47] and others. Several variations of GANs have also proliferated such as Couple GAN, Deep Convolutional GAN (DCGAN) [146], InfoGAN [34], BIGAN [48], LAPGAN [47] and others to address different domain needs and to expand its frontiers. Despite all these breakthroughs, GANs still suffer from challenges such as lack of a clear evaluation criteria [176] and mode collapse/drop [63]. A complete mode collapse occurs when the

generator produced samples that do not resemble the training data and mode dropping occurs when the generated samples are not diverse (same image or scene in every sample).

3.2 Supervised GAN

The original GAN framework is unsupervised and can be used to generate visually plausible images, however, the modeller has no control over the category of the image. Supervised GANs extend the framework by including class information in order to achieve class driven sampling. Supervised GANs learn to control sample synthesis during training through conditioning. GAN models are conditioned in either the generator or discriminator (or even both) in one of the following ways;

- Directly concatenating labels at the input or later layers of both D and G.
- Through spatial bilinear pooling of class information.
- Applying the inner-product of the features layer and the label embedding, also known as projection discriminator.
- Learning a classifier alongside the discriminative model.

The most widely used supervised GAN frameworks are the conditional GAN (C-GAN) [126] and Auxiliary Classifier GAN (AC-GAN) [140].

3.2.1 Conditional GAN

Conditional GAN (C-GAN) [126] extends the original GAN models with conditional probabilities in the GAN value function. For a sample x drawn from the real data, the model computes the conditional probability $p(x|y)$ for some label y in the the set of class labels. Equation 3.5 shows the conditional probabilities in the value function of a C-GAN model. In practice, the supplied class labels is passed onto both the generator and the discriminator during training. Research has shown that label conditioning improves the quality of the generated images and also give control over the samples generated.

$$\min_D \max_G V(D,G) = E_{x\sim p_{data}(x)}[logD(x|y)] + E_{z\sim p_z(z)}[log(1-D(G(z|y)))] \tag{3.5}$$

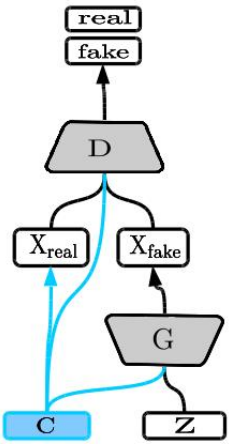


Figure 3.3: *Structure of the conditional GAN nodel.*

Where y is the label of the training data x which is the same as the intended generated image from the noise vector z. In the initial C-GAN experiments, labels were used as a one hot encoding input to both G and D. The usefulness of this framework was demonstrated on image tagging task using Flickr dataset [82]. Image features were extracted from a pre-trained AlexNet and a separate skip-gram model was applied to learn the language model. These were combined with C-GAN to generate tags for images. Figure 3.3 shows the structure of a conditional GAN model depicting the extension from the original GAN. In Figure 3.3, D and G represents the discriminator and generator, x_{real} represents the training data, x_{fake} represents the generated images, C is the set of class labels, Z is the noise vector and real/fake is the probability output which represents if the image is from the real or generated set.

Subsequently, numerous GAN frameworks have extended the capacity of the C-GAN model. For instance, Self-Attention Generative Adversarial Networks (SAGAN) [197] is a conditional GAN that compliment convolution with attention mechanism in image generation. SAGAN was able to generate highly detailed images at high-resolution on Imagenet dataset and reported a better Inception Score (IS) and Freschet Inception Distance (FID) than the existing state-of-the-art GAN model (see section 3.5.2 for more on IS and FID). In this context, self-attention layers captured global dependency and improved global image structure in generated images. This is crucial in capturing features that have geometric constraints.

Similarly, Brock *et al.* [26] developed BigGAN model that generates higher-resolution images with high fidelity. Their experiments were also run on Imagenet and result reported set a new state-of-the-art in Inception Score (IS) and Freschet Inception Distance (FID) from 52.52 and 18.65 to 166.3 and 9.6 (previously set by SAGAN). BigGAN is a *scaled up* SAGAN with a projection discriminator and spectral normalization in the

generator. The authors found out that a very large batch size is beneficial in the training of a GAN model and increasing width of the model as against depth only also scaled the model better. Conditional batch normalization and ResNet skip-connections ensure that deep models benefit from deeper architectures without losing label information.

3.2.2 Auxiliary Classifier GAN

Auxiliary Classifier GAN (AC-GAN) adds a classifier component to the GAN model. The classifier can share the layers of the discriminator and add a final softmax layer at the end or implemented as a separate model in the GAN framework. The idea was introduced initially in SGAN [138] where the classifier component was trained along with the discriminator model with an extra class label for generated samples. The model was tested on image generation and semi-supervised learning using MNIST dataset. Odena *et al.* extended this idea in [140] where the AC-GAN model is proposed. The framework uses generator conditioning and a classifier in the discriminator to push for control sampling. The AC-GAN model classified samples into existing classes while generating high-resolution images. AC-GAN uses two set of likelihood functions in training namely, the samples likelihood (L_s in equation 3.6) and class likelihood (L_c in equation 3.7). AC-GAN was used to address class perceptual diversity in generated samples and to generate more realistic images of real-world objects.

$$L_s = E[logP(S = real|X_{real})] + E[logP(S = fake|X_{fake})] \tag{3.6}$$

$$L_c = E[logP(C = c|X_{real})] + E[logP(C = c|X_{fake})] \tag{3.7}$$

The discriminator maximizes the sum of the two log likelihoods while the generator is trained to maximize the difference between $L_s - L_c$. Figure 3.4 illustrates the structure of an AC-GAN model with the classifier and class information highlighted in blue. AC-GAN model was tested on Imagenet and CIFAR-10 datasets and results showed improved sample quality with better IS score than existing state-of-the-art and good SSIM when compared to the real images. Although AC-GAN samples look more realistic than other models but the samples still fell short of being realistic. Also, AC-GAN model is affected by class diversity.

The AC-GAN framework has been extended into different domains and with different approaches to implementing the auxiliary classification. Such extensions include TAC-GAN [41], FM-GAN [181], VAC-GAN [17] and others. Bazrafkan *et al.* [17] suggested a

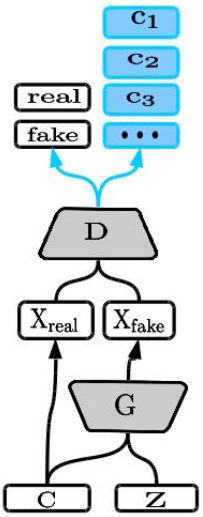


Figure 3.4: *Structure of the auxiliary classifier GAN model.*

Versatile Auxiliary Classifier GAN (VAC-GAN) which trains a separate classifier model alongside the discriminator model. The model generated better quality images than its predecessor (AC-GAN) through visual inspection and in terms of reported classification accuracy. The authors demonstrated that mixing the discriminator losses affected model convergence in training and may not apply to other GAN variants. Controllable Generative Adversarial Network ControlGAN [108] also implements a separate classifier outside the discriminator similar to VAC-GAN [17]. The difference between the two is that in ControlGAN make use of an equilibrium term in the objective function. This proved effective in generating images with detailed attributes such as arched eyebrows, big lips, wearing lipstick and wearing earrings which were evident through visual inspection.

An alternative conditioning approach that considers secondary labels other than the original class labels was proposed in Class Splitting Generative Adversarial Networks [65]. This is an extension to the AC-GAN framework that considers further conditioning of the generator in the latent space using learned features from training data. Secondary labels were obtained using K-means over the feature vectors of the last layer of the critic. The resulting clusters were then used as additional conditions on the generator latent space. This approach share some similarities with Info-GAN except that classes are learned from feature vectors as against a noise vector. The training algorithm supports both supervised and unsupervised training with the secondary labels as another source of conditioning.

It is clear that making the discriminator perform some extra tasks such are classification improves the GAN performance. What is not clear is the relationship between

classification and generative performances. The study by Shu *et al.* [163] brings to light the connection between auxiliary classification with the learned distribution and why AC-GAN has a high Inception Score. In this experiment, the authors demonstrated theoretically that, AC-GAN constraints the model to learn distributions in the right direction by avoiding sampling near the decision boundary. Such behaviour may imply that the model could learn incorrect distribution in certain scenarios. To demonstrate this ability, the authors experimented with MNIST, where they merged class 0 with 1 to form one class and 0 with 2 to form another class. The results showed that AC-GAN restricted the model to sampling only one from the first class and 2 when the second class is sampled because 0 is an overlap or fuzzy boundary between the two classes. The research also shows that AC-GAN trains to maximize the mutual information term in underlying inception score; hence, AC-GAN outperformed other models in terms of the Inception metric.

3.3 Improvements on GAN Framework

GANs are still evolving, and the research community is actively discovering better ways to best model a GAN framework. Nonetheless, we classify these improvements into three, namely architecture, an improvement on objective functions and improvement on optimization/training techniques.

3.3.1 Architecture

Architectural improvements are those enhancements that are related to the structure of the model and model layers used in a GAN framework. These improvements include using different layers like convolution layers or residual blocks, multi-modelling with other task such as classification along adversarial learning, using multiple D and G and other elegant techniques used in model design.

Deep Convolutional GAN

Both the discriminator and the generator were implemented in the original GAN paper as fully connected neural networks. On the other hand, research has shown that Convolutional Neural Networks (CNN) are known to work better in the image domain irrespective of the task at hand. To this end, Radford *et al.* [146] replaced the fully connected layers with strided convolutional layers in both the generator and the

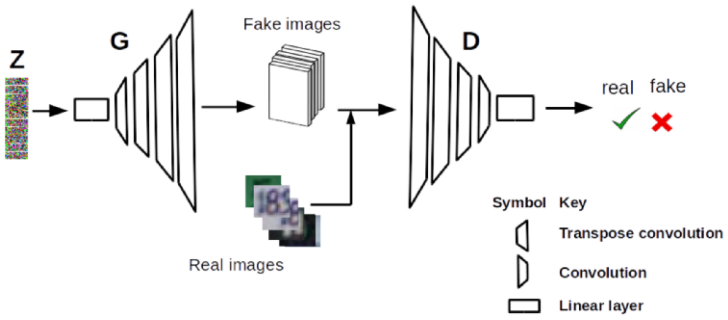


Figure 3.5: *Architecture of DCGAN.*

discriminator of the GAN framework. This class of GANs are called Deep Convolutional Generative Adversarial Networks (DCGAN). The improvement created stability for the model, even in higher resolution training and achieved better samples quality than previous approaches. Essentially, the DCGAN discriminator became a regular CNN without a fully connected layer while the generator upsamples noise into a structured image. The success of DCGAN in image generation has made them the standard structure adopted for subsequent GAN frameworks developed. Figure 3.5 depicts the structure of DCGAN.

AC-GAN, TAC-GAN and similar frameworks that implement a classifier alongside adversarial learning fall into the category mentioned above. Other architectural changes use more than one generator/discriminator in the image generation process. For instance, a double generator approach to GAN framework was used by Wang and Gupta [184] in generating images from image depth information. The framework is called Style and Structure GAN or S^2GAN, and it generates images by pipe-lining the process into structure and texture processes. S^2GAN has two separate generators (structure and style generator networks) that produce structure and style (texture) components of an image. Both of which are trained separately before a joint training with the discriminator. Structuring procedure makes use of depth information to generate scene structure from random noise. The output of which is fed into the style generator with some noise to generate a final photo-realistic image. The authors made use of pixel-wise constraint and Fully Convolutional Network (FCN) to align appropriately the structure and style components of images generated. S^2GAN achieve significant image quality improvements with 71% of Amazon Mechanical Turker, suggesting images were realistic. On the other hand, Nguyen *et al.* used a double discriminator architecture to train an adversarial three-player game [136]. The framework was called double discriminator GAN (D2-GAN). The authors showed that the new improved objective function is more stable and resistant to mode collapse than other approaches. The new objective is a combination of a KL and reverse KL-divergence.

Similar to [136] is Triple-GAN [37] which also uses a three player game. However, Triple-GAN does not use multiple generator/discriminator but rather a separate classifier network, a generator network and a discriminator network. They argue that the role of classification and discrimination are two opposing objectives that should not be combined in the discriminator functions [37]. This is particularly true considering that the two models may have different convergence points. Triple-GAN uses a classifier and generator conditioning to drive class specific sampling. The classifier C, produces pseudo-labels for each sample drawn from the true distribution and the generator is also sampled from noise by conditioning on true labels. These two sample sets are sent to the discriminator for discrimination. The third batch of sample labels and data from the true distribution (positive samples) are again used to train the discriminator. The authors used pseudo discriminative losses and reinforcement algorithm to train the model making sure that G and C improve each other as D approaches optimum.

Encoder-Decoder GAN

An auto-encoder GAN is a hybrid model that combines an auto-encoder model with a GAN either as an additional model or replacing either the generator/discriminator or both as the case may be. Existing research has shown that image generation benefits from this arrangement and also provides better stability in training. The adversarial objective is used to train this set-up and sometimes appended with an auto-encoder reconstruction loss or another form of regularization to guarantee convergence. For instance, Adversarial Learned Inference GAN (ALI-GAN), replaced the generator network with an auto-encoder network [52]. The encoder maps sample data to noise and the decoder maps noise prior to input space. These are both fed into the discriminator in an adversarial framework. The discriminator objective is now to detect sample from both models as well as propagate error into them. This allows the framework to learn example generation and mutual coherence inference which do not require conditional densities. It is worth mentioning that ALI-GAN does not rely on explicit re-construction of loss. Samples generated are clearer that previous GAN approaches and misclassification rate reported was lower than DCGAN and vanilla GAN.

On the other hand, Zhao et al. in [201] proposed a variant of GAN that uses an auto-encoder as a discriminator. The network uses an energy function to identify regions near samples and assign them to lower energy and higher elsewhere. Hence the name Energy Based Generative Adversarial Networks (EBGAN). The energy function is used as a reconstruction error, and a repelling regularizer is used to prevent the auto-encoder from generating samples around certain points consistently. Essentially, the regularizer term is used to prevent mode dropping; hence, EBGAN is stable in training and shows

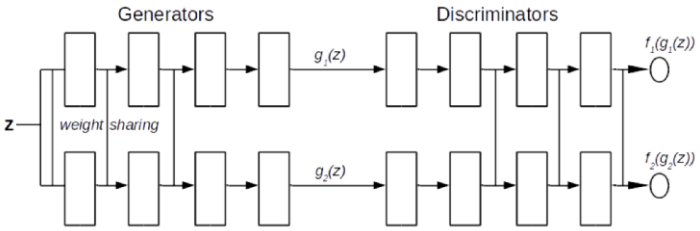


Figure 3.6: *Structure of Coupled GAN.*

better convergence with greater scalability.

Another similar models include the Boundary Equilibrium GAN (BEGAN) [19]. BEGAN employed auto-encoders and Wasserstein distance to produce images with high fidelity at high resolution. A novel approach to training using an equilibrium term that balancing the convergence of the generator and discriminator during training was also introduced by BEGAN. The equilibrium term in the objective function also controls the trade-off between visual quality and diversity of samples generated. Similar to [11], BEGAN does not match data distribution directly but rather auto-encoder loss using Wasserstein distance. Hence, the objective of the discriminator is to auto-encode real images and differentiate them from fake samples. Experiments were ran using Celebrity faces dataset with 360 thousand sample faces. Results obtained indicated an inception score of 5.62 which is only second to de-noising feature matching and better than all other GAN models available.

Hierarchical GAN

Hierarchical GANs split the image generation process into a pipeline with a series of generators applied to images usually starting from a lower to a higher resolution image. The target of most models in this category is to enhance the image quality as well as generate images at a higher resolution. For instance, the GAN framework was enforced in a Laplacian structure in tackling high-resolution image generation by Denton *et al.* in [47]. A cascade of conditional GANs were arranged to sample at each level a residual that is used to generate images at different resolutions with the labels coming from images generated at the previous layer in the pyramid. Each model was trained independently of the model in the preceding layer. This structure enabled natural image generation at a higher resolution than the original GAN network. These class of GANs are called Laplacian GAN or LAP-GAN. Results reported by the authors shows that LAP-GAN has better log-likelihood than the classic GAN and 40% of the human test mistaken generated images for real samples.

Similarly, Liu and Tuzel [114] proposed a multi-modal GAN that learns a joint distribution without the need for tuple inputs. Samples are drawn from two separate marginal distributions, and a pair of GAN models are used to synthesize images from two different domains. This GAN framework is referred to as Coupled Generative Adversarial Networks (COGAN). COGAN stands out from other approaches because of its weight sharing feature which was enforced in the first layer of the generator and the last layer of the discriminator. The intuition behind sharing first generator layers is based on the fact that the two domains share some high-level concept. Again, weight sharing reduces the number of parameters used in COGAN. Figure 3.6 shows the structure of COGAN. The training was carried out with back propagation in a constrained minimax game of two teams with two players each. The COGAN value function is shown in Equation 3.8.

$$\begin{aligned}\min_{f_1,f_2}\max_{g_1,g_2} V(f_1,f_2,g_1,g_2) = E_{x_1\sim p_{x_1}}[-logf(x_1)] + E_{z\sim p_z(z)}[-log(1-f(g_1(z)))] \\ +E_{x_2\sim p_{x_2}}[-logf(x_2)] + E_{z\sim p_z(z)}[-log(1-f(g_2(z)))]\end{aligned} \tag{3.8}$$

where f_1, f_2 and g_1, g_2 represents the discriminators and generators of the two GAN models and x_1, x_2 representing the two distinct domains. COGAN was tested on unsupervised domain adaptation task from MNIST to USPS [158] dataset and the authors reported improved classifier performance over the-state-of-the-art (two stream architecture in [152]).

Karras *et al.* [89] approached hierarchical image generation through progressively growing the GAN model during training in order to achieve high-resolution image generation. The model is expanded by adding new layers starting from a small image size until a high-resolution (1024 by 1024 by the authors) is reached. In this way, the model learns to generate a general image structure before fine image details are learned. This created stability in training as new layers are gently introduced with all layer trainable from the start to finish. The same condition remains true even with other loss functions like the least-square objective.

3.3.2 Objective Function

The original GAN objective function has many desirable properties; however, it is challenging to train with, and there is no certainty that the model will converge in all cases. These problems have led to the development of many GAN objectives that are easier to train, flexible with architectural choices and guaranty convergence under

certain conditions.

Wasserstein Objective

GANs like other neural networks are also affected by vanishing gradients. This could be as result of the logical loss employed which saturates quickly and may lead to mode collapse. [11], [154] and [176] suggest ways to improve training through careful selection of hyper parameters, employing techniques such as batch normalization and discrimination or using a different objective function in the model. Arjovsky *et al.* in [11] suggested the use of Wasserstein distance to tackle convergence issues in GAN models. By optimizing their GAN framework based on Wasserstein distance, the authors reported stable training which is directly related to distance minimization of the two distributions. The model then back propagate through to determine the new parameter set using Equation 3.9.

$$L(p_r, p_g) = W(p_r, p_g) = \max_{w \in W} E_{x \sim p_r}[f_w(x)] - E_{z \sim p_g}[f_w(g_\theta(z))] \tag{3.9}$$

where $W(p_r, p_g)$ represents the Wasserstein distance between the real data distribution p_r and the generated data distribution p_g, f_w is a true derivative among k-Lipschitz continuous functions. That is, for f, a real valued function $f : \mathbb{R} \rightarrow \mathbb{R}$ is k-Lipschitz continuous for some $k > 0$ such that $x_1, x_2 \in \mathbb{R}$, $|f(x_1) - f(x_2)| \leqslant k|x_1 - x_2|$ (see[3] for more details). Here, k is known as a Lipschitz constant for the function $f(.)$. The discriminator model is used to learn w to find the good f_w, hence, for some parameter θ the model train to converge f_w through gradient propagation by sampling in z. These class of GANs that makes use of this algorithm are called WGAN. In practice, k is kept small in between two carefully chosen bounds (1-Lipschitz constant) and gradient clamping is used enforce weight limits to a certain range. WGAN achieve stability in training with smaller distance/loss representing better image quality and the model generated images that are of higher quality than the original GAN objective. The objective function also removes some architectural restrictions.

WGAN bounds the discriminator by constants which are independent of the data distribution (a hyper-parameter setting). However, Mroueh and Sercu [133] argued that to imposed k constant on the critic (discriminator) is independent of data and distribution learned as such significantly reduces its capacity. This also limits its usability for a semi-supervised learning approach. In their research, they proposed a second order moment Integral Probability Metric (IPM) as a Rayleigh Quotient. This is a form of

[3] https://lilianweng.github.io/lil-log/2017/08/20/from-GAN-to-WGAN.html

weights control on the critic that is data dependant which they have demonstrated to be stable and maintains the capacity of the critic. The objective function uses an augmented Lagrangian as shown in equation 3.10.

$$L_f(p, \theta, \lambda) = \hat{\mathcal{E}}(f_p, g_\theta) + \lambda(1 - \hat{\Omega}(f_p, g_\theta)) - \frac{\rho}{2}(\hat{\Omega}(f_p, g_\theta) - 1)^2 \tag{3.10}$$

where λ is a Lagrange multiplier and $\rho > 0$ is a quadratic weight penalty, $\hat{\Omega}$ is a constraint and f is the critic. The authors reported stable training, faster convergence and a competitive semi-supervised learning results.

WGAN [11] have one of the most stable objective functions, however, the study by [66] showed that the weight clipping approach to control the gradients leads to some undesirable properties. Although the authors in [11] used some architectural choices (such as batch-normalisation) that limit the effect of these properties, experiments have shown that these effects were observed when deeper architectures are used [66]. One of such properties is underused of capacity by limiting the critic/discriminator to k-Lipschitz constraint thereby learning much simpler functions. Another feature is the case of vanishing and exploding gradients that were observed with deeper architectures. To address these problems, gradient penalty was proposed as an alternative to gradient clipping to push for the 1-Lipschitz constraint in [66]. The updated critic loss is shown in equation 3.11.

$$L = \mathop{\mathbb{E}}_{\tilde{x} \sim \mathbb{P}_g}[D(\tilde{x})] + \mathop{\mathbb{E}}_{\hat{x} \sim \mathbb{P}_r}[D(x)] + \lambda \mathop{\mathbb{E}}_{\hat{x} \sim \mathbb{P}_{\hat{x}}}[(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1)^2] \tag{3.11}$$

λ is a hyper-parameter known as the penalty coefficient. The proposed objective was tested on varying architectures(such as DCGAN and ResNet) and settings as compared to original GAN objective. Experiments were run on LSUN bedrooms and CIFAR-10 datasets and results reported showed improved training speed and sample quality. Furthermore, the generated samples had the best Inception scores (with state-of-the-art IS on CIFAR-10).

Softmax Objective

Lin in [111] proposed a change to the training objective of GAN by applying a soft-max cross entropy loss during training. This was proven to be equivalent to minimizing the Jensen Shannon Divergence (JSD) between generated and real data. The two loses L_D and L_G for the discriminator and the generator were proposed in Equations 3.12 and

3.13.

$$L_D = \sum_{x \epsilon B_+} \frac{1}{|B_+|} D(x) + lnZ_B \tag{3.12}$$

$$L_G = \sum_{x \epsilon B_+} \frac{1}{|B|} D(x) + \sum_{x' \epsilon B_-} \frac{1}{|B|} D(x') + lnZ_B \tag{3.13}$$

where B_+ is the mini batch set of real samples, B_- is the set of mini batches from the generated samples and B is the combine batches from both the generated and real set. Z_B is taken as a softmax partition function over the set B. This class of GANs are called softmax-GANs. Reduction in loss here also correlates to better samples generation.

Least Squares Objective

Least square GANs (LSGAN) [121] were created as an alternative objective to the GAN object to tackle the problem of vanishing gradients. LSGAN ensures gradient flow even for samples that fall within the correct side of the decision boundary but are not realistic. This pushes the generator to generate samples towards the decision boundary. Equations 3.14 and 3.15 shows the discriminator and generator objectives in LSGAN.

$$\min_D V_{LSGAN}(D) = \frac{1}{2} E_{x \sim p_{data}(x)}[(D(x) - b)^2] + \frac{1}{2} E_{z \sim p_z(z)}[(D(G(z)) - a)^2] \tag{3.14}$$

$$\min_D V_{LSGAN}(G) = \frac{1}{2} E_{z \sim p_z(z)}[(D(G(z)) - c)^2] \tag{3.15}$$

where a and b are the labels for fake and real data, and c is the value that G uses to fool D with fake data. The authors demonstrated that LSGAN minimizes the Pearson χ^2 divergence between the sum of $p_{data} + p_z$ and $2p_z$ for some $b - c = 1$ and $b - a = 2$ ($a = -1, b = 1, c = 0$). Least Square GANs are more stable than regular GANs and generates higher quality images than DCGAN [146] and EBGAN [201].

Multi-class Objective

This class of objective category have a hybrid objective that add an extra classification loss to the adversarial loss such as the ones in AC-GAN equations 3.6 and 3.7. Multi-class objectives are developed with labels and supervision in mind to achieve class driven sampling. For instance, Springenberg trained a multi-labelled discriminator for GAN in [169] (CatGan). Cat-GAN was used to learn class separation in data from few learned categories with class labels without explicitly modelling *p(x)*. This was realized through training a discriminator that learns multi-class classification as well as separating fake from real samples. The discriminator was trained using images of different categories and the generator is allowed to sample from uniform conditional distributed noise categories as well. The discriminator and generator (L_D and L_G) losses in CatGAN are given in Equations 3.16 and 3.17.

$$L_D = \max_D H_X[p(y|D)] - E_{x\sim X}[H[p(y|x,D)]] + E_{z\sim p(z)}[H[p(y|G(z),D)]] \tag{3.16}$$

$$L_G = \min_G -H_G[p(y|D)] + E_{z\sim p(z)}[H[p(y|G(z),D)]] \tag{3.17}$$

Information Maximizing GAN (InfoGAN) [34] adds a mutual information cost to the GAN formulation. The idea is to use a latent code c, to represent the salient semantic structures in images together with the noise vector as the input to the generator. Latent code represents features such as thickness of stroke, pose angle and other common image features. Different from other approaches is the fact that c is learned in an unsupervised manner. The latent code was enforced on the generator through variation information maximization by learning an auxiliary distribution over the latent codes ($Q(c|x)$). The InfoGAN objective is shown in equation 3.18.

$$\min_{G,Q}\max_D V_{InfoGAN}(D,G,Q) = V(D,G) - \lambda L_I(G,Q) \tag{3.18}$$

Where L_I is a variational lower bound, λ is a hyper parameter whose value is set to one for a discrete categorical latent code. InfoGAN was used to learn disentangled representations like the shape and stroke width of digits in MNIST dataset, background and number in SVHN dataset and azimuth and face angle in CelebA and 3-D faces.

Regularized Objective

Similar to WGAN, Qi in [145] developed a framework to train adversarial networks by using Lipschitz loss called Loss Sensitive Generative Adversarial Networks (LS-GAN). They demonstrate that the loss is a difference between the two distribution $p_{gen}(z)$ and $p_{data}(x)$. This difference is not a fixed constant and could grow smaller as the generator learns better density distributions. This novel approach to adversarial learning allows the network to improve on noise samples with high losses only rather than concentrating on samples with lesser loss. Model outputs show better fidelity of samples and high classification accuracy in a semi-supervised domain using a conditional variant of the objective. Equation 3.19 shows the LS-GAN objective.

$$S(\theta, \phi^*) = \underset{x \sim P_{data}(x)}{\mathrm{E}} L_\theta(x) + \lambda \underset{\substack{x \sim P_{data}(x) \\ z_G \sim P_{G^*}(z_G)}}{\mathrm{E}} (\Delta(x, z_G) + L_\theta(x) - L_\theta(z_G))_+ \tag{3.19}$$

Where λ is a positive balancing parameter,$\Delta(x, z_G$ is the margin between x and z_G, and $(a)_+$ represents $max(a, 0)$ for some fixed generator G^*. Mode regularize GAN [29] split adversarial training into two parts, namely, a manifold and a diffusion step. In manifold step, the generated images are matched with real ones using an encoder and an L2 loss. The second step ensures a "fair distribution of probability mass" on the generation manifold to make it as close as possible to the real samples. This objective was used to address missing mode problem in GAN training. The generator G and the encoder E, are optimized according to the regularized loss function in Equation 3.20 and 3.21.

$$T_G = -E[logD(G(z))] + T_E] \tag{3.20}$$

$$T_E = E_{x \sim p_d}[\lambda_1 d(x, GoE(x)) + \lambda_2 logD(GoE(x))] \tag{3.21}$$

where $GoE(x)$ is a composition model of the generator and an encoder, and *lambda* is a hyper-parameter. Another regularized objectives is Deep Regret Analytic GAN (DRAGAN) [95] which used regret minimization algorithm. The authors demonstrated that GAN experience mode collapse as result of local minima caused by sharp gradient from real samples in the discriminator. DRAGAN avoided mode collapse by regularizing the discriminator to constraint its gradient within the data space. In practice, gradient penalty [66] is applied and small perturbation was used to keep real data away from the data space.

3.3.3 Optimization and Training Procedures

Various enhanced approaches to optimize a GAN model have evolved over the years. The fact that GAN models are notorious to train has led to the proliferation of these techniques. Some of the techniques like batch normalization were extended from machine learning field while other techniques such as batch discrimination [154] were developed specifically for GANs. In general, we group these techniques into normalization techniques, regularization techniques and training tricks.

Normalization Techniques

Normalization techniques scale features/weights within a limited range. It is common practice in GAN to use normalized input (between -1 and 1), and a randomly chosen noise vector with a small standard deviation (say 0.02). Some of these settings are known to encourage stability in practice. For instance, batch normalization of inputs was first used in DCGAN [146] between layers to stabilize training and ensures better gradient flow within the network. Different from early GAN frameworks, DCGAN employed Adam optimizers as against directly training with momentum. Other normalization approaches include spectral normalization, weight normalization and layer normalization.

Xiang and Li showed how performance gain and improved image quality could be achieved using weight normalization [188]. They argued that batch normalization contributes to mode collapse and poor samples in GAN training. The weight normalized model showed consistent increase in quality of generated samples as the training progress. Again, experiments compared models using similar GAN architecture with and without batch normalization. They showed that training may be accelerated with batch normalization but was at the expense of the generated samples quality.

An optimal discriminator is always a problem for a GAN setup because its gradient at that point is approaching zero. To control the power of the discriminator Miyato *et al.* [128] proposed spectral normalization. Spectral normalization is essentially a stabilization technique that constraints the Lipschitz norm of the weights to one (bounded from above by one) with little overhead on the overall computational cost. Performances were compared to existing normalization techniques such as weight normalization, weight clipping and batch normalization using different losses and hyperparameter tuning. In almost all combinations, spectral normalized GANs achieved better results than other approaches considered (in terms of Inception Score, image quality and diversity) .

Regularization Techniques

Regularizers are employed to reduce the effect of over-fitting in deep learning. Some these are implemented on weight parameters while others are on the objective functions as discussed in section 3.3.2. Jacobian clamping [139], for instance, was implemented on the weight parameters. It has proved to reduce inter-run variance in GAN performances and sample quality/metrics. Experiments conducted with Jacobian clamping using existing models showed improved IS and FID on different datasets.

Roth *et al.* [151] introduced an analytic regularizer using noise. This technique stabilized GAN training by avoiding mode dropping and artefacts in training using different architectures and datasets. The authors demonstrated that noise-induced regularization is effective in making models immune to "dimensional misspecification". The stability of the model was tested on LSUN, CIFAR-10 and CelebA datasets. Results showed better stability in training with higher image quality than directly training with noise.

Training Heuristics

Training tricks are discovered from mundane use and have become a standard based on empirical results. For instances, Salimans *et al.* in [154] introduced feature matching, mini-batch discrimination, historical averaging, one-sided label smoothing and virtual batch normalization to encourage the convergence of GANs during training. These are heuristically motivated; however, results showed better performance from GAN models employed during experimentation. With feature matching the discriminator is trained to distinguish images based on image statistics of samples. This prevented the generator from overtraining on the discriminator. Mini batch discrimination, on the other hand, prevents the generator from collapsing by allowing the discriminator to screen samples in batches. Historical averaging uses the update parameters of players (D and G) based on previous value averages. Virtual batchnorm normalizes inputs based on initially chosen batch statistics. One sided label smoothing slightly reduced the confidence of a classifier on positive labels through reducing the tight bound, for example from 1.0 to 0.9. The reliability of these techniques is based on experimentation; hence, there is no certainty that any random combinations could yield good results.

More concretely, Evolutionary GAN (E-GAN) [182] used a fitness function from evolutionary algorithms to determine a healthy gradient update from a variety of GAN objectives used as mutation functions. The technique makes use of 3-variants GAN objectives namely, least-squares, mini-max and heuristic forming a set of estimated

parameters. A single version of the generator (the best) is maintained after each iteration. The fitness function stabilized the model in training and prevented mode collapse. Experiments were carried out on CIFAR-10, LS-SUN and CelebA datasets and E-GAN showed better stability to architecture deformation than the approaches considered while producing visually plausible samples. However, E-GAN takes longer to train.

3.4 Application Domains

GAN models are state-of-the-art in estimating distributions and in image generation. The flexibility of the GAN framework makes them suitable for many computer vision tasks. Some of these areas includes semi-supervised classification, image tagging, image generation, image correction, face ageing and domain adaptation tasks.

3.4.1 Face Ageing

Face ageing has many applications such as security, search for missing persons, information management and recognition systems; hence, face ageing is an active research area in computer vision. With the advent of deep CNN in face verification, performances have surpassed human level on popular face benchmarks [175, 157]. Despite several successes in this domain challenges still, exist such as cross-age verification and face identification in unconstrained environment [7]. The uniqueness of facial structure and various ageing process creates disorder and inconsistencies in facial images [116]. Moreover, getting facial images of the same subjects at a specific age can be very challenging [185].

Therefore, a successful age-specific generation method model the age of the samples while preserving their identity. Different GAN models have been developed to fulfil these criteria. For instance, age-specific generation in GAN was achieved by using a CGAN through embedding age information in the generator and discriminator [8]. Similarly, AC-GAN framework was employed by [7] to induce age information through classification and generator conditioning. Again, Antipov *et al.* in [7] used a restricted latent vector to ensure consistency alongside a perceptual loss in generating age-specific facial images. Others preserve the identity of the subject by using various techniques such as style transfer [166], cycle consistency loss and perceptual loss [185] along with a deep generative model.

Furthermore, Liu *et al.* [116] personalized the face ageing process according to age features learned using Age-DCGAN. Age-DCGAN is based on DCGAN [146] architecture

and include label conditioning in both generator and discriminator. A personality latent vector z, with age features y' are extracted from a high dimensional image using encoders. Both z and y' are then passed to a generator to create an image which is sent to the discriminator in a typical GAN set up. Age-DCGAN uses a second discriminator to distinguish y from y'.

Identity Preserving Conditional GAN (IPCGAN) [185] also uses a perceptual loss between generated images and the original image to preserve the identity of the subject in the image. Perceptual loss consists of a style and content loss similar to [116]. IPCGAN has three components; a CGAN model, identity preserving module and a pre-trained AlexNet [99] as an age classifier. Experiments were conducted using CACD dataset and samples were evaluated by employing people to visually inspect the samples and rate the quality of generation, verify face identity and age classification.

The GAN generator is a versatile model but capturing the exact data distribution with tiny details is elusive even with an optimal generator and a large dataset [7]. Local Manifold Adaptation (MLA) [7] uses age normalization method to model age progression/regression in synthetic faces. MLA constrain the generator latent space to a learned facial encoding jointly with age-specific encoding. This uses an Age-cGAN model to preserve the identity of the subject as the sample is rejuvenated or aged. The authors synthesized facial images of varying age categories, then normalize these faces prior to face verification. Normalizing across age categories provided further invariance to age thereby, improving the performance of deep learning face verification software across different benchmarks.

3.4.2 Image Correction

Traditional image enhancement uses many parameters and is often down to the skill of the user which takes significant amount of time to tweak [35]. Chen *et al.* [35] proposed learning an image enhancer using a GAN model. Unlike other approaches, this model only relied on a set of images with the required characteristics as inputs. Enhancement was achieved by transferring the desired characteristics on the image from the set of input images while preserving the identity of the image. This approach shared some characteristics with image-to-image translation using CycleGAN. The authors proposed a 2-way GAN model using U-Net as the generator trained with an adaptive Wasserstein objective. Experiments were carried out on MIT-Adobe 5k dataset and results obtained showed higher image enhancement quality than DPED, 8RESBLK, CRN, FCN and UNet in term of PSNR and SSIM.

Shrivastava *et al.* in [162] describes a framework that refined synthetic images to

make them more realistic. The authors demonstrate that models trained on refined images perform better than those trained on synthetic images. Images are refined using unlabelled real samples. The framework consists of a refiner network that is built based on ResNet architecture. This refiner network is trained using adversarial loss in a GAN fashion. Self regularizing loss was also introduced to preserve annotation information in refined images. Image artefacts were controlled through local losses and model convergence was enhanced through buffer refinement, and current batch loss to update the discriminator similar to historical averaging in [154]. The authors reported results better than the current state-of-the-art in gaze and pose estimation using MPIIGaze and NYU hand pose datasets by a 21% and 8% margins respectively.

Schawinski *et al.* [156] applied GAN in reconstructing astrophysical images. Images used for this experiment were collected from Sloan Digital Sky Survey(4550 images in total, 4105 for train and 455 for testing). A second image set was generated by applying random Gaussian noise. The two image sets were used to train a GAN that learns to regenerate the original images. The results showed better performance compared to deconvolution techniques with a better PSNR ratio of 37.2DB. However, it was also reported that the model was only able to recover image parts that have been learned during training.

A context based semi-supervised learning framework using adversarial loss was proposed by Denton *et al.* in [46] called Context-Conditional Generative Adversarial Networks (CC-GAN). This was tested on in-painting of image patches and semi-supervised classification task. The model architecture of the generator is inspired by DCGAN in [146]. The discriminator was a VGG11-A [164] (without the fully connected layers). The generator takes as input the image with a hole and a pathway to a reduced sized image as the conditioned context. The in-painted and generated image were fed separately as input to the discriminator during training. This framework was tested on STL-10 dataset and PASCALVOC07, results showed visually plausible patches were generated by different model combination.

3.4.3 Image Tagging

The initial CGAN model was tested on Flickr images to generate more descriptive tags from annotations and generated meta data [126]. CGAN demonstrated the ability to accept language embeddings and labels in producing the descriptive text. On the other hand, Reed et al. trained a variant of GAN (GAN-CLS, GAN-INT) to generate realistic images from text embeddings [150]. The model translated a textual description into an image. The framework takes noise drawn from a distribution and concatenates it with

a sampled text query. The discriminator is trained to learn correct text embeddings with realistic samples. The procedure was tested using Oxford-102 dataset, CUB-200 dataset, Caltech-UCSD and MSCOCO dataset. Results obtained showed that the model was able to take a word embedding and generate plausible images when sampled. Similarly, Text Conditioned Auxiliary Classifier GAN (TAC-GAN) [41] synthesizes an image from text descriptors using a variant of AC-GAN.

3.4.4 Network Visualization

GANs were recently applied in understanding the neuron activation in a trained model by Nguyan *et al.* [135]. They applied a deep generative network to learn image prior in an activation maximization problem. This allowed the synthesis of high stimuli for different neurons and consequently more realistic visualization of activations. The authors used the GAN architecture and objective function in [197]. Different from other approaches, this prior is capable of estimating neuron activation on other datasets without relearning the model but with some architectural constraints. When the encoder and the network to visualize have different architectures, the visualisation quality degrades with how far apart the two models are. This was tested using a trained model from the Caffe model zoo (CaffeNet, GoogleNet, ResNet) and visualisations produced a high level of realism in the images.

3.4.5 Semi-supervised Learning

Semi-supervised learning is applied where labels are not available or difficult to obtain. One of the early semi-supervised GAN frameworks is [138] where Odena trained a classifier using a GAN on MNIST which was used to classify withheld samples. Subsequently, state-of-the-art results were reported by [154] on MNIST, SVHN and CIFAR-10 datasets on semi-supervised classification. ALI-GAN was also tested on SVHN and CIFAR-10 for semi-supervised learning with and without feature matching and in both instances ALI-GAN out-performed all GAN models [52]. However, the results obtained were still below the supervised and best semi-supervised approaches. Dai and Yang in their study [40], demonstrated that a perfect generator prevents the learning of a good semi-supervised discriminator. Consequently they proposed a complementary generator for learning a good discriminator for semi-supervised learning. They also reached similar conclusion as [52]; that feature matching GANs performs better in semi-supervised learning. There procedure was tested on MNIST, SVHN and CIFAR-10 datasets, and the method showed consistent performance gain on the benchmarks when feature matching was used.

Semi-supervised learning using GAN was re-visited by Perez *et al.* in [143]. They applied WGAN in predicting poverty from satellite images. Their dataset was a set of images over Africa. However, only 5% of the data was labelled. Their critic (discriminator) was a DC-GAN with a ResNet-50 structure. Losses where added in a multi-task learning fashion. The model was allowed to predict fake/real classes with other classes that corresponds to Assets Wealth Index (AWI). EBGAN [201] was also tested on semi-supervised learning on PI-MNIST with 100,200 and 1000 labels considered. Results reported shows improvement on former ladder networks' bottom-layer-cost model approach.

3.4.6 Domain Adaptation

Domain adaptation tries to transfer representations learned between two different domains e.g painting to real images (source to target). GAN-based domain adaptation techniques have made significant progress in both supervised and unsupervised settings. The most successful GAN based domain adaptation frameworks is the CycleGAN [203].

CycleGAN introduced the cycle-consistency loss that learns two separate GANs. Thus, when the two are connected produces an identity mapping between a source and a target domain. The significance of this was that the two distributions are completely disjoint with no need to pair training samples. This was demonstrated in style transfer examples between zebra and horse, summer and winter images and so on. Numerous other successful extension of this followed such as DiscoGAN [90], DualGAN [192] and StarGAN [36]. StarGAN [36] revolutionized the domain adaptation approach by adapting CycleGAN to multi-domain image-translation using a single model.

Similarly, Sankaranarayanan *et al.* [155] used a GAN architecture with a parallel feature extraction and classification networks to learn a joint feature distribution in an unsupervised domain adaptation problem. Contrary to other approaches, this technique works even in extreme conditions where image generation is challenging (limited number of samples in classes).

In medicine where computer aided diagnostics have become very popular, domain adaptation is one technique with ability to reduce diagnosis error and patients' exposure to risk of testing procedures. For instance, exposure to radiation during CT scans is known to be a source of cancer at later stages in life [137]. As such there has been a drives to generate CT images from other sources that are less lethal. Nei *et al.* estimated CT scan images from MRI images in [137]. The technique trained a GAN model on a pair of CT and MRI images of the same subject. To work around high image resolution, the procedure was carried out on patches that were later averaged into a

target image. Image gradient loss was used to reduce the blurriness of the samples generated. The model was evaluated using Peak Signal-to-Noise-Ratio (PSNR) and Mean Absolute Error (MAE) and results obtained showed that the model out-performed the state-of-the-art SRF+ method.

3.5 Evaluation

Evaluating a generative model is still an open question for research. Some researchers often use the fidelity of samples while others try to estimate the distribution learned through log likelihood estimate. GANs are evaluated using quantitative and qualitative measures and both of these metrics have strengths and weaknesses. For instance, quantitative metrics may not detect mode collapse or lack of diversity in generated samples while qualitative measures may not necessarily correspond to human perception due to variety of probabilities and lack of meaningful image similarity metric [22]. Human judgement has also been used to evaluate the quality of samples in GANs [154, 47, 146]. The following sections categorize some of the popular metrics used in evaluating GAN model performances.

3.5.1 Likelihood Estimates

The target of a GAN model is to capture the data distribution and an optimum generator should produce samples very close to the real data distribution. Thus, an ideal metric should measure the distance between the two distributions. However, likelihood is not feasible at high dimensions; hence, the alternative log-likelihood is used in this regard. The initial experiment in [63] used a Gaussian Parzen window to estimate distance between the distribution p_{gen} and p_{data}. Results shows a competitive value compared to existing generative models. Similarly, [47, 126, 59] also used Parzen window to estimate log likelihood. However, later researches have shown that these estimates do not always reflect other factors like sample quality [176].

3.5.2 Inception Metrics

The two widely used metric benchmarking GAN models are Inception Score (IS) and Fréschet Inception Distance (FID) [22]. Salimans *et al.* in [154] suggested the use of Inception model to approximate conditional class metric over data. Inception model is a trained CNN on Imagenet dataset by Szegedy *et al.* [173]. This is calculated as the mean score of correctly classified samples ($p(y|x)$) and a marginal probability over the

classes ($p(y)$). These two criteria are computed using Kullback-Leibler divergence as shown in Equation 3.22 . The inception score metric was found to correlate well with human judgement. Subsequently, this metric was employed by a number of researchers as a way to evaluate the performance of models and compare them to others such as [19] and [201]. However, Inception Score does not capture model diversity or account for when the generator collapsed to producing similar samples [140].

$$IS(G) = E_{x \sim p_g} D_{KL}(p(y|x)||p(y)) \tag{3.22}$$

Fréchet Inception Distance (FID) [75] addresses some limitations of Inception Score by modelling the distribution learned using the mean μ, and covariance matrix Σ, of the features extracted from the inception model. Equation 3.23 shows how FID is evaluated where Tr is the sum of all diagonal elements. The statistics of both the real and fake images are compared and a small FID indicates closeness and better quality images. FID is more resistant to distortions and correlates better to human perception. Moreover, FID metric is more meaningful to other datasets containing non Imagenet classes.

$$FID((\mu, \Sigma), (\mu_g, \Sigma_g)) = ||\mu - \mu_g||_2^2 + Tr(\Sigma + \Sigma_g - 2(\Sigma.\Sigma_g)^{\frac{1}{2}}) \tag{3.23}$$

3.5.3 Classification as a Metric

The problem with Inception metrics (IS and FID) is the reliance on pre-trained Imagenet model. These metrics may be inadequate when considering a dataset that has different classes other than Imagenet [14] such as faces. Moreover, when classification is the target, perhaps a metric that captures class property may be more appropriate [161]. GAN*train* and GAN*test* [161] evaluate the diversity and quality of generated images using a classifier. With GAN*train*, the classifier is trained on generated samples and real samples are used as test images. For GAN*test*, the classifier is trained on real images and tested on generated image. The idea is that if the generator has captured the true data distribution, the difference between the two accuracies should be very small. These two metrics were found to correlate well with precision and recall of generated samples. GAN*train* and GAN*test* was used to evaluate condition WGAN-GP, SNGAN, DCGAN and PixelCNN++ models on MNIST, CIFAR-10, CIFAR-100 and Imagenet datasets. Compared to FID and other metrics where model performances are within a certain range (especially for best models), wide margins were recorded from GAN*train* and GAN*test* accuracies for the same models. These two metrics were able

to capture subtle performance differences amongst the best models.

Away from log likelihood estimates, researchers also try to evaluate their models by using other supervised models. For instance, DCGAN was evaluated as a features extractor on a supervised learning task by fitting an SVM on the unsupervised learned representations [146]. The results of classification on CIFAR-10 shows that DCGAN+SVM is only second to exemplar CNN. The results demonstrated good generalization ability because the model was pre-trained on Imagenet-1k. This suggest transferable features are indeed learned by the model. A similar experiment was ran using SVHN and state-of-the-art results were reported. Similarly, classification accuracy on generated data was used by Wang and Gupta [184] to test the quality of samples generated.

3.5.4 Other Quantitative and Qualitative Metrics

Popular quantitative metrics employed include mode score [29], Maximum Mean Discrepancy (MMD) [64], Wasserstein critic [11], Classifier Two-sample Test (C2ST) [109], classification performance [85], Generative Adversarial Metric (GAM) [83], reconstruction error [188], SSIM [140] and others. Other qualitative measures are nearest neighbours, rapid scene categorization [63], preference judgement [198], mode drop and collapse [171] and network internals [34]. Borji [22] argued that any evaluation criteria employed should have low sample and computational complexity, favour the model with high sample fidelity, diversity and disentangled latent spaces. The model should also have well defined bounds and be sensitive to image distortion and above all agree with human perceptual judgements. Thus, the evaluation criterion chosen should have both qualitative and quantitative properties. To check for mode collapse and ensure variability of samples generated by AC-GAN, Odena *et al.* used multi-scale structural similarity measure (MS-SSIM) as a metric [140]. Mean MS-SSIM of training set was calculated and was used to ensure that the generator never collapse to generating similar samples. Similarly, Bousmalis *et al.* employed a content similarity loss to control variance in both generated and training samples [25]. Re-construction error was also used by Xiang and Li in [188] to assess the quality of generated samples. This is an alternative to negative log likelihood estimates which require generating many samples to compute close to accurate results. They try to estimate the latent representation of an image using gradient descent starting from a zero vector. Although the method is slow, the authors used it as a post training step.

In general, choosing a metric to measure performance is a bit tricky. According to Theis *et al.* [176] the choice of an evaluation metric for a generative model depends on the objective for which the framework was designed. The study focused on the popular

methods of evaluating probabilistic generative models such as direct log likelihoods, Parzen window estimates, nearest neighbours and fidelity of samples. According to their findings, good results in one metric does not translate to good results in another or the overall model. For instance, log-likelihood and visual fidelity of models were found to be mostly independent, and in high dimensions, Parzen windows estimates are far from true likelihood. The authors concluded that there is a trade-off in choosing a metric for evaluation and is subject to the purpose on which the model was designed. In order words, if the purpose of a model is to generate samples only, the visual fidelity of samples will be a good enough metric.

To summarize, GANs have proved to be the best available estimates for data distributions, and it is evident that they can generate images that are plausible enough to be a true representative of the dataset. The advancements in the value functions and architecture of the GAN framework addressed similar challenges such as stability on different datasets and metrics. The question then becomes which one of these is a better representation. According to Lucic *et al.* in the article *"Are GANs Created Equal?"* showed how the performance of some major GAN algorithms is dependent on design choices such as datasets, hyper-parameters and computing budgets. Hence, comparing them on reported minimal FID and inception score may not be enough. Both inception score and Frechet Inception Distance are flawed [120]. They argued that while both correlate to image quality they fail to identify a "memory GAN" and struggle with mode collapse in GANs. They propose that F1 score be used to complement the robustness of Inception based metrics namely, FID and IS. Therefore, not only are the metrics dependent on the target of the experiment but also a good enough evaluation uses more than one metric to arrive at a more concrete conclusion.

3.6 Chapter Summary

This chapter discussed the concept of GANs with their limitations and the progress made so far in different areas. We discuss in details the architecture of a typical GAN model and the improvements made in terms of the structure, the training procedure/optimisation and the objective function. We also discussed in detail supervised GAN models and some methods used in evaluating the performances of a GAN. Notably, we discussed supervised GANs like AC-GAN [140], optimisation techniques such as spectral normalization [129] and DC-GAN [146] which have become the standards in training a supervised GAN. The next chapter, investigates further the performance of state-of-the-art deep models in a poorly annotated data.

Chapter 4

Deep Learning and Labeling Dilemma

In this chapter, we analyze the performance of deep convolutional models on poorly annotated data. We demonstrate empirically the challenges of fish recognition in the wild. In particular, we examine the performance of transfer learning and traditional augmentation approach in the absence of well-annotated training data. The findings in this chapter appeared in the proceedings of the 18th International Conference on Engineering Applications of Neural Networks (EANN 2017).

4.1 Need for Labeled Data

One of the factors that has influenced the recent advances in applying supervised deep learning to image detection and classification is the availability of massive and cleanly annotated datasets with numerous categories (like Imagenet). However, such datasets are not always available, and the process of building one is mostly manual, labour intensive and time-consuming [12]. This is because real data such as images and videos can be very noisy, complex and redundant with extreme values. These challenges increase the workload and cost of preparing a massive dataset from scratch. Imagenet, for instance, follows the ontology of WordNet[1] and even with the manual process and crowdsourcing, Imagenet are yet to cover all the noun categories of WordNet which was the initial goal. The 1.2 million images that were annotated initially required nineteen man-years [127] and WordNet ontology is still growing even after the inception of

[1] https://wordnet.princeton.edu/

Imagenet. Another example is Youtube videos[2], which receives about 100 hours of uploads every minute (ten million frames). To use humans for labeling every frame is almost close to impossible, given the size of data. This leaves a very high percentage of the available data unlabeled.

Established deep models such as YOLO [149], SSD [115] and R-CNN [61] used in detecting and classifying objects are examples of AI's attempt to automate the annotation process. However, training these frameworks still rely on massively labeled data to achieve best results. Moreover, the number of objects covered by these models is limited. Thus, extending to more objects means re-annotating and re-training. In specific domains, that are distinct from the original domain, transfer learning may not necessarily work. Hence, a data collection to train a model from scratch may be necessary.

4.2 Acquiring Labeled Data

Data acquired is mostly unlabeled and may require many preprocessing steps before they can be used. One of such process is annotation and it can be achieved with the help of annotation software or tool. This process is largely manual and a massive dataset will require a lot of resources. Annotators use manual annotation tools such as Sloth[3], LEAR[4], VGG Image Annotator[5] and FastAnnoation[6] as standalone tools for annotating images. Annotation tools such as LabelMe[7] are available online. For online tools, there may be privacy issues with data sources and confidentiality. This leaves us with the question of whether it is possible to always rely on crowdsourcing whenever a new dataset is to be built. Moreover, crowdsourcing is an expensive and prolonged process. Figure 4.1 shows an example of annotation process by drawing bounding box around fish instances using Sloth.

Some of the challenges of existing image annotation systems were outlined by Russell *et al.* in [153] as what to label, complexity, validation and the label text itself. The downside of these tools is that users will have to annotate the images manually. Hence, accuracy is strongly reliant on human annotators. Also, controlling the quality of annotations is a source of concern. Some researchers addressed some of these issues by proposing validation techniques. Automatic and selective validations are employed [21,

[2]https://research.google.com/youtube8m/
[3]https://sloth.readthedocs.io/en/latest/
[4]https://lear.inrialpes.fr/people/klaeser/software_image_annotation
[5]http://www.robots.ox.ac.uk/vgg/software/via/
[6]https://github.com/christopher5106/FastAnnotationTool
[7]http://labelme.csail.mit.edu/Release3.0/

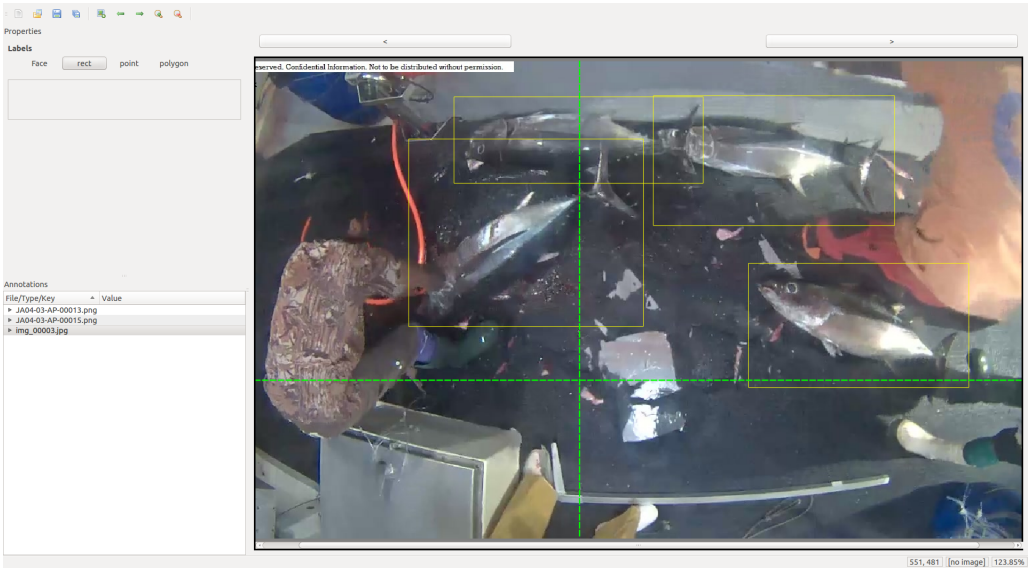


Figure 4.1: *Image annotation and labelling by manually drawing bounding boxes around fish species using Sloth.*

44] to reduce the effort or cost of annotation. Others leave this open to the annotator to decide what to label and how to segment it [153]. An approach to the complexity of labels could be to limit the label class by providing a pop-up from which users may select. However, this is only applicable when defined classes are established.

4.3 A Case of Fish Classification

In this section, we consider the *Fish Classification* problem as a case study. We aim to assess the performances of established deep learning models under different scenarios. The experiments are designed to show how deep models are affected by the number of labeled examples. We start from raw data to annotation, to training and testing. We also examine common techniques used to address some of the challenges along the pipeline.

Fish detection and recognition are important for conservation agencies, marine life scientist, fishing industry and governments to maintain fish supply and balance in the ecosystem. Increase in continental reef monitoring, and deep-sea surveillance has created the need for more imagery analysis. In this domain, images are obtained from mounted cameras that capture continuous data for marine biologists. The rate at which data are generated from underwater cameras, fishing boat cameras, Automatic Underwater Vehicles (AUV) and conveyor belt cameras challenges human manual approach to count and sort fish species. Therefore, image-based techniques are now popular in

this domain [168, 24, 18].

Because of its economic importance, many approaches have been proposed in the detection and classification of fishes. Researchers employ specialised software and hardware to monitor the marine eco-system. This has helped them in studying fish species behaviour [168], classifying fishes into different species [124, 43], counting individual species and also tracking their movements[103]. To support the growing needs of the research community, competitions such as Kaggle[8] and Seaclef/LifeClef[9] provide richly annotated datasets for researchers to push the research frontiers.

However, challenges still exist in identifying fish species from images and videos. In this domain, images obtained here are largely noisy and are affected by illumination. Furthermore, camouflage and presence of multiple objects in a frame affect segmentation and subsequent localization of the object of interest. Hence, successful techniques rely heavily on preprocessing to achieve good results [110, 24, 103, 18].

In this regard, we investigate the performance of deep convolutional neural network in the context of noisy images. We hypothesize that deep learning-based methods performance will deteriorate when lacking clean well-labeled set of images. To demonstrate this, we build an experimental framework to evaluate the performance of deep models using a challenging and complex set of images provided by Kaggle[10].

4.3.1 Experimental Framework

This section describes the techniques used in the study. Details of CNN architecture and model initialization are also discussed.

VGG Model

VGG networks were proposed by the Oxford Visual Geometry Group (VGG) [164]. These networks are 11, 13, 16 and 19 layers deep, also known as VGG-11, VGG-13, VGG-16 and VGG-19. These models ranked first and second place in the Imagenet classification challenge in 2014. The models are one of the most widely used CNN models in image classification today. For this experiment, we considered an untrained VGG-16 network. The model contains five blocks of 13 convolution layers and three fully connected layers. It makes use of a filter size of three for all convolution layers. It also employs a max-pooling layer between successive convolution blocks with a unit

[8] https://www.kaggle.com
[9] http://www.imageclef.org/lifeclef/2016/sea
[10] https://www.kaggle.com/c/the-nature-conservancy-fisheries-monitoring

stride for downsampling. The three fully connected layers contains 4096, 4096 and 1000 ReLu activated units respectively(see [164] for details). The VGG-16 network architecture is shown in Figure 4.2.

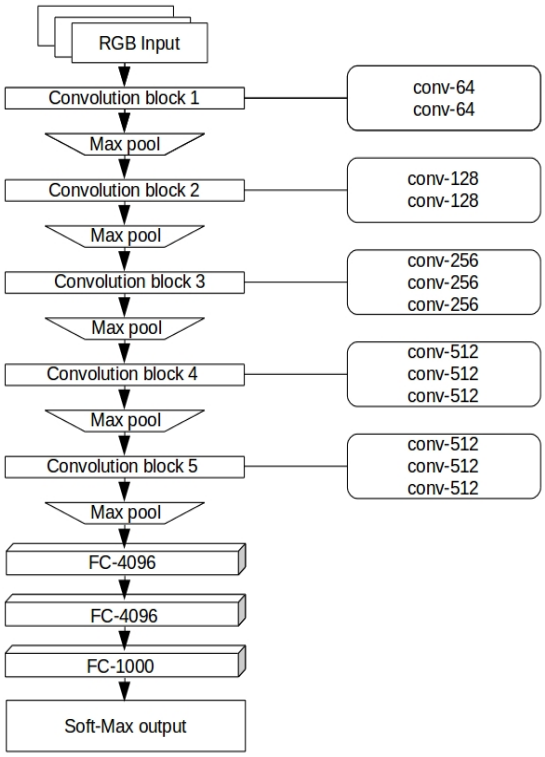


Figure 4.2: *VGG-16 architecture*

Given that the dataset used has only eight categories, the final layer was replaced with an 8 way soft-max classifier to suit experiment.

Transfer Learning

The second approach used was proposed by applying transfer learning to the VGG-16 model from a pre-trained network on Imagenet. Transfer learning attempts to reproduce similar results from experience on a previous task. Transfer learning enables a new model to inherit learned parameters from a model trained on another task. This has proved to be effective where training images are scarce [54, 193]. The intuition behind this is to have a model that has already converged for comparison purposes. Model architecture is the same as the one in 4.3.1 but its weights and biases were initialized from learned parameter after training on Imagenet dataset. The motivation behind using transfer learning is that given the size of the dataset, we try to fine-tune

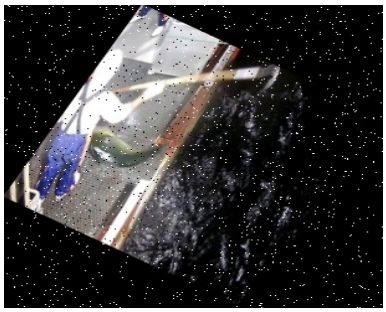
the network as against learning new features from scratch with the hope that better results could be achieved.

The Fish Datasets

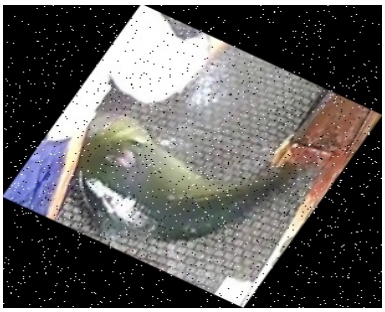
In this experiment, we used a dataset of images provided by Kaggle[11]. It contains 3777 images of fishes. The fish categories include Albacore tuna, Bigeye tuna, Yellowfin tuna, Dolphin, Lampris guttatus, Sharks, other categories and images with no Fish, labeled as ALB, BET, YFT, LAG, DOL, SHARK, OTHER and NoF. It is worth pointing out that these images were extracted from video footage of fishing boats. Fish detection in these images is challenging even to humans. Light variation in images, presence of multiple objects, pulse variation and partial occlusion, makes fish recognition very challenging. A sample image from this dataset is shown in Figure 4.3a. Again, there is a significant class imbalance among fish classes with almost 70% of the samples in ALB and YTF classes.

(a) Original Image.

(b) Annotated Image.



(c) Original + Noise Image.



(d) Annotated + Noise Image.

Figure 4.3: *Sample fish images used in training.*

A second dataset was generated from the original images by annotating all the images using Sloth[12]. The new dataset contains 3777 fish images. Annotation was done by

[11] https://www.kaggle.com/c/the-nature-conservancy-fisheries-monitoring
[12] http://sloth.readthedocs.io/en/latest/

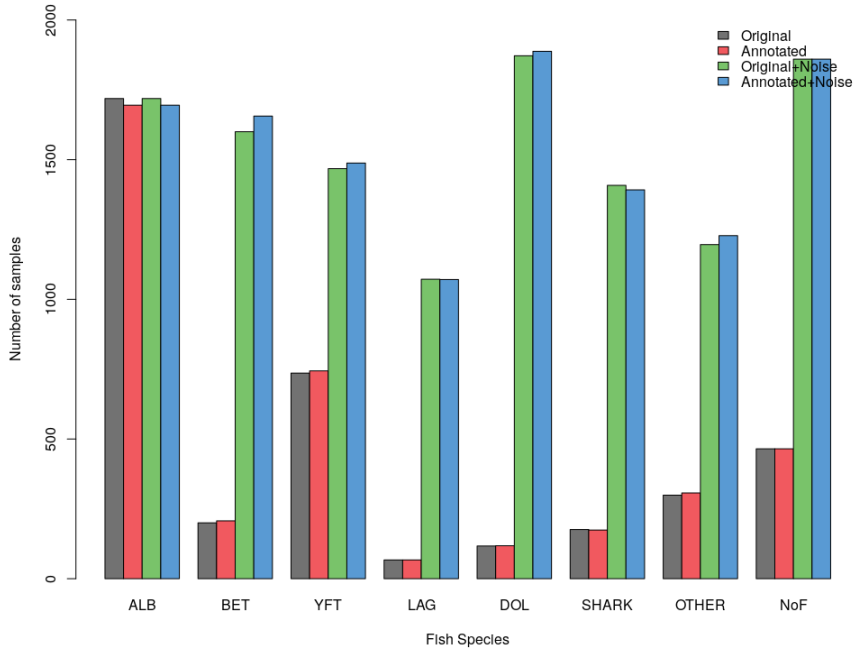


Figure 4.4: *Distribution of fish images in the different datasets.*

isolating individual fish instances from an image using a bounding box. The bounding box was made big enough to accommodate the fishes and incorporate other surrounding objects to maintain variability in the training set. Annotated instances contain a complete fish with head and tail visible or partially occluded head or tail but not both. It was observed that images are very similar in terms of the environment but differ with object view angles, light variation and times with shadows. These differences were considered visible enough to distinguish adjacent image frames as such no further cleaning was required. Figure 4.3b shows the result of this process.

A third dataset was created from the previous datasets. Images were generated from both the original and annotated images. The motive behind this is to address the biased nature of image distribution among classes. We also intend to achieve optimum model performance with more data. The new dataset contains 12,275 images across 8 categories. These images were synthesized by applying random noise and affine transform. Images were distorted using varying degree of rotation angles (between 15 and 105 degrees) and noise intensities. This is similar to the work done by Dostovistkiy in [50] to generate training samples. This is to create enough distortion to generate distinct images from the originals. The result is shown in figure 4.3c and 4.3d. Figure 4.4 shows the distribution of samples in every class in all the datasets and a summary of the datasets is presented in Table 4.1.

Table 4.1: *Summary of the fish datasets used.*

Dataset	Number of Images	Noise	Affine
Original	3777	✗	✗
Annotated	3777	✗	✗
Original+Noise	12275	✓	✓
Annotated+Noise	12275	✓	✓

Model Initializations and Settings

Experiments were ran on NVIDIA DGX-1 machine[13]. Full advantage of the multiple GPU system was taken and this significantly reduced training and test time. The models were implemented using Keras[14] with tensorflow[15] back end. Before training was initiated, all images were resized to 224 by 224. This is to accommodate them in the VGG-16 model. Each model was trained using all datasets described above. At the beginning of training, images were shuffled, then split into test and train with 75% of data used for training and the remaining 25% of data for testing. Experiment on VGG-16 was carried out using a learning rate of 10^{-2} over 16 epochs and training was done using stochastic gradient descent with a batch size of 32. A weight decay was chosen as a ratio of learning rate to number of epochs and a momentum of 0.9 was maintained. The settings were to ensure faster convergence of models. Dataset normalization was applied by simply dividing each pixel by 255 for both training and test set whereas the original VGG-16 experiment normalized by subtracting the mean pixel value from each pixel. This does not affect model accuracy but training time. We also differ in the choice of weight decay because subsequent experiments revealed that a dynamic weight decay works better than a statically chosen one. Training batch size was significantly lower than the one proposed in VGG-16 because the problem has significantly smaller dataset. Moreover, with smaller batch size shorter gradient updates can be realized. Apart from resizing, no further preprocessing was applied. During testing, we did not employ random crop or other methods as in [164], images were resized and the network was allowed to freely process the images.

Initial training settings for VGG-16 model were maintained for transfer learning as well. However, all the layers of the pre-trained model were fine tuned. No layer was fixed, hence the model was allowed the freedom to update parameter values for better performance similar to the methodology in [160]. In terms of training complexity, the model can be trained in under an hour with on 8 16GB GPUs. The dataset is small

[13] http://www.nvidia.com/object/deep-learning-system.html
[14] https://keras.io/
[15] https://www.tensorflow.org/

and fits into memory with out any loss in performance and training was stopped as the loss approaches zero in the final epoch.

Performance Evaluation

Log loss and accuracy metrics were used to evaluate the models. Multi-class logarithmic loss is shown in equation (4.1) below;

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} log(p_{ij}) \tag{4.1}$$

Where N and M represents the size of the sample and categories respectively, y_{ij} is the correct prediction of sample i being in category j, and p_{ij} is the estimated probability that the sample i belongs to the class j. Logarithm loss penalizes the accuracy of the classifiers on false positives. Probabilities were obtained as predictions from the soft-max layer in the networks. Table 4.2 below shows the log loss summary of the experiments conducted.

Table 4.2: *Log loss of models.*

Model	Original	Annotated	Original+Noise	Annotated+Noise
VGG-16	0.54	1.20	0.12	0.38
VGG-16(transfer)	18.45	27.88	0.10	30.61

The accuracy of a classifier is the ratio of number of correct prediction from sample to the total number of samples to be predicted. Accuracy is represented as follows.

$$accuracy = \frac{number\quad of\quad correct\quad predictions}{total\quad number\quad of\quad all\quad cases\quad to\quad be\quad predicted} * 100 \tag{4.2}$$

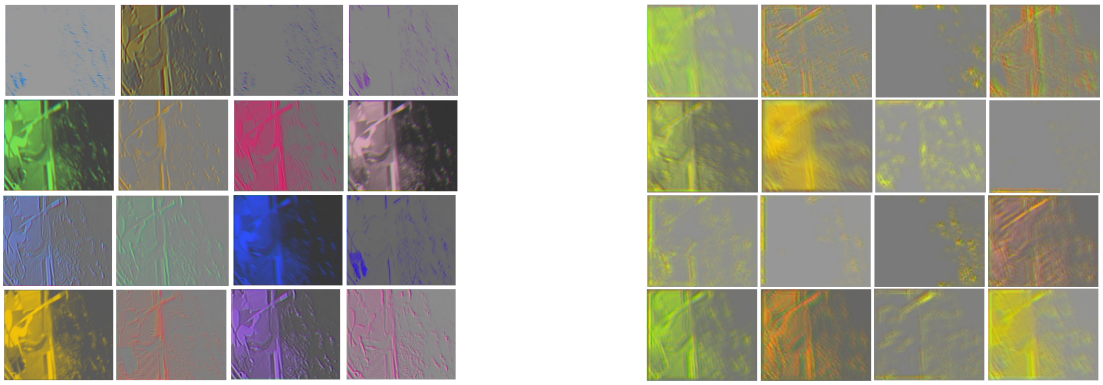
Table 4.3 shows test accuracy of models.

Table 4.3: *Test accuracy of models on different fish datasets.*

Model	Original	Annotated	Original+Noise	Annotated+Noise
VGG-16	97.20%	90.17%	99.38%	98.00%
VGG-16(transfer)	86.60%	79.81%	99.54%	77.80%

High accuracy of models was observed during testing on original dataset. This could be attributed partly to the fact that images were obtained from fishing boat cameras. In a still camera with 24 frames per second set up, not much difference exists between adjacent frames. Although the object view angles and illumination may vary. Again,

closely looking at the feature maps from the network layers revealed that prominent background objects also contributed to this. Learning was tuned towards these objects as against the fish instances. This can be seen clearly in the cross section of feature maps from the first and fourth convolutional layers in the Figure 4.5.

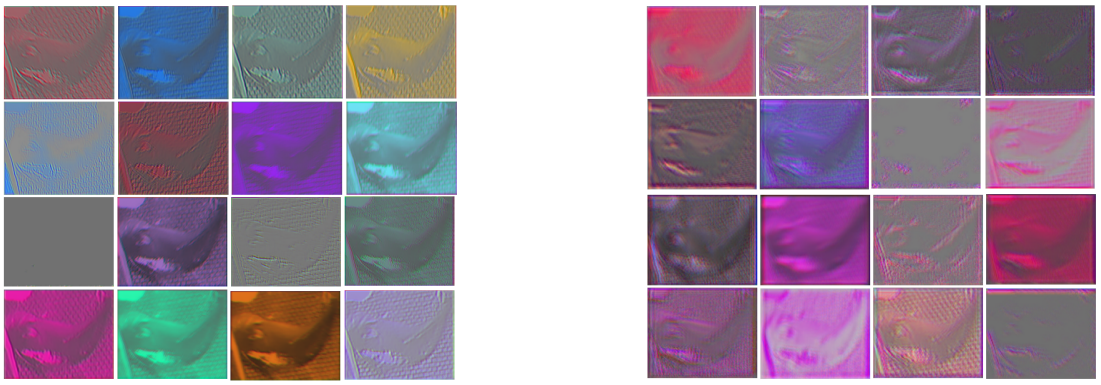


(a) Convolution Layer 1.

(b) Convolution Layer 4.

Figure 4.5: *Feature maps from original images.*

This effect became more obvious as we go deeper into the network. When training is done on these noisy images, it over-fits on stationary objects that re-appear in images. This adds to the high accuracies recorded. However, these effects were minimal in the experiment with annotated dataset. Fish instances dominate images and this suggests that learning is based on object of interest. Fish parts are visible through the feature maps even as we go deeper into the network. A cross section of feature maps from the first and fourth convolutional layers is shown in Figure 4.6.



(a) Convolution Layer 1.

(b) Convolution Layer 4.

Figure 4.6: *Feature Maps from annotated fish images.*

The imbalanced classes in the dataset also contributed to unexpected model results. We attempt to address this challenge through simple augmentation when generating more

data for the experiment. Hence, large margin between classes was checked to reduce the variance. Test results shows higher recall when more training data is available. Initial experiments with annotated data performed poorly than the original dataset but we observed increase in performance when more training examples become available. A summary of sensitivity analysis of VGG-16 model (untrained) on the datasets is shown in Table 4.4. Experiments on the new dataset showed significant increase in accuracy by both models but did not reduce the effects observed. However, transfer learning model log loss was far worse than expected. This could be associated with its strong confidence in false classifications. Another reason could be the variation between images used and the Imagenet images. Transfer learning works best when the two datasets are closely similar.

Table 4.4: *Summary of VGG-16 model performance.*

Dataset	Precision	Recall	F1-score
Original	0.82	0.82	0.81
Annotated	0.43	0.54	0.47
Original+Noise	0.92	0.91	0.91
Annotated+Noise	0.96	0.96	0.96

Furthermore, the results show that in noisy images, the network learns general features that are common to all objects in the images. We observed that features from these noisy-prominent objects become more dominant as we go deeper into the network. As such, they prevent the network from learning specific fish features required for category classification. With well-annotated images, the network learns deep features that are category specific and for correct classification; however, the size of the dataset and class-imbalance problem created more challenges. Transfer learning is an emerging area in deep learning that has established its presence in recent literature and has shown stringent results. But in this study, transfer learning from a pre-trained model on Imagenet was not effective. Learned features are transferable, but a closely related dataset could have produced better results. These results further solidifies that optimum performances are obtained when careful annotation of images is carried out. Manually annotating a massive dataset is challenging and automatic annotation requires labels agnostic techniques or other techniques such as segmentation and objectness approaches to achieve good results.

4.4 Conclusion

In this chapter, an experimental framework was developed to analyze the performance of deep learning models on a poorly annotated dataset. We investigated the performance of VGG-16 model on a fish classification challenge. Results obtained showed that the model performed below expectations on a poorly annotated data. Again, transfer learning and simple augmentation techniques were not effective in this particular domain. Two challenges in this scenario were identified. Firstly, the problem of the class-imbalance among the different fish categories. Training deep models on a biased dataset also contributed to the undesired performances. Secondly, the problem of lack of enough labeled examples. Deep models are not data efficient; hence, a huge amount of annotated instances are required to achieve good performance. This demonstrates the reliance of deep models on a carefully annotated dataset for the optimal solution to be obtained. The subsequent chapters propose alternative ways of handling the challenges highlighted.

Chapter 5

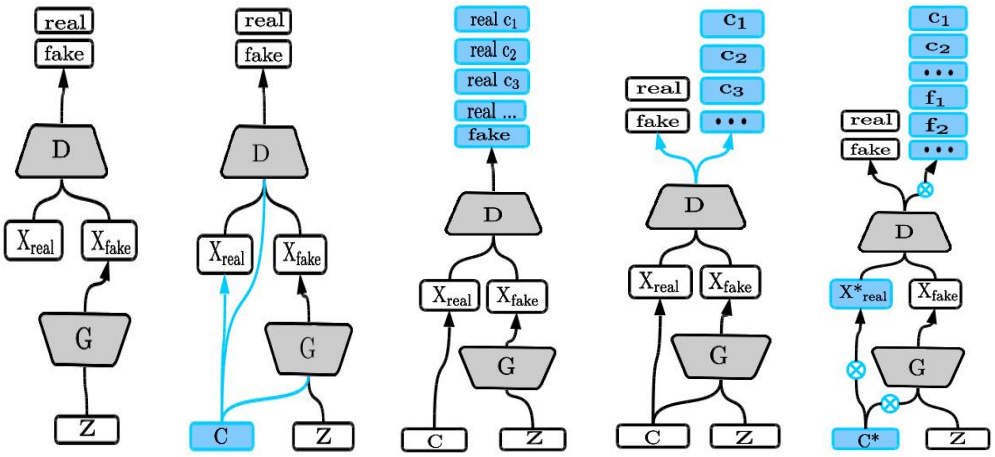
Few Shot Classifier GAN

In this chapter, we propose a new method to train a classification model using minimal number of labels. The chapter describes a novel GAN model called Few Shot Classifier GAN (FSC-GAN). FSC-GAN is capable of generating images while training a classifier alongside an adversarial objective. FSC-GAN extends label conditioning in GAN by training on multiple fake classes using both labeled and unlabeled data. The results appeared as part of the proceeding of International Joint Conference on Neural Networks (IJCNN 2018).

5.1 Overview

Image classification [190] is a challenging task requiring a large amount of labeled dataset to train accurate models at optimal performance. With the advent of deep learning technologies, there is a huge demand in obtaining massive labeled dataset [99, 76]. One major limitation is that massively annotating labels is a labour-intensive task [21]. Augmenting with synthetic data is an alternative strategy to bypass the unavailability of labeled training data. Unfortunately, such models trained only on synthesized data largely underperform.

In this chapter, we are interested in performing classification with limited labeled data because when only a small labeled samples can be acquired, unlabeled data could also be considered. Also, we are motivated to achieve co-generation and co-classification, in the sense that the generation will improve the classification and the classification will improve the generation cooperatively.



(a) GAN (b) C-GAN (c) S-GAN (d) AC-GAN (e) FSC-GAN

Figure 5.1: *Comparing existing GAN architectures with FSC-GAN model.*

Our work falls into the general problem domain of data labeling [153]. In particular, fine-grained classification [74] is an important problem with practical applications. Despite much recent progress, it remains a challenge to generalize classification and generation with lack of labeled data. Our key observation is that incorporating more fake classes plays an important role in training the GAN model at a finer-grained level, which may improve overall performance. In this chapter, we present a step towards fine-grained few-shot classification with the Generative Adversarial Networks. In contrast to the state-of-the-art, our GAN is more versatile and less restrictive in terms of input and output.

The core idea is to carefully fuse supervised and unsupervised learning via switchers within the connections of the GAN. Therefore, the GAN can be fed with labeled or unlabeled input data. Our proposed method classifies real samples into real classes and then isolate fake samples into their respective unknown fake classes. We solve this problem by leveraging fine-grained classification thanks to two mechanisms, *fake class embedding* and *multiple fake classes*. Drawing inspiration from AC-GAN [140] and semi-supervised GAN (SGAN) [138], our key idea is to associate classes with new samples by conditioning generation on class embedding. In contrast to previous work, our method seeks to classify real samples into predefined classes and further isolate fake samples into their respective fake classes, taking benefit of semi-supervised learning to improve the classifier accuracy.

In this work, the technical contribution is a novel GAN architecture taking as input labeled and unlabeled training data and performing fine-grained classification thanks to a multiple fake classes strategy. Our method is designed to handle image generation

losses and unconditional generation when unlabeled data are used during training. To the best of our knowledge, our model is the only one able to achieve fine-grained classification along image generation compared to other GANs in Table 5.1. We demonstrate the effectiveness of our system by evaluating our solution on publicly available datasets.

5.2 Why Few Shot Classifier GAN?

Automatic data labelling techniques have been proposed in [12] and [30]. The construction of a fully labeled dataset is supported by learning methods such as semi-supervised [92], one-shot [178] and active learning [39]. In particular, semi-supervised learning combines labeled with unlabeled data. Data augmentation [50] is an alternative strategy to bypass the absence of sufficient labeled training data by transforming original samples. Finally, data synthesis generates artificial data by training models exclusively on synthesized data [25, 162]. Naturally, the intuitive zero-sum game principle of GANs is an appealing strategy for data labeling.

Table 5.1 outlines the properties of various GANs related to this work. The first Vanilla GAN [63] introduces the Kullback-Leibler divergence as a distance-based distribution similarity to produce highly-detailed images.

Model	Supervised	Unsupervised	Few labels	Multi fake classes
`Vanilla GAN` [63]	✗	✓	✗	✗
`S-GAN` [138]	✓	✓	✗	✗
`AC-GAN` [140]	✓	✓	✗	✗
`C-GAN` [126]	✓	✗	✗	✗
`CatGAN` [169]	✓	✓	✓	✗
`CC-GAN` [46]	✓	✓	✓	✗
`SS-GAN` [170]	✓	✓	✓	✗
`TAC-GAN` [41]	✓	✓	✗	✗
`Few-shot GAN`	✓	✓	✓	✓

Table 5.1: *Comparing the properties of few-shot GAN with existing GAN models.*

GAN has a lot of applications targeted for images processing, such as image data augmentation [137], high-resolution image generation [89], image reconstruction [156], text-2-image generation [150], natural image generation [47]. Moreover, various GAN architectures have been proposed for classification [140], semi-supervised labeling [138] and other domains [48, 47].

Nevertheless, tailoring GANs for classification is a tedious task [40]. Vanilla GAN [63] is an unsupervised adversarial model that allows only a scalar output representing if a

sample is real/fake. Therefore, no classification can be performed by the discriminator of Vanilla GAN since this model only accepts unlabeled data as input. Our method differs by performing generation in conjunction with classification. However, Conditional GAN [126] generates data conditioned on class labels via label embeddings in both discriminator and generator. Similar to Categorical GAN [169] (CatGAN), our method integrates a classification loss function to learn a classifier from unlabeled or partially labeled data.

Conditioning on labels brings to light the possibility of semi-supervised classification using GANs by forcing the discriminator network to output class labels. In semi-supervised GANs [138] (SGAN), the training is realized by combining a single fake class with known classes. This additional fake class is required to categorize samples from the generator. In our approach, we combine conditioning and embedding to cope with the well-known limitation of semi-supervised GANs, namely being unable to handle unlabeled data. Auxiliary Classifier GAN (AC-GAN) [140] is also conditioned on the class labels to generate visually plausible images. Our work is close to the AC-GAN in the sense that we exploit label conditioning. However, our *few-shot classifier GAN* is not restricted to outputting a single class label for every sample.

Contrary to AC-GAN and CGAN that only rely on full labeled datasets, our model can perform conditional generation for labeled data and unconditional generation of unlabeled data. The model is adapted to output class labels even for fake images. Our model is also auxiliary because we output a numeric value deciding if the image is real or fake and further classify them into multiple fake/real classes. Compared to all GANs in Table 5.1, our model is the only model able to perform fine-grained classification along image generation so far, even if the expected class is not provided for training. We leverage this fine-grained property by injecting multiple fake classes with embedding. Moreover, the key difference of our approach against AC-GAN and SGAN is that our classification is not limited to real classes. Our approach combines supervised and unsupervised learning to handle both unlabeled and labeled data.

5.3 Formulating FSC-GAN

5.3.1 Fake Class Encoding

"Real" refers to label or images provided as part of the training set, while "fake" refers to generated label or images. The set of real labels $\mathcal{C} = \{0, \cdots, N-1\}$ for the N classes are extracted from the training data (indexed from 0 to 9 for digits). For each

class c in the training data, a corresponding fake class label c^+ is added automatically as described in the following procedure. The index of each digit is converted into a one-hot encoding vector. Then, we generate a set of fake class labels $\mathcal{C}^+$ by accommodating a longer vector representation. The one-hot encoding of the real classes is padded with $|\mathcal{C}|$ zeros shifted to the right. From this one-hot representation, the corresponding fake class is generated by padding zeros at the left of the real label. For example, if the real label 0 is encoded over $|\mathcal{C}|$ bits as `1000000000`, we now represent this class by `10000000000000000000`. Then, the corresponding fake label is `00000000001000000000`. The resulting set of all labels is denoted $\mathcal{C}^* = \mathcal{C} \cup \mathcal{C}^+$.

5.3.2 FSC-GAN Objective

Our *Few-shot Classifier GAN* consists of two convolutional neural networks competing against each other: a generator model G and a discriminator model D, where the discriminator tries to classify real objects and objects synthesized by the generator, and the generator attempts to confuse the discriminator. This model is designed to classify real and fake samples. This optimization problem requires a min-max solution obtained by solving the overall functional:

$$\min_{D} \quad \max_{G} \quad V_{fshot}\left(D, G\right) \tag{5.1}$$

where D and G mimic a two-players min-max game with value function $V_{fshot}\left(D, G\right)$. Then, the optimal solution is reached when both models can not make a significant gain over its opponent.

Similar to classical AC-GAN, the generator G takes as input a random noise vector $\mathbf{z} \in \mathbb{R}^d$ where d is the vector size and $\mathbf{c}$ is a label when the corresponding class is available. In the absence of class labels in the training set, G takes only $\mathbf{z}$ as input. G is trained to be an image producer aiming to generate sampled images expected to lie within the distribution of the training data. The classifier is incorporated within the discriminator model D to produce better samples. Then, D is trained to discriminate between image generated by G against real training images.

The value function $V_{fshot}\left(D, G\right)$ is defined as a piecewise function acting as a *network switcher*, as follows:

$$V_{fshot}(D, G) = \begin{cases} \tilde{\mathcal{C}^*} = \{\emptyset\} \ : \ V_{gan}\left(D, G\right) \\ \tilde{\mathcal{C}^*} \neq \{\emptyset\} \ : \ V_{acgan}\left(D, G\right) \end{cases} \tag{5.2}$$

(a) Discriminator (b) Generator

Figure 5.2: *The Few-shot Classifier GAN generated images by transpose convolution to avoid up-sample resizing. Both diagrams show the arrangement of layers for the architecture of the Discriminator and the Generator. The discriminator produces two outputs: a classifier output determining the class, and an output determining the type of image (real or fake).*

where $\tilde{\mathcal{C}}^*$ is the class labels set involved in the current batch and $V_{gan}(D,G)$ is the expected value of the unconditioned probabilities over D and G. Alternatively, $V_{acgan}(D,G)$ is the expected value of the conditioned probabilities over D and G depending on labels for classification. The network switcher V_{fshot} enforces the GAN model to perform unconditional discrimination (V_{gan}) in the absence of labels and to perform conditional discrimination (V_{acgan}) when labels are available.

$$V_{gan}(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{5.3}$$

where the prior distribution is denoted by $p_{\mathbf{z}}$ and $\mathbf{z}$ is the set of prior noise drawn from a uniform distribution. The generator samples the latent representation variable $\mathbf{z}$ only to generate images. Subsequently, V_{acgan} is activated when G and D are conditioned with the class labels set $\mathcal{C}^*$ during training.

$$V_{acgan}(D,G) = \mathcal{L}_{\mathrm{s}} + \mathcal{L}_{\mathrm{c}} \tag{5.4}$$

where $\mathcal{L}_{\mathrm{s}}$ and $\mathcal{L}_{\mathrm{c}}$ respectively denotes the log-likelihood of the expected sampling and classification. Formally, the sampling loss $\mathcal{L}_{\mathrm{s}}$ and the classification loss $\mathcal{L}_{\mathrm{c}}$ write as:

$$\mathcal{L}_{\mathrm{s}} = \mathbb{E}[\log P(S = real|X_{real})] + \mathbb{E}[\log P(S = fake|X_{fake})] \tag{5.5}$$

$$\mathcal{L}_{\mathrm{c}} = \mathbb{E}[\log P(C = c|X_{real})] + \mathbb{E}[\log P(C = c|X_{fake})] \tag{5.6}$$

The discriminator $D(X) = (P(S|X), P(C|X))$ isolates fake versus real samples and then performs the classification of all samples whether real or fake. $P(S|X)$ is a probability distribution over samples and $P(C|X)$ is probability distribution over labels. $X_{fake} = G(c, z)$ is a batch of generated images and X_{real} is a batch of real images used to train the discriminator D. In the presence of labels, sampling is conditioned on class labels C and the model considers both sampling and classification losses. Our GAN has two sets of outputs: a scalar determining if the image is real or fake and a set of discrete values representing the labels corresponding to real or fake samples.

5.3.3 Network Switcher

We inject the *network switcher* inside our deep neural architecture to manage multiple learning strategies by forcing the learning to switch to the desired mode for the training. This solution is better than trivially switching between two different models (namely, AC-GAN and GAN) by avoiding duplication of generators and discriminators. In particular, this binary switcher creates an algorithm branch within the computational graph to switch to supervised or unsupervised training. This network switcher is expressed as an exclusive OR operator (XOR) ensuring that the learning strategy fits the nature of the given batch. As shown in the Figure 5.1, the switcher is depicted using the $\otimes$ operator.

5.3.4 G and D Black Boxes

Our *few-shot classifier GAN* is a deep convolutional GANs to produce better visual quality samples. The Generator and Discriminator are both expressed as deep convolutional neural networks with a fixed number of layers, leaky Relu[1] activation functions and hyper-parameters. We tune the hyper-parameters and the number of layers to fit the desired image resolution. Figure 5.2 depicts the inner architecture of the generator and the discriminator.

In the generator G, we use a series of transpose convolutions with varying strides to upsample images at the desired resolution. The first two layers of the generator are fully connected with no in-between batch normalization. The outputs of the second layer are reshaped into an 7×7 image with 128 channels. The third layer is a transpose convolution using a single stride and outputs a 7×7 image with 256 channels. The forth layer is a transpose convolution with a stride of 2 outputting a 14×14 image

[1] https://keras.io/layers/advanced-activations/

with 128 channels. Finally, the final layer uses a transpose convolution outputting an 28×28 image with a single channel.

The discriminator D is a conventional CNN down-sampling image batches into a feature vector representation suitable for classification. The discriminator is composed of four convolution layers with strides of 2 in each layer. We use batch normalization between layers to accelerate the convergence, excepting in the final layer. The subsequent layers are two parallel linear layers: a classification output and a GAN output. The classification layer returns logits while the GAN layer returns the sigmoid activated output (fake or real).

5.3.5 Dual Training

The training procedure is summarized in the provided pseudo-code (Algorithm 1). The procedure described in Algorithm 1 takes as input data and label batches. Each batch is tailored with a given ratio of labeled to unlabeled data. The training process is performed by alternating between supervised and unsupervised training since the number of labeled samples may differ in training.

In Algorithm 1 the number of epochs is $e = 500$. The inner loops among labeled and unlabeled samples are balanced to guarantee the stability of the loss function. The first loop (k steps) iterates over labeled data only by performing stochastic gradient descent over the discriminator and generator via the discriminator. This loop evaluates the sampling and classification losses functions (line 6). The overall loss is updated at the end of each iteration within the inner loop. Similarly, the second loop (j steps) iterates over unlabeled data, but only sampling loss is evaluated before updating the overall loss. The number of iterations k and j depends proportionally on the ratio of labeled and unlabeled data to produce an importance-based sampling. However, if the training dataset is balanced then k equals j.

When training with labeled data, the discriminator D is trained to maximize $\mathcal{L}_{\mathrm{s}} + \mathcal{L}_{\mathrm{c}}$ while the generator G is trained to maximize the entropy between $\mathcal{L}_{\mathrm{s}}$ and $\mathcal{L}_{\mathrm{c}}$. Our discriminator is trained with image batches from G and the original training data. When labels are not available both D and G are trained using V_{gan}.

5.4 Learning from Fewer Examples

An extensive experimental analysis is conducted to evaluate the accuracy of the proposed model with multiple fake classes. The proposed GAN architecture is used to

Algorithm 1 Training Algorithm

```
1: procedure TRAIN(data_batches, label_batches)
2:     for e epochs do
3:         for k steps do
4:             Fetch next labeled mini batches
5:             Perform Stochastic Gradient Descent on D
6:             Perform Stochastic Gradient Descent on G
7:             Evaluate(L_s, L_c)
8:             Update D and G losses
9:         end for
10:        for j steps do
11:            Fetch next unlabeled data mini batches
12:            Perform Stochastic Gradient Descent on D
13:            Perform Stochastic Gradient Descent on G
14:            Evaluate(V_gan)
15:            Update D and G losses
16:        end for
17:    end for
18: end procedure
```

perform all experiments in which the ratio of unlabeled and labeled samples is varied during the training process. Experiments were ran in a `NVIDIA DGX-1` supercomputer with multiple GPUs using the `TensorFlow` framework. Finally, performances of the proposed model are reported in terms of accuracy for a variety of training configurations.

5.4.1 Dataset Tuning

We run our experiments over two state-of-the art datasets of images, namely the MNIST dataset [105] and the SVHN dataset [134]. These datasets were selected because no pre-processing is required. Unlabeled data are derived from the datasets by neglecting provided labels.

The MNIST dataset is large database composed of a train set (60000 images) and a test set (10000 images) with size-normalized, centered, fixed-size and single-channel images representing handwritten digits. Each digit has its corresponding label. In our experiments, the train and the validation sets are fused to create a new training set to evaluate our GAN. Using this dataset, our experiments based on varying the number of unlabeled samples start by considering all labels from the training set. Then, the number of labeled samples is decreased by 10k at each run until the lower bound of 50k unlabeled and 10k labeled samples is reached.

The Street View House Numbers (SVHN) dataset is significantly harder and more challenging. The SVHN is a real-world dataset (73k train set and a 26k test set) composed of three-channels noisy images of house numbers obtained from Google Street View. The class distribution in the training set varies between 5k to 13k instance per class. Using this dataset, our experiments start with 73k fully-labeled samples and the testing is performed only on 10k randomly-selected samples from the test set. The unlabeled set is enriched with 10k samples selected from the train set at each pass until the lower bound of 60k unlabeled samples is reached.

5.4.2 Setup and Parameters

We have implemented our approach using `TensorFlow` [1]. We use 10 million trainable parameters for MNIST and 30 million trainable parameters for SVHN. We bypass the imbalanced data problem by collecting an equal number of labeled samples from each class when designing our training dataset for our labeled-to-unlabeled experiments. Even if the size of labeled samples set decreases, it is worth noting that the size of the training set remains unmodified along the experiments. However, the training set is extended with fake classes for multiple fake class experiments. During the testing phase, the learned classifier is evaluated over 10k samples from hold out real test samples.

A batch size of 32 is used for all datasets and all experiments. We normalize all input before the training. We use the classical Adam Optimizer [91] with a learning rate of 10^{-3} for the gradient descent optimization of the generator and the discriminator. Also, we consider a prior vector of 100 dimensions from the uniform distribution. Since grid search is computationally expensive with these hyper-parameters, random search was used to determine the appropriate values.

5.4.3 Performance Evaluation

We evaluated the accuracy of the output obtained by our learned classifier with multiple fake classes in comparison with the output produced by our learned classifier with only a single fake. The evaluation metric we employ to measure the accuracy of the trained classifier is defined as the total number of correctly classified test samples divided by the overall number of test samples.

We reported the quantitative results for this accuracy in Table 5.2 and 5.3 for experiments conducted over 10k test samples by varying number of class labels during the training. Sample images are collected at the end of the overall training. We depict generated images from training on MNIST and SVHN with 50k and 60k unlabeled data

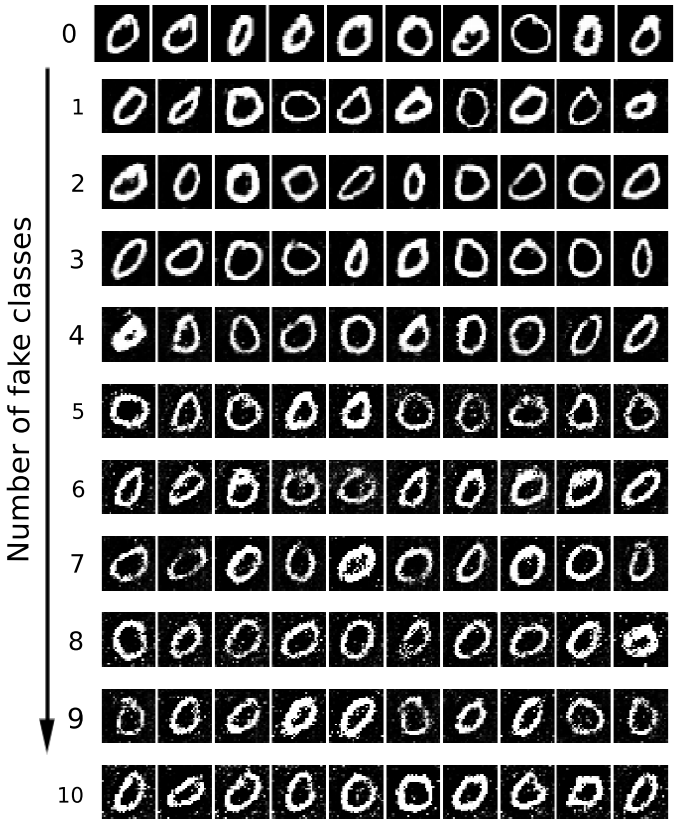
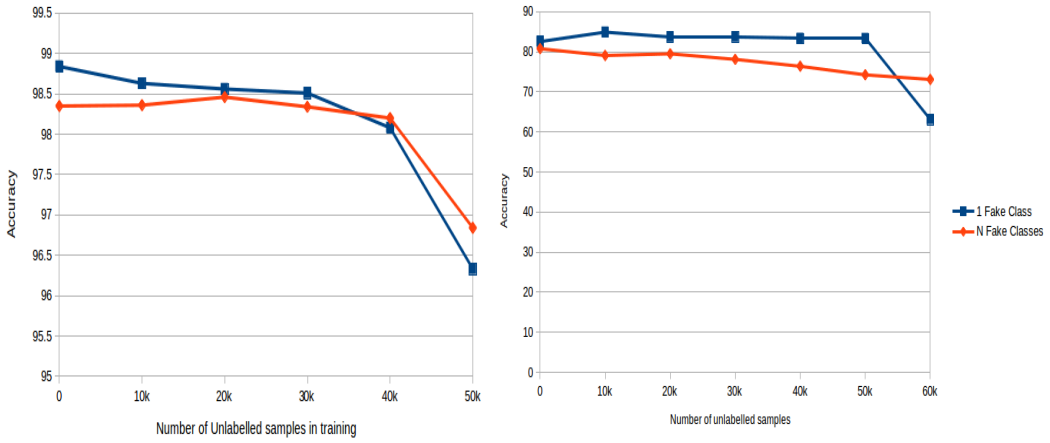


Figure 5.3: *For the digit 0, we display the output of our proposed GAN without fake classes (first row). The second row represents the output obtained while considering a single fake. Then, we increase the number of fake classes for all other following rows (top to bottom). Visual results show that the pixel corruptions grow proportionally when multiple fake classes are considered during the training.*



Accuracy for MNIST Accuracy for SVHN

Figure 5.4: *We plot the accuracy of our Few-shot Classifier GAN as a function of the number of unlabeled samples. For this test, we trained our GAN with semi-supervised learning over the MNIST and SVHN datasets. The multiple fake classes mode outperforms the single fake class mode in presence of 70% of the data are unlabeled.*

respectively in Figure 5.5. The performance of our *Few-shot CGAN* is summarized in Figure 5.4.

Table 5.2: *We report the precision accuracy of semi-supervised learning applied on the MNIST dataset with different configurations of fake classes on 10k hold out samples.*

Unlabeled Samples	Single Fake Class	Multiple Fake classes
0	98.84	98.35
10k	98.63	98.36
20k	98.56	98.46
30k	98.51	98.34
40k	98.08	98.20
50k	96.33	96.84

Further experiments were conducted by varying the number of fake classes from 0 to N (where N is the total number of real classes) to examine how fake classes affect the generation of images. For this experiment, the GAN is re-trained from scratch using biased sampling. Figure 5.3 shows the effect on the quality of image generation when the number of fake classes increases. Samples are collected when the training is completed.

5.5 Discussion

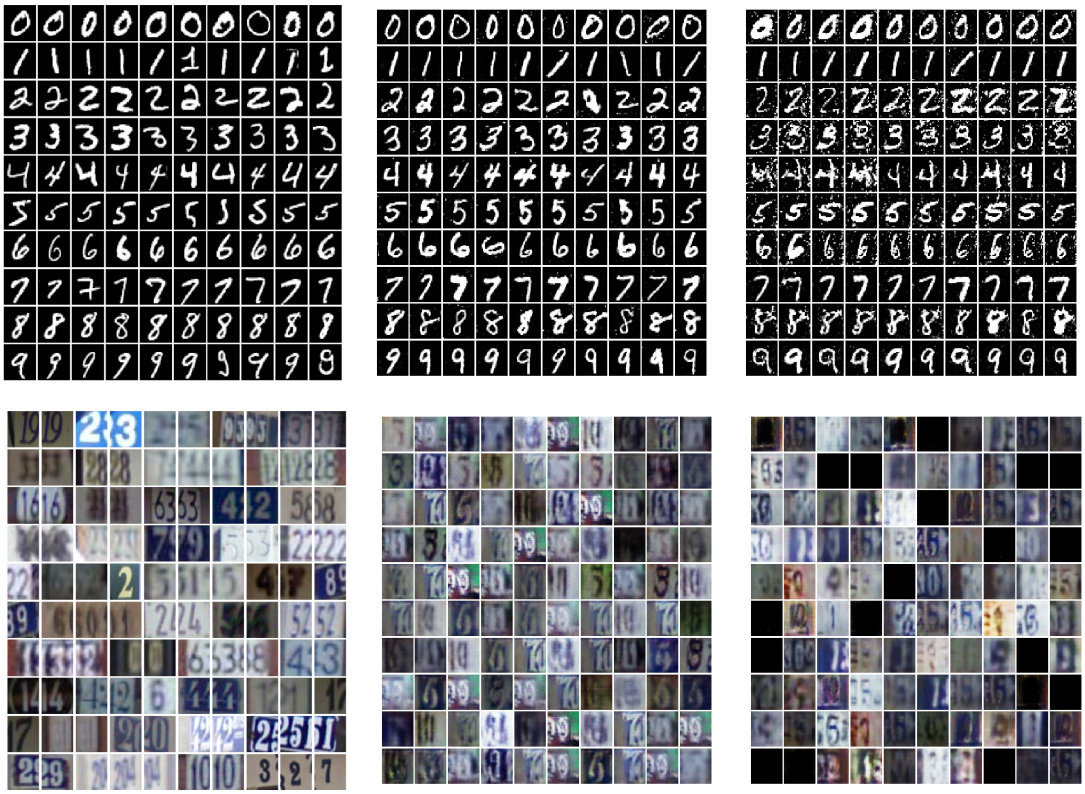
We observe that the accuracy drops when the number of labeled samples decreases in training for both SVHN and MNIST. A wider margin is observed during experiments with the SVHN dataset because of the challenging characteristics of this specific dataset. For both datasets, our *Few-shot GAN* outperforms the classification process in *multiple fake classes* mode with the presence of fully labeled data. Also, our *Few-shot GAN* performs better in *multiple fake classes* mode than in *single fake class* mode, when 70% of the data are unlabeled. Our *Few-shot GAN* performs significantly better on SVHN dataset (by a factor 10) when 60k unlabeled samples are used.

Table 5.3: *We report the precision accuracy of semi-supervised learning applied on the SVHN dataset with different configurations of fake classes on 10k hold out samples.*

Unlabeled Samples	Single Fake Class	Multiple Fake classes	P-values
0	82.55	80.76	0.989
10k	84.90	79.07	0.9596
20k	83.67	79.49	0.9661
30k	83.68	78.11	0.9455
40k	83.34	76.38	0.9137
50K	83.30	74.25	0.8482
60k	63.11	73.10	0.3936

Unfortunately, generated samples exhibit visual artifacts when our GAN is used in the multiple fake class mode. Visual qualitative results show that the quantity of visual artifacts grows proportionally when multiple fake classes are considered during the training. Moreover, we notice the apparition of artifacts when unlabeled samples become dominant over labeled samples and when the GAN relies less on the classification loss. Better performances are also observed when the classifier is trained on not too good and not too poor samples as demonstrated by the model performances. That is, despite the fact that the model did not generate exceptionally good data, the classifier was able to learn from the poor fake samples. This finding is similar to the conclusions in [40]. Finally, we observe that bias sampling does not significantly improve the quality of generated samples. We suggest to devise a deeper architecture or training with more epochs to solve this problem.

Again, the p-values presented does not suggest any statistical significance between single and multiple fake classes. Since the p−values presented are larger than 0.05 therefore, we fail to reject the null hypothesis. This implies that the two versions of FSC-GAN are not significantly different based on the p-values. That said, the p-values only gives



Original sample | Generated sample (single fake class) | Generated sample (multiple fake classes)

Figure 5.5: *The first row contains MNIST samples and the second row contains SVHN samples. On the left-hand side, we display real image samples from the training data, samples from training with a single fake class are displayed in the middle. Finally, we display generated sample images when trained with multiple fake classes on the right-hand side. Generated samples are obtained from 50k and 60k unlabeled training data on MNIST and SVHN respectively.*

information about the learned classifier and does not reflect the performance of the generator or the discriminator model in this context. Thus, p-values do not inform the discussion around whether the correct distribution was learned or whether the model generated good quality samples which are the key aspects of generative modelling. Moreover, the target of the experiment is to show that FSC-GAN set-up is stable and can be used to learn a good classifier model in the absence of enough labels.

Acquiring labels at scale is currently done manually. In few-shot context where we lack labeled training data, classifying images and labelling data is still a tough problem. In this work, we focused on the design of a novel adversarial architecture incorporating latent label embedding, network switchers and multiple fake classes to to train a classifier. One of the greatest appeals of our approach is its label-agnostic property. Also, FSC-GAN supports a wide range of strategies from fully supervised, semi-supervised

to weakly supervised learning that was not possible with other alternative GAN previously.

In contrast to other fine-grained classification techniques, our method takes advantage of the generator to trick the discriminator into classifying generated data along with their labels whatever the input is. We leverage this property by exploiting the continuum of the known labelling space. One of the central differences is that we do not learn how to represent real labeled data only but also how to learn powerful representations from unlabeled data. The most important aspect of our FSC-GAN is its capability to output unknown sub-categories (namely the fake classes).

As a result, our proposed GAN is a useful tool to learn a stable classification in the presence of few labeled examples mixed with a significant amount of unlabeled examples. An important advantage of our method is that it can switch between full supervised learning and semi-supervised learning thanks to the network switchers. Moreover, our evaluation confirms that discriminated samples improve the overall accuracy when the dataset lacks labeled samples. In most cases, our work suggests that the proposed approach performs similarly to traditional GAN in the presence of a sufficient amount labels and provides better results in the absence of labeled samples during the training phase. The main limitation of our technique is that the generated sample quality could be affected when multiple fake classes are used.

5.6 Conclusions

In conclusion, this chapter proposed a new GAN model (FSC-GAN) and a new method to train supervised GANs with small labeled training data. FSC-GAN is able to generate images and train a deep classification model alongside. The model uses multiple fake classes, network switchers and fine-grain classification idea to achieve the task. This model provides valuable insights into the fine-grained classification problem, and opens a new horizon to perform deep learning with less amount of data. That said, FSC-GAN suffers from three major limitations.

The first challenge is the quality of the generated images. The images produced with FSC-GAN have artifact and may not be usable as an alternatives source of labeled training data for other processes such as data augmentation. For these to be achieved, there is neeed for FSC-GAN to generate better quality samples that will match the real samples in the training data. An important advancement toward this direction would be a new family of fake loss functions optimized for human visual perception. That is the genrated samples should be visually indistinguishable from the real data by the

naked eye.

Secondly, the model takes prolonged epochs to converge. The model proposed needed to be trained for a long periods before it starts producing good samples even on small datasets like MNIST. Long training time creates complexity in traing and a scaling problem when the dataset becomes large. This may require widening the hyper-parameters search and tuning the model further using other optimization techniques not explored in this chapter.

Thirdly, the model requires a carefully curated training data with fairly balanced classes to obtain the required results. Experiments that combine labeled and unlabeled data used ony balanced labeled classes. However, real-world problems data are not always balanced. Such scenarios could be potential pitfalls for this model. These challenges identified are addressed in the succeeding chapter.

Chapter 6

Image Generation and Classification in Class-Imbalanced Datasets

In this Chapter, the problem of class-imbalanced data in classification is addressed using a new GAN model called Multiple Fake Classes GAN (MFC-GAN). We consider a multi-classification problem where MFC-GAN is applied as an augmentation technique to synthesize more samples from minority class instances. MFC-GAN is an extension of FSC-GAN introduced in Chapter 5. MFC-GAN provides a better quality image, faster convergence time while inheriting all the desirable features of FSC-GAN. The findings appeared in NeuroComputing Journal (2019).

6.1 Class-Imbalance Problem

The class-imbalance problem arises when the samples in a dataset are dominated by one class usually the negative class. It is common across different domains such as security, banking and medicine. This could occur in a binary classification or a multi-classification task [51]. Models trained on a class-imbalanced data tend to be biased towards the majority class. Existing approaches address this problem either at the data level or the algorithm level [97]. Data re-sampling techniques such as undersampling and oversampling are applied at data level to ensure equal representation of instances amongst classes. Algorithmic solutions include modifying the learning objective to ensure equal participation of all classes during training.

Data augmentation is a common technique employed to synthesize more training data. Artificial variations are useful in minimizing any bias in data collection and class-imbalance problem. For instance, in image domain, augmentation techniques used could range from simple image flips [6], random crops [99], noise [6] distortions to more advanced techniques like PCA colour augmentation [99] and image-pairing [84]. A data augmentation technique can be a source of more training data [57] or a regularizer [84] thereby improving generalization. These techniques have proved to be effective in learning from class-imbalanced datasets. However, in extreme class-imbalance cases, applying augmentation to few samples may not provide the required variations to produce distinct samples to re-balance the dataset. Furthermore, the problem becomes compounded in a multi-class problem as the performance of a class may be affected while trying to improve another [56]. Besides, existing techniques may not necessarily be useful in deep learning [49].

More recently, Generative Adversarial Networks (GAN) have been used to generate images with high visual fidelity [89]. Researchers have shown that these images can be used as extra training data to support other processes such as classification [57, 204]. A GAN model produces quality samples with the required variations similar to the training data. Different GAN models have been proposed for data augmentation in previous works [51, 9, 122, 57, 15]. Also, GAN was used to tackle imbalanced data in a binary classification problem using non-image data in [51] and used by Antoniou *et al.* [9] as an augmentation approach to improve image recognition accuracy. Our approach shares some similarities with these researches but differs in the sense that we use a different GAN model in image classification domain. Moreover, we are interested in performing multiple classification with an imbalanced training data. With scarce minority classes, image generation can be challenging because a useful augmentation sample needs to be plausible, diverse and from the required minority class [122, 9].

In summary, resampling methods do not perform well in hugely imbalanced datasets. Traditional data augmentation methods are still widely used. However, these are limited and often don not generate enough data variance, especially in extreme cases. GAN-based methods provide a more realistic solution to generate data samples and handle class-imbalance (i.e. a multi-modal [9, 122], image-translation [204]). Unlike these methods, our model is simpler to train and generates specific-class samples even in extreme cases.

6.2 Multiple Fake Classes GAN

In this section, we propose Multiple Fake Classes Generative Adversarial Network (MFC-GAN). MFC-GAN preserves the structure of the minority classes by learning the correct data distribution and produce unique images whenever it is sampled. We demonstrate the usefulness of MFC-GAN by addressing class-imbalance problem in a multi-classification task. MFC-GAN differs from other GAN models that implement a classifier alongside the discriminator such as S-GAN [138], AC-GAN [140] and similar frameworks in the sense that we use a multi-fake class GAN model. Multiple fake class feature was implemented in Few-Shot Classifier GAN (FSC-GAN) in Chapter 5 to generate samples and perform classification. Incorporating more fake classes in the FSC-GAN resulted in artifacts appearing in generated samples which may hinder using such samples as candidates for augmentation. MFC-GAN model extends FSC-GAN idea and demonstrates that artifacts can be reduced significantly by conditioning image generation on real class labels only and modifying the classification objective. Thus, fake class labels are only employed when classifying generated images.

Incorporating more fake classes in this context stabilizes training early and generates plausible samples with fewer epochs. Our argument is that since both minority and majority classes come from the same distribution, these classes share some common features. Hence, features learned from majority classes should aid in learning the minority classes. Consequently, class conditioned generation will focus the model into sampling minority classes. Our approach trains MFC-GAN on the imbalanced dataset then generate and augment synthetic minority class instances to the original training data. A Convolutional Neural Network (CNN) is then trained on the augmented dataset. We evaluated our approach using four imbalanced datasets namely; E-MNIST [38] and created artificial imbalance in MNIST [104], SVHN [134] and CIFAR-10 [98] by reducing the number of samples in specific classes. Significant performance gain was obtained when MFC-GAN was used as an augmentation model when compared to the baseline (CNN classification without augmentation) and other common and state-of-the-art methods (SMOTE [28] and AC-GAN [140]).

Labels are encoded in MFC-GAN in a similar way to FSC-GAN; hence, labels are prepared using the label encoding described in Chapter 5. To generate class specific samples, we conditioned MFC-GAN generator using real labels only. Label conditioning encourages the generator to work towards producing realistic samples and controls the generation of class-specific samples [138]. When training MFC-GAN, we classify real images into real classes and generated images into different fake classes. MFC-GAN is trained with a modified AC-GAN objective. The objective maximises the log-likelihood

of classifying real samples into real classes C and fake samples into fake classes C' as shown in Equations 6.1, 6.2 and 6.3.

$$\mathcal{L}_{\mathrm{s}} = \mathbb{E}[\log P\left(S = real|X_{real}\right)] + \mathbb{E}[\log P\left(S = fake|X_{fake}\right)] \tag{6.1}$$

$$\mathcal{L}_{\mathrm{cd}} = \mathbb{E}[\log P\left(C = c|X_{real}\right)] + \mathbb{E}[\log P\left(C' = c'|X_{fake}\right)] \tag{6.2}$$

$$\mathcal{L}_{\mathrm{cg}} = \mathbb{E}[\log P\left(C = c|X_{real}\right)] + \mathbb{E}[\log P\left(C = c|X_{fake}\right)] \tag{6.3}$$

Where $\mathcal{L}_{\mathrm{s}}$ is used to estimate the sampling loss, which represents the probability of the sample being real or fake. $\mathcal{L}_{\mathrm{cd}}$ and $\mathcal{L}_{\mathrm{cg}}$ are used to estimate the classification losses over the generator and the discriminator. X_{real} represents the training data and X_{fake} is the set of generated images.

As can be seen in Equation 6.2 and 6.3, MFC-GAN classification objective differs from what was implemented in AC-GAN and FSC-GAN. Both FSC-GAN and MFC-GAN discriminators classify generated samples into different fake classes. This prevents classifying unrealistic samples into real classes by providing fine-grained training to the model. However, MFC-GAN differs from FSC-GAN in the way the loss function of the generator is defined as can be seen in Equation 6.3. In other words, in our model, the generator is penalized according to how far the generated sample is from the real class label. Notice, that in the FSC-GAN model, the generator model is penalised according to how far the generated sample is from fake class label. By having this key difference in our model, we ensure that poor generated samples guarantee higher loss, which is not necessarily the case in the FSC-GAN settings. This has also promoted early convergence of the model where MFC-GAN model proved to be able to generate plausible samples with far fewer epochs than both AC-GAN and FSC-GAN.

Furthermore, for every iteration, Equation 6.2 means that the discriminator classifies samples as real or fake with the associated class (i.e., real class 1 or fake class 1) while Equation 6.3 means that with every generator iteration, it tries to classify fake samples as real classes. As the generator performance improves, only subtle differences exist between the two sets of images (fake, real) and this acts as a regularizer that penalizes the discriminator as the model approaches optimal performance. Similar to FSC-GAN, MFC-GAN is also capable of handling labeled and unlabeled data in training. Depending on the availability of labels, the network switcher feature [3] enables both models to alternate between two training modes. This switcher is a piece-wise function that

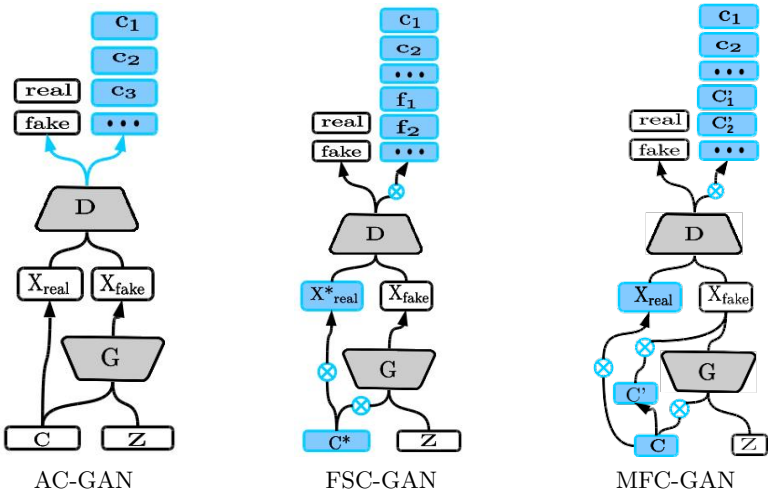


Figure 6.1: *Comparing MFC-GAN architecture with AC-GAN and FSC-GAN models. C is a set of labels, z is a random noise vector, G is the generator, D is the discriminator, real and fake are GAN outputs representing the probability of an image being real or fake, $c_1, ..c_n$ are the set of real classes, $f_1, ..f_n$ and $c'_1, ..c'_n$ are sets of fake classes, X_{real} is the original training images, X_{fake} is the set of generated images and $\otimes$ is the network switcher feature that alternates between labeled and unlabeled training.*

oscillates between supervised and unsupervised training. Although, there is a slight difference in the way classification loss is evaluated (as shown in Equation 6.2). Figure 6.1 compares the structure of MFC-GAN to FSC-GAN and AC-GAN. With labeled data, the MFC-GAN discriminator is trained to maximise the sum of $\mathcal{L}_{\text{s}}$ and $\mathcal{L}_{\text{cd}}$ while the generator is trained to maximise the difference between $\mathcal{L}_{\text{s}}$ and $\mathcal{L}_{\text{cg}}$. In this setup, the MFC-GAN generator is sampled using a noise vector conditioned on real class labels. In the absence of labels, MFC-GAN is trained using $\mathcal{L}_{\text{s}}$ only and behaves like a vanilla GAN model as shown in Equation 6.4. In the latter case, the generator is sampled using a noise vector only. Although, in these experiments, this feature was not exploited. Further comparisons and discussions around there differences can be found in Section 6.4 and Figure 6.2.

$$V(D,G) = \begin{cases} \mathcal{C} = \{\emptyset\} \ : \ \mathcal{L}_{\text{s}} \\ \mathcal{C} \neq \{\emptyset\} \ : \ \mathcal{L}_{\text{s}} \pm \mathcal{L}_{\text{c}} \end{cases} \tag{6.4}$$

6.3 Learning from Imbalanced Data

The architecture of both the discriminator and generator used on MNIST and E-MNIST were adopted from FSC-GAN, as detailed in Chapter 5. Regarding SVHN and CIFAR-10 experiments, we used the same architecture as in the original AC-GAN model [140], and added spectral weight normalization [128] in both generator and discriminator for both AC-GAN, FSC-GAN and MFC-GAN. This was to ensure a fair comparison.

In order to evaluate the performance of our method, we compared it with AC-GAN [140] which is one of the best supervised generative models. We also compared our method with Synthetic Minority Oversampling Technique (SMOTE) [28] which is one of the most common methods for generating data to handle class-imbalanced datasets. This was achieved by first training a classifier on the original dataset. This forms a baseline for comparing performances of the models. Then MFC-GAN, AC-GAN, and SMOTE were used to generate more samples from the minority classes. The resulting samples were then augmented into the original dataset and classification was performed again using CNN. The performance of the CNN on the three different augmented datasets are then compared and discussed. Algorithm 2 provides a schematic overview of this experiment.

Algorithm 2 Experimental procedure

procedure DATA AUGMENTATION
$d \leftarrow$ *original imbalanced dataset*
train:
    $MFC\text{-}GAN(d)$
    $AC\text{-}GAN(d)$
    $FSC\text{-}GAN(d)$
augment:
    $d_{mfc} \leftarrow d + MFC\text{-}GAN_{samples}$
    $d_{smote} \leftarrow d + SMOTE_{samples}$
    $d_{acgan} \leftarrow d + AC\text{-}GAN_{samples}$
    $d_{fscgan} \leftarrow d + FSC\text{-}GAN_{samples}$
classify:
    $r_1 \leftarrow CNN(d)$
    $r_2 \leftarrow CNN(d_{mfc})$
    $r_3 \leftarrow CNN(d_{smote})$
    $r_4 \leftarrow CNN(d_{acgan})$
    $r_5 \leftarrow CNN(d_{fscgan})$
compare$(r_1, r_2, r_3, r_4, r_5)$
end procedure

Furthermore, the fidelity of generated minority samples from MFC-GAN was compared to state-of-the-art AC-GAN.

All models were implemented using `tensorflow 1.0`[1] and `Keras 2.0`[2]. SMOTE was implemented using[3]. Models were evaluated subjectively based on the plausibility of samples (i.e. visual inspection) and objectively by assessing the classification performance after augmentation.

6.3.1 Datasets

The models were tested using four publicly available datasets. These are, MNIST [104], E-MNIST [38], SVHN [134] and CIFAR-10 [98] datasets.

MNIST is a dataset of hand-written digits with ten classes $(0-9)$ consisting of 28×28 grey-scale images. MNIST has a total of 50k images training set, 10k images for validation and 10k test images. Both the training and validation sets were merged to form a more significant training set, and the test set was used as a holdout sample in classification. MNIST is a balanced dataset, and so we induced imbalance among its classes by undersampling. Two classes were chosen arbitrarily and their instances were reduced significantly to mimic a multi-classification imbalance problem. We could have chosen more but given the size of the dataset, we do not want to inhibit learning due to the number of training examples. Different experiments were run with adjacent classes chosen as minority classes in each run. The first run considers 0 and class 1 as minority, then classes 2 and 3 and so on. In each run only 50 samples in these classes were used (about 1% of the original). The rest of the classes remained unchanged and experiments were carried out on the new imbalanced MNIST dataset.

E-MNIST is an extended version of MNIST. The dataset also consists of 28×28 grey-scale images with 62 classes $(0-9,\ A-Z$ and $a-z)$. For our experiments, the *byclass* grouping was used with $814,255$ samples in total. The dataset consists of $697,932$ training samples and $116,323$ samples for testing. The distribution of samples across classes in the training data is not balanced; thus, experiments on this dataset did not require inducing artificial imbalance. E-MNIST contains many classes with a considerably small number of samples than others with 21 out of 62 classes having less than 3000 samples. These classes include class G, K, Q, X, Z, c, f, i, j, k, m, o, p, q, s, u, v, w, x, y & z, where the ten least populated were used in our experiment.

SVHN dataset contains google street view of house numbers across ten categories $(1,2,3,4,5,6,7,8,9,0)$. This dataset consists of 32×32 pixels images with 73k and 26k train and test images set. These images appear noisy with other numbers in the

[1] https://www.tensorflow.org/
[2] https://keras.io/
[3] https://github.com/tgsmith61591/smrt

background and the dataset is not balanced. Similar to MNIST, we induced artificial imbalance by considering 50 samples in classes 1 and 2 to form a multi-class imbalance scenario with the rest of the classes unaltered.

CIFAR-10 dataset is made up of 32×32 images of real objects. It has fifty thousand training images grouped into ten classes namely, Aeroplane, Automobile, Bird, Cat, Deer, Dogs, Frog, Horse, Ship and Truck. Sample distribution across these classes is balanced with five thousand samples in each class. We induced artificial imbalance by considering 50 samples in *Aeroplane* and *Automobile* classes. The dataset has ten thousand test set with one thousand samples from each category. In all the datasets, the test sets were used as a hold out in evaluating the classification model.

6.3.2 Samples Generation

We perform augmentation by synthesizing more samples.AC-GAN, FSC-GAN and MFC-GAN were first trained using the imbalanced datasets described in Section 6.3.1. The three models were then used to generate minority samples, these samples were then used to augment the original datasets. Samples generated using SMOTE were produced by repeatedly applying SMOTE to oversample the class of interest as the minority sample and the rest of classes as the majority sample.

Regarding SVHN and CIFAR-10, the four models MFC-GAN, FSC-GAN, AC-GAN, and SMOTE were used to generate the class of interest (the minority class). These are classes 1 and 2 in SVHN and *Aeroplane* and *Automobile* classes in CIFAR-10. As for E-MNIST, we chose classes $G, K, Q, f, j, k, m, p, s, y$ as the class of interest (minority classes). These were chosen because they have the least number of instance. Every class in the MNIST dataset was considered a minority class (by undersampling each of them at different runs).

6.3.3 Image Classification

Our classification model is Convolutional Neural Network (CNN). The CNN used for MNIST and E-MNIST has three layers with a soft-max activation layer on top. The first two layers are convolution layers with 3×3 kernels which are followed by a 2×2 max-pooling layer. The two layers have a filter map of size 32 and 64 respectively. This is followed by a fully connected layer with 128 neurons that feeds into the final soft-max layer (with 10 and 62 output neurons for MNIST and E-MNIST respectively). All layers are ReLu activated, and a dropout ratio of 0.5 was used in the fully connected layer. Adadelta optimiser [195] (an extension of Adagrad) was used with default settings and

weights were initialised using random uniform distribution. The same model was used in SVHN experiment but with a different input channel and input size to accommodate the images.

For CIFAR-10 experiment, we increase the number of convolution layers to three (with channel sizes 32,32 and 64) and reduced the dropout ratio to 0.2. The number of neurons in the fully connected layer was also increased to 512 and the CNN was trained with SGD optimizer using learning rate (lr) of $1 \times 10^{-}3$ and decay of $lr/epoch$. The initial experiment trains the CNN on the original imbalanced dataset. Then the model is trained by augmenting the dataset using one of the approaches considered. Both CNNs were trained using a batch size of 64 for CIFAR-10 and 100 for the others over 25 epochs and we evaluated on the holdout test sets from each of the datasets described.

The choice of the CNN models above was made to evaluate the proposed method (MFC-GAN) on generating images of minority classes. This was achieved by first, classifying the original datasets using CNNs, then classifying the augmented datasets and comparing the results. In this way, we have an objective measure for the quality of samples generated by our model and how it compares to other methods. This is in addition to the subjective evaluation based on the visual inspection of the generated images.

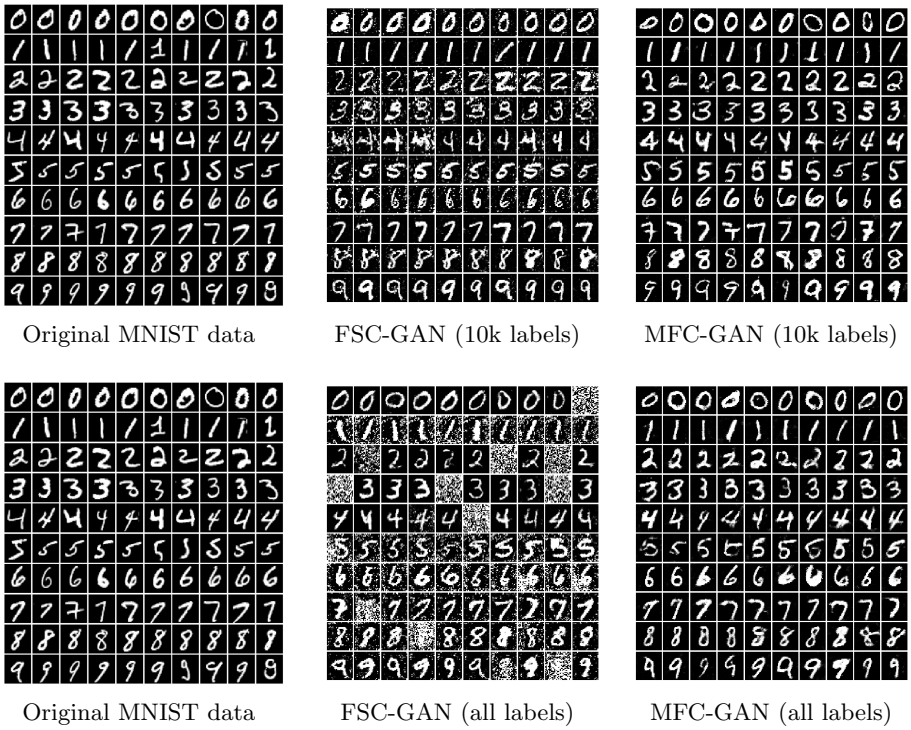


Figure 6.2: *FSC-GAN versus MFC-GAN on MNIST dataset*

6.4 Quantitative and Qualitative Evaluation

A preliminary experiment comparing MFC-GAN against FSC-GAN [3] was carried out using the MNIST dataset. This was achieved by reducing the number of labeled instances in the dataset across all classes. Figure 6.2 shows that MFC-GAN generated better quality samples and considerably reduced the amount of artifacts. The results also show that MFC-GAN can effectively handle both labeled and unlabeled instances. It is worth noting that MFC-GAN generates good quality images even in the presence of a large number of unlabeled instances (50K unlabeled instances, Figure 3.2c). The training time was also reduced considerably (by a factor of 10) with MFC-GAN producing plausible samples at about 50 epochs while FSC-GAN reaches optimum at 500 epochs. The results suggest that MFC-GAN would be a suitable model for augmentation.

MFC-GAN was also applied to imbalanced datasets to evaluate the quality of generated samples. The models were initially evaluated subjectively using visual inspection. Figures 6.4, 6.3, 6.5, 6.6 and 6.7 compare the original images and the generated samples. The minority classes in MNIST, SVHN, and CIFAR-10 dataset are highlighted using a red line for the different experiments conducted. For E-MNIST, we report the performance from the ten minority classes. Using MFC-GAN model, we were able to generate the minority classes without artifacts. Thus, the samples are good candidates for augmentation. As can be seen, poor minority class samples were generated by AC-GAN model and in some cases, it was biased toward the majority class. The classification performances are reported in tables 6.2, 6.1 and 6.3. Several common evaluation metrics were used in the experiments including balanced accuracy, sensitivity, specificity and Geometric Mean (G-Mean). These metrics were computed as follows:

$$Sensitivity = \frac{tp}{tp + fn} \tag{6.5}$$

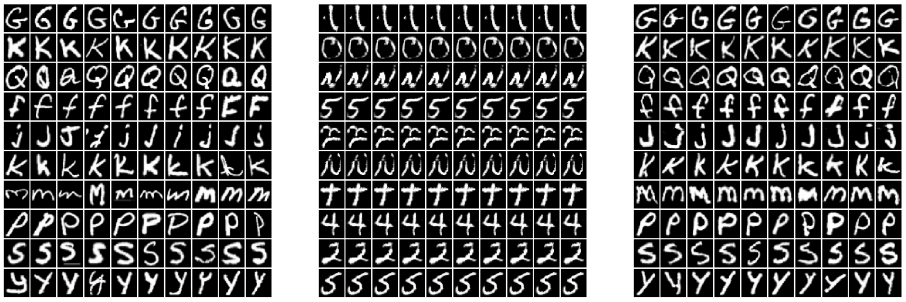
$$Specificity = \frac{tn}{tn + fp} \tag{6.6}$$

$$G\text{-}Mean = \sqrt{Sensitivity \times Specificity} \tag{6.7}$$

$$F1\text{-}score = \frac{2tp}{(2tp + fp + fn)} \tag{6.8}$$

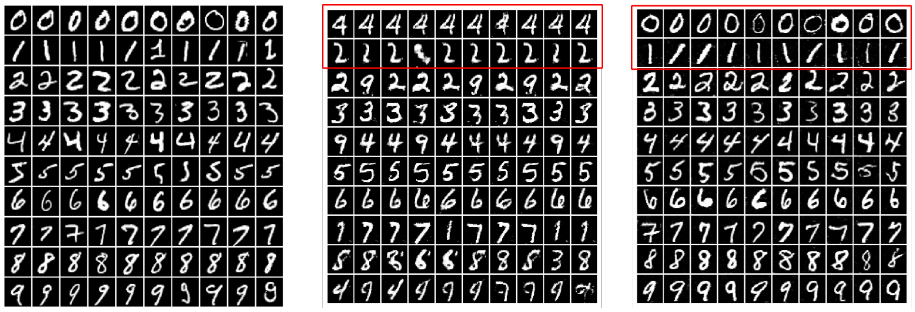
$$BalancedAccuracy = \frac{tp + tn}{2} \tag{6.9}$$

$$Prescision = \frac{tp}{tp + fp} \tag{6.10}$$



Original E-MNIST samples | AC-GAN samples | MFC-GAN samples

Figure 6.3: *Original images (left) with AC-GAN and MFC-GAN generated samples (middle, right) from E-MNIST dataset with minority class instances highlighted in red.*



Original MNIST samples | AC-GAN samples | MFC-GAN samples

Figure 6.4: *Original images (left) with AC-GAN and MFC-GAN generated samples (middle, right) from MNIST dataset with minority class instances highlighted in red.*

where tp stands for true positive, tn denotes true negative, fp and fn denotes false positive and false negative respectively.

6.5 Discussion

Tables 6.1, 6.2 and 6.3 show that the CNN achieved better performances when it was trained on the MFC-GAN generated samples. Higher sensitivity, balanced accuracy and G-Mean demonstrate that the MFC-GAN model was able to generate samples from minority classes in a multi-classification problem. It has to be pointed out that all the figures in all tables have been rounded to the nearest two decimal points. Results also show that MFC-GAN out-performed SMOTE and AC-GAN on all SVHN and CIFAR-10 minority classes, and in 7 out of 10 E-MNIST and MNIST, minority classes. The fidelity and diversity of MFC-GAN minority samples made classification easier for the CNN. The diversity of generated samples indicates no sign of mode collapse in

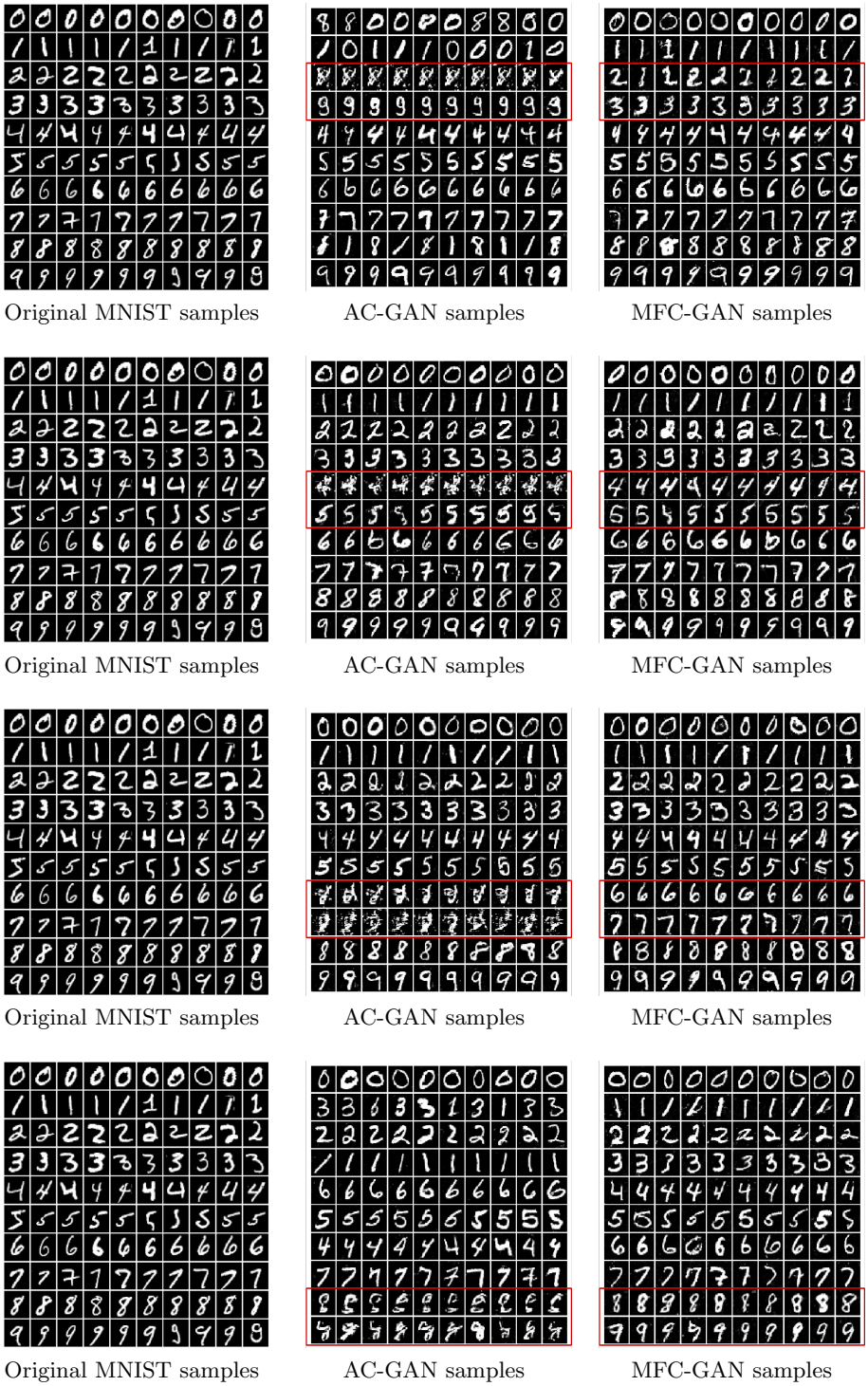


Figure 6.5: *Minority class instances (highlighted in red) in different runs.*

Original SVHN samples AC-GAN samples MFC-GAN samples

Figure 6.6: *Original images (left) and generated images from AC-GAN and MFC-GAN, minority classes are highlighted in red rectangle*

Table 6.1: *Results of SMOTE, AC-GAN, FSC-GAN and MFC-GAN classification performance on MNIST when each class is used as a minority.*

Metric	Model	0	1	2	3	4	5	6	7	8	9
Sensitivity	Baseline	0.83	0.93	0.64	0.73	0.68	0.70	0.73	0.65	0.62	0.58
	SMOTE	0.92	0.94	0.76	0.89	0.81	0.87	0.87	0.79	0.79	0.76
	AC-GAN	0.77	0.89	0.55	0.71	0.58	0.88	0.85	0.66	0.68	0.70
	FSC-GAN	0.78	0.87	0.60	0.58	0.49	0.51	0.61	0.48	0.38	0.41
	MFC-GAN	**0.98**	**0.98**	**0.83**	0.85	0.76	0.71	**0.88**	**0.90**	**0.89**	**0.83**
Specificity	Baseline	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	SMOTE	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	AC-GAN	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	FSC-GAN	1.00	1.00	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00
	MFC-GAN	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Accuracy	Baseline	0.91	0.97	0.82	0.87	0.84	0.85	0.86	0.83	0.81	0.79
	SMOTE	0.96	0.97	0.88	0.95	0.90	0.93	0.91	0.89	0.90	0.88
	AC-GAN	0.89	0.95	0.78	0.85	0.79	0.94	0.92	0.83	0.84	0.85
	FSC-GAN	0.89	0.94	0.80	0.79	0.74	0.75	0.80	0.74	0.69	0.63
	MFC-GAN	**0.99**	**0.99**	**0.92**	0.92	0.88	0.85	**0.94**	**0.95**	**0.94**	**0.92**
Precision	Baseline	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99
	SMOTE	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99
	AC-GAN	1.00	1.00	1.00	0.99	1.00	0.99	1.00	1.00	0.99	0.95
	FSC-GAN	1.00	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00	0.97
	MFC-GAN	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.99	0.99	**0.99**
F1-score	Baseline	0.91	0.96	0.78	0.84	0.81	0.82	0.84	0.79	0.77	0.73
	SMOTE	0.96	0.97	0.87	0.94	0.89	0.93	0.93	0.88	0.89	0.86
	AC-GAN	0.87	0.94	0.71	0.83	0.73	0.94	0.92	0.80	0.81	0.80
	FSC-GAN	0.88	0.93	0.75	0.73	0.65	0.67	0.76	0.65	0.55	0.44
	MFC-GAN	**0.99**	**0.99**	**0.91**	0.91	0.87	0.83	**0.93**	**0.94**	**0.94**	**0.90**
G-Mean	Baseline	0.91	0.97	0.80	0.85	0.83	0.84	0.85	0.81	0.79	0.76
	SMOTE	0.96	0.97	0.87	0.94	0.90	0.93	0.94	0.89	0.89	0.87
	AC-GAN	0.88	0.95	0.74	0.84	0.76	0.94	0.92	0.82	0.83	0.83
	FSC-GAN	0.88	0.93	0.77	0.76	0.70	0.71	0.78	0.69	0.62	0.64
	MFC-GAN	**0.99**	**0.99**	**0.91**	0.92	0.87	0.84	**0.94**	**0.95**	**0.94**	**0.91**

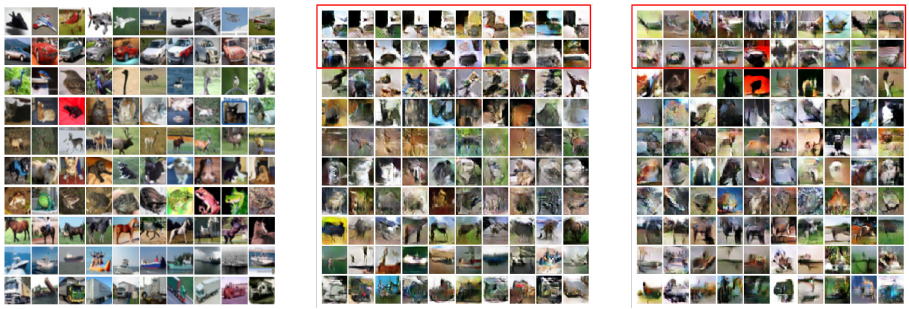
Table 6.2: *Results obtained from SMOTE, AC-GAN,FSC-GAN and MFC-GAN on ten E-MNIST minority classes.*

Metric	Model	G	K	Q	f	j	k	m	p	s	y
Sensitivity	Baseline	0.84	0.81	0.82	0.02	0.62	0.56	0.00	0.10	0.00	0.29
	SMOTE	0.82	0.73	0.80	0.25	0.84	0.58	0.23	0.38	0.01	0.48
	AC-GAN	0.77	0.76	0.87	0.14	0.57	0.57	0.00	0.21	0.00	0.18
	FSC-GAN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MFC-GAN	**0.89**	0.69	**0.94**	**0.48**	0.80	**0.68**	0.22	**0.77**	**0.14**	**0.65**
Specificity	Baseline	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	SMOTE	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	AC-GAN	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	FSC-GAN	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	MFC-GAN	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Accuracy	Baseline	0.92	0.90	0.91	0.51	0.81	0.78	0.50	0.55	0.50	0.65
	SMOTE	0.91	0.86	0.90	0.62	0.92	0.79	0.61	0.69	0.50	0.74
	AC-GAN	0.89	0.88	0.94	0.57	0.78	0.79	0.50	0.61	0.50	0.59
	FSC-GAN	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
	MFC-GAN	**0.94**	0.84	**0.97**	**0.74**	0.90	**0.84**	**0.61**	**0.89**	**0.57**	**0.82**
Precision	Baseline	0.91	0.64	0.91	0.43	0.72	0.79	0.00	0.55	0.00	0.53
	SMOTE	0.93	0.64	0.93	0.36	0.48	0.70	0.41	0.54	0.25	0.42
	AC-GAN	0.96	0.63	0.88	0.43	0.81	0.74	0.33	0.61	0.17	0.62
	FSC-GAN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MFC-GAN	0.80	0 .63	0.61	0.36	0.50	0.61	0.40	0.36	0.13	0.33
F1-score	Baseline	0.88	0.71	0.86	0.03	0.66	0.65	0.00	0.17	0.00	0.38
	SMOTE	0.87	0.68	0.86	0.29	0.62	0.64	0.29	0.45	0.01	0.45
	AC-GAN	0.86	0.69	0.88	0.21	0.67	0.65	0.00	0.32	0.00	0.28
	FSC-GAN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MFC-GAN	0.85	0.66	0.74	**0.41**	0.62	0.64	**0.29**	**0.49**	**0.13**	0.44
G-Mean	Baseline	0.92	0.90	0.90	0.12	0.78	0.75	0.00	0.32	0.00	0.54
	SMOTE	0.91	0.76	0.89	0.49	0.92	0.76	0.48	0.62	0.08	0.69
	AC-GAN	0.88	0.76	0.93	0.37	0.75	0.76	0.05	0.46	0.05	0.42
	FSC-GAN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MFC-GAN	**0.94**	0.83	**0.97**	**0.69**	0.90	**0.83**	0.47	**0.88**	**0.37**	**0.80**

Table 6.3: *SMOTE, AC-GAN, FSC-GAN and MFC-GAN performance on SVHN (Class 1 and Class 2) and CIFAR-10 (Aeroplane and Automobile) minority classes.*

Metric	Model	Class 1	Class 2	Aeroplane	Automobile
Sensitivity	Baseline	0.01	0.00	0.07	0.04
	SMOTE	0.18	0.31	0.06	0.07
	ACGAN	0.00	0.02	0.07	0.05
	FSC-GAN	0.02	0.09	0.00	0.00
	MFC-GAN	**0.51**	**0.68**	**0.07**	**0.08**
specificity	Baseline	1.00	1.00	1.00	1.00
	SMOTE	1.00	1.00	1.00	1.00
	ACGAN	1.00	1.00	1.00	1.00
	FSC-GAN	1.00	1.00	1.00	1.00
	MFC-GAN	**1.00**	0.99	**1.00**	**1.00**
Accuracy	Baseline	0.50	0.52	0.53	0.52
	SMOTE	0.59	0.65	0.53	0.53
	ACGAN	0.50	0.51	0.53	0.52
	FSC-GAN	0.51	0.54	0.50	0.50
	MFC-GAN	**0.75**	**0.83**	**0.54**	**0.54**
Precision	Baseline	1.00	0.99	0.93	1.00
	SMOTE	0.99	1.00	0.97	0.98
	ACGAN	1.00	1.00	0.93	0.89
	FSC-GAN	0.99	0.99	1.00	1.00
	MFC-GAN	0.98	0.96	0.80	0.81
F1-score	Baseline	0.02	0.09	0.12	0.08
	SMOTE	0.30	0.47	0.11	0.12
	ACGAN	0.00	0.03	0.12	0.09
	FSC-GAN	0.04	0.16	0.00	0.00
	MFC-GAN	**0.67**	**0.79**	**0.14**	**0.14**
G-Mean	Baseline	0.09	0.21	0.25	0.21
	SMOTE	0.42	0.56	0.24	0.25
	ACGAN	0.00	0.13	0.26	0.22
	FSC-GAN	0.14	0.30	0.00	0.00
	MFC-GAN	**0.71**	**0.82**	**0.27**	**0.28**

the model. Thus, with multiple fake classes, the GAN model was able to distinguish among classes better. A similar performance was recorded across all methods using the specificity, and this is reasonable as most classification models will accurately predict the majority class instances (tn).



Original CIFAR-10 AC-GAN samples MFC-GAN samples

Figure 6.7: *Original sample images (left) with AC-GAN and MFC-GAN generated samples (middle, right). Minority classes are highlighted in red*

FSC-GAN samples did not improve the classification in all experiments conducted as can be seen in Tables 6.1, 6.2 and 6.3. The results obtained showed that the classifier performed below the baseline when FSC-GAN samples were added to the training data. This is because FSC-GAN generated poor samples even when the number of classes is fairly balanced as shown in Figure 6.2. The other datasets are more challenging than MNIST and FSC-GAN goes into mode collapses when trained on the imbalanced datasets. The results indicate how negatively FSC-GAN is affected by the class-imbalanced problem.

AC-GAN model performed poorly on all the datasets in minority class image generation. This was evident by the below-average performance of the CNN when it was trained on AC-GAN samples. As can be seen in Figures 6.4, 6.5, 6.6 and 6.7, AC-GAN generated plausible majority class instances, however, the quality of generated minority class instances dropped significantly. In some cases, the model completely failed and became biased towards the majority class instances. This is consistent with the findings observed by [122]. For some specific classes a mode dropping in AC-GAN was observed, and the model generated the same image in all samples as can be seen in Figure 6.7b.

It was also observed that classification improvement was achieved when oversampling using SMOTE rather than augmenting with AC-GAN generated samples (Tables 6.1, 6.2 and 6.3). SMOTE achieved slightly better recall than MFC-GAN on two E-MNIST minority classes as seen in table 6.2. This is because E-MNIST has more

samples in the minority class (with the smallest class having 1896 samples). However, on the other datasets, SMOTE did not perform well when the number of minority class instances drops significantly. This also proves that MFC-GAN maintains good performance even with minimum number of samples in comparison with SMOTE and AC-GAN.

While good results have been obtained on MNIST, E-MNIST and SVHN, poor performances were recorded on CIFAR-10 by all models on minority class instances. AC-GAN model collapsed completely on CIFAR-10 while salient features required to distinguish samples effectively where not synthesized by MFC-GAN. These results might be attributed to the relatively small size of these images (i.e, 32×32 CIFAR-10 image patches) and the level of details within such tiny size. Although the samples generated by these models may look realistic, the characteristic features that will be vivid enough to train a classification model were missing. Increasing the number of minority samples from 50 to 100, 150, 200, 250 and 300 showed better but not significant improvement in performance. That said, as can be seen in Table 6.3, MFC-GAN produced slightly better performance amongst all these models.

Interestingly, poor results were obtained by all models for some specific minority classes. In particular, in the E-MNIST's minority classes m and s (Table 6.2). These minority classes were entirely missed by the baseline classifier, and very poor performance was reported using SMOTE, FSC-GAN and AC-GAN. MFC-GAN has also performed poorly in these classes. These results might be due to the similarity between some of these minority class instances and other majority class instance (i.e., class s is similar to classes 5, S, 2, z).

6.6 Conclusion

In this Chapter, a new data augmentation method using Multiple Fake Class Generative Adversarial Networks (MFC-GAN) was presented and evaluated on different public datasets. The classifier performance improved across standard metrics used for evaluating class-imbalanced datasets when compared with other popular augmentation and oversampling techniques. This implies that, MFC-GAN is a more reliable source of labeled data to rebalance classes in extreme imbalance scenarios. In the subsequent Chapter, we apply MFC-GAN to a real-world application domain.

Chapter 7

Application Domains

In this Chapter, we apply the model developed in Chapter 6 to real-world application domains. Two challenging domains are chosen where the classes of interest are under-represented. First, we consider the problem of face generation from minority attributes. Second, the problem of symbol generation from a Process and Instrumentation Diagrams (P&ID) with extreme imbalance ratio among symbol classes is addressed. The findings in this Chapter is part of the proceeding of the International Joint Conference on Neural Networks (IJCNN 2019).

7.1 Generating and Classifying Facial Attributes

Facial attribute classification is challenging because face features vary significantly from one person to another [53]. Face pose angles, different lighting conditions and variety of clothing such as eyeglasses, caps, and jewellery can create significant occlusion. Furthermore, imbalance in facial attribute classes creates a new dimension to the problem. Attribute classification approaches can be grouped into two categories. The first category considers the local image patches by feeding in outputs from attribute detectors. The problem with this group is the sole reliance on the efficiency of the detection model [53]. The second approaches process the global image to extract the required features and classify attributes. The latter methods are more robust and have provided state-of-the-art performances recently. Furthermore, global approaches have been implemented as multi-tasking approaches [53] and in some cases employing specific models to classify each attribute [117, 200]. More recently, multi-task models have explored the correlation between attributes to improve classification performances as shown in [71].

One of the common ways of improving classification performances in deep models is

through augmentation. Data augmentation neutralizes class imbalance and improves generalization by creating more diverse synthetic samples. However, in extreme imbalance cases, data augmentation may fail to produce enough variations in samples. In a facial dataset like CelebA, images are multi-labeled and a global augmentation approach on the images will not change the imbalance ratio. Moreover, augmenting specific facial attributes like hair colour, gender, eyeglasses or smile; presents a more challenging task for simple augmentation techniques [142]. A realistic approach will be to train a generative model that can capture these facial attributes while generating plausible samples that are suitable for augmentation.

Generating specific facial feature in images has many desirable applications such as security, fashion and in supporting other processes such as classification. Supervised GANs (like C-GAN) provides the required functionality to generate faces with specific attributes. AC-GAN, on the other hand, possesses some characteristics of C-GAN specifically conditional image generation. An extra classification task in AC-GAN reenforces class-specific generation and improves sample quality and diversity. Research into this area revealed that face generation with auxiliary classification frameworks mostly relies on a hybrid approach using an auto-encoder model to learn or extract features before the GAN model is trained. For instance, Fine-grained Multi-attribute GAN (FM-GAN) [181] was used to generate plausible faces with precise age using facial attributes. The model is a modified AC-GAN that incorporates attributes into the generator. The authors used the conditional reconstruction of the embeddings and considered three sets of attributes: age, gender, and ethnicity. FM-GAN was trained on CelebA and the synthesized images that were used to augment MORPH II dataset[1]. The new dataset was evaluated using a Convolutional Neural Network (CNN) and results obtained showed that the classifier performed better when the synthetic samples were added. Other Similar approaches includes Balancing GAN (BAGAN) [122], MelanoGAN [15] and Data Augmentation GAN (DAGAN) [9].

All these models share some similarity with our approach, that is synthesizing more samples for augmentation. However, we use a different GAN model with multiple fake classes, and we investigate face generation from attributes that are under-represented. CelebA dataset is one of the most widely used benchmarks for facial attributes classification and face generation. While significant achievements have been recorded on this dataset, some interesting potentials still remain untapped. Hand *et al.* [70] pointed out that the dataset is biased towards posed celebrity images that are not indicative of the real world. Looking at the attribute distribution across images, we can see that the dataset is biased toward frontal faces, smiling and mostly young celebrity pictures.

[1] https://ebill.uncw.edu/C20231_ustores/web/classic/store_main.jsp?STOREID=4

Again, models trained on this dataset without putting into account such biases might perform poorly on a different domain. And balancing by re-sampling a class directly affects other class distribution as well [70]. Generative Adversarial Networks (GANs) are state-of-the-art in image generation and supervised GANs offers a class-specific sample generation from labels. However, GANs like other neural networks when trained on imbalanced classes are affected by this problem. For instance, both Auxiliary Classifier GAN (AC-GAN) [140], C-GAN and GAN [63] avoid generating minority classes in extreme class imbalance cases [122].

In this Chapter, MFC-GAN is trained on a facial image dataset, and we demonstrate that class imbalance problem could be addressed through re-sampling of the minority class. The model is evaluated on face image generation problem from face image attributes that are under-represented in the CelebA dataset [117].CelebA attributes are represented as binary labels describing the presence or absence of a face feature such as a beard or no beard. The images distribution across these attributes varies significantly thereby creating a class imbalance problem. We explored this problem and created more scenarios by reducing the number of samples in the minority classes. Our experiments considered the eyeglasses attribute, and the generation facial images with eyeglasses from reduced number of instances in the class. Furthermore, the generated samples were used as additional samples to rebalance the classes and improve classification performance in a convolutional neural network.

7.2 Symbols Generation and Classification

Engineering drawings like Process and Instrumentation Diagrams (P&ID), circuit schematic diagrams and architectural drawings are needed in the design, construction and maintenance of facilities in many industries such as the oil and gas industry. Some of these diagrams are legacy hand-drawn diagrams. These documents contain vital information that is frequently referenced and interpreted for efficient decision-making processes. However, reading such drawings is challenging, time-consuming and requires expert knowledge in a domain where standards are dynamic. Given that there is a substantial volume of such documents in the industry, there is a growing need to digitize these diagrams for synergy and easy maintenance of many crucial facilities[131].

Automating symbol recognition and classification is challenging in complex engineering drawings like P&ID because diagrams are generally large and noisy. Again, diagrams contain connected components and lots of text that comes with context information such as flow dependent instructions. Coupled with the poor image quality, symbols in diagrams can be composed of many symbols, and in a specific context, the whole or

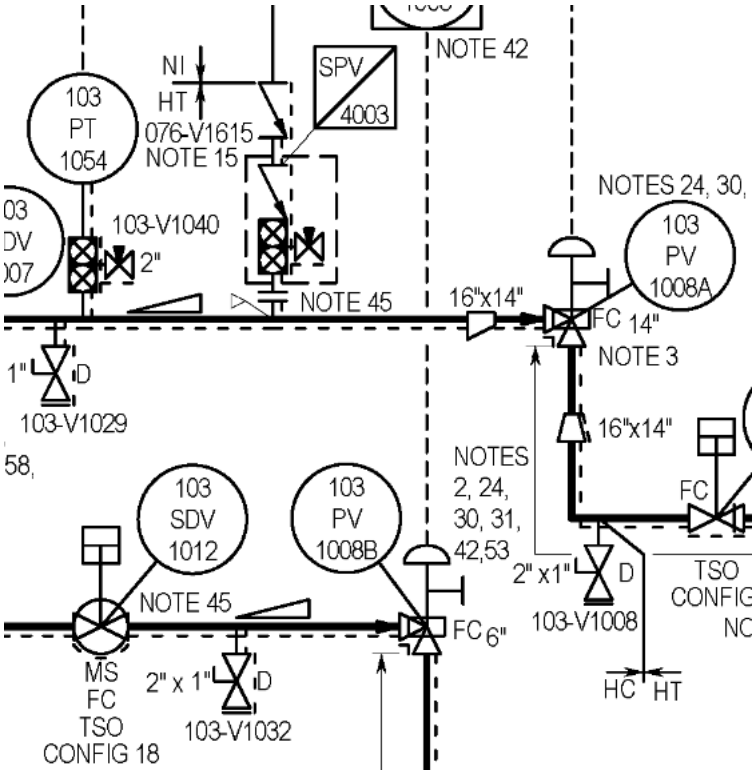


Figure 7.1: *A section of Process and Instrumentation Diagram (P&ID).*

section of the diagram is a single symbol. The composition relationships complicate the symbol recognition process. The problem of extreme class-imbalance among symbol classes further complicates the automation process. Figure 7.1 shows an example of P&ID diagram from an oil facility.

Traditional symbol recognition approaches, such as [194] produced a symbol recognition system that can match symbols from different domains. The procedure makes use of three simple steps, a loop identification step, a partial symbol identification stage then a symbol matcher. The system makes use of an existing symbols database as input and applies components matching on new symbols read. The system was successful in recognizing symbols from photocopies of different domain drawings such as electrical circuits, flowchart, chemical plant, logic. Similarly, scanned logic diagrams were used in [42]'s experiment to test for the presence of logic symbols, and the experiment recorded a 98.3% accuracy over 53 images with seven classes of gates. The procedure starts with the removal of text labels and connecting wires using morphological opening operation with a selected structuring element. Symbols are extracted and identified using three criteria; the Euler number of the symbol (the difference between the connected components and holes), the spike ratio and the circularity of the symbol. Other techniques developed include junction detection [144], heuristics digitization [130], SIFT

features [13] and multi-sliding window [58].

A common feature of the techniques mentioned in the preceding paragraph is the use of heuristics and a feature engineering method before a classifier is trained. The tight choices of tools limit the generalization and re-usability of these approaches across different domains. Again, symbols are not orientation dependent and have many contextual information in a P&ID. Also, essential problems in this domain such as class-imbalance problem and the need for domain knowledge were not addressed. In the following sections, we apply MFC-GAN in improving symbols classification in a P&ID. To the best of our knowledge, this is the first time a GAN model is being used to generate symbols from P&ID. The closest research to this is LoGAN [125] and BAGAN [122] which were used to generate trademark logos and traffic signs respectively. Moreover, we consider the problem of image generation in a class-imbalanced dataset with multi-classification scenario. We train MFC-GAN on a set of symbols obtained from a P&ID, and we demonstrate that MFC-GAN is capable of generating symbols even in extreme imbalance cases. Our experiments consider the nine least populated symbol classes and we show that the performances of a convolutional neural network could be improved by augmenting synthetic images from MFC-GAN.

7.3 Data Augmentation Approach

MFC-GAN model is trained on the two datasets described in section 7.4.1 and 7.4.2. The trained model was used to generate plausible samples from the minority classes. The generated minority samples are then added to the original training set to rebalance the dataset. Finally, we validate the approach on classification task using a Convolutional Neural Network (CNN).

Multiple fake classes were prepared from the binary facial attributes in the CelebA and symbols datasets similar to the procedure described in Chapter 5. The model was also trained using the objective function described in Chapter 6; however, the training procedure employs oversampling to emphasize equal participation of the minority classes. Algorithm 3 summarizes the training procedure used.

Both *steps* and *mini steps* are hyper-parameters which are tunable, and they control the behaviour of the oversampling routine. For these experiments, the *steps* variable was kept at a value of 1000 and a *mini steps* of 50 was used. But the sampling in the symbols generation was weighted according to the number of samples in the class to encourage learning on extreme cases.

A similar GAN model was used for both experiments. The generator model has one

Algorithm 3 MFC GAN Training procedure

for i in *iterations* **do**
  mini batch $\leftarrow$ *next training batch*
  evaluate $\mathcal{L}_{\mathrm{D}}$ using *mini batch*
  evaluate $\mathcal{L}_{\mathrm{G}}$ using *mini batch*
  if i in *steps* **then**
    for $j < k_{ministeps}$ **do**
      mini batch $\leftarrow$ *next minority batch*
      evaluate $\mathcal{L}_{\mathrm{D}}$ using *mini batch*
      evaluate $\mathcal{L}_{\mathrm{G}}$ using *mini batch*
    end for
  end if
end for

linear layer and five transpose convolution layers with strides of two in each layer. Batch normalization was used between adjacent layers and all layers were activated using Leaky ReLu apart from the final layer which is sigmoid activated. The generator takes as input a random noise vector and the facial attributes as embeddings. The output is a 64×64 coloured image for faces and a grey scale image in the case of symbols which were sent to the discriminator for training. The discriminator is trained on two set of images, the real training samples, and the generated samples. The first four layers are convolution layers with strides of two which are activated using Leaky ReLU and batch normalization is used between layers. The final layer is parallel linear layer sigmoid output and a classification layer. We used a batch size of 100 and a learning rate of $1e-4$. Spectral normalisation [128] was used in both the generator and the discriminator, and we also experimented with gradient penalty [66] in our face generation experiments.

7.4 Experimental Design

Our experiments analyze class-specific image generation and classification in a class imbalanced dataset. Experiments were conducted on celebrity faces with attributes dataset (CelebA dataset) and engineering symbols generation using class labels. For the faces experiment, we considered eyeglasses attribute as minority class with 13193 instances. Different experiments were carried out on a reduced number of instances using this minority class. We considered 200, 500, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000 and 10000 minority instances in different runs. Setting the number of minority class instances to range between 200 to 10000 allowed us to assess our model in generating data in extreme imbalanced scenarios where the majority class

instances represent almost 95% to 99% of the data, while the minority class instances presence ranges roughly between 0.1% to 5% of the data. For the symbols experiment, we consider the nine least populated symbols namely, Angle choke valve, Angle Valve, Injection Sample Point, Back Pressure Regulating Valve, PS Gate Valve, control valve, Through Conduit Gate Valve, control valve globe and a Pressure Regulating Valve. These symbols have between two and forty two instances in each class.

For our image generation experiment, we report the quality and diversity of the generated minority samples after each run. For classification experiments, we extend the training data with generated minority samples from trained models (AC-GAN and MFC-GAN). Then, a CNN classifier is trained on the extended dataset, and the classification performances on the minority classes are reported.

7.4.1 Face Dataset

CelebA was created by annotating images from CelebFaces dataset with a face bounding box, facial landmarks and attributes annotations. It consists of 202k images with forty binary facial attributes. CelebA is used as a benchmark in face detection and facial landmarks detection such as eyes, nose and mouth and facial attribute classification. CelebA attributes include curly hair, goatee, bald, male, eyeglasses, and other fine-grained attributes like wearing lipstick, heavy make-up, 5 O'clock shadow, arched eyebrows, and others. The multi-label attributes of an image open some interesting scenarios when investigating the dataset. These include the relationship between some attributes such as young and attractive, the biased distribution of attributes across samples and an unconstrained environment in facial images which creates variation among similar attributes. For our experiments, the dataset was used to perform face generation and classification of facial attributes using a low number of instances of a particular class. The dataset was preprocessed by cropping the head region using the face annotation bounding box and some heuristics. The crop was made just enough to accommodate the chin to the hair with ears visible on both sides (where applicable). The cropped image was then resized to 64×64 image patch. Before training, the images were normalized and labels preprocessed as described in Section 7.3. The dataset was split into a train and a test set. The test set is made up of six thousand samples with an equal number of majority and minority samples.

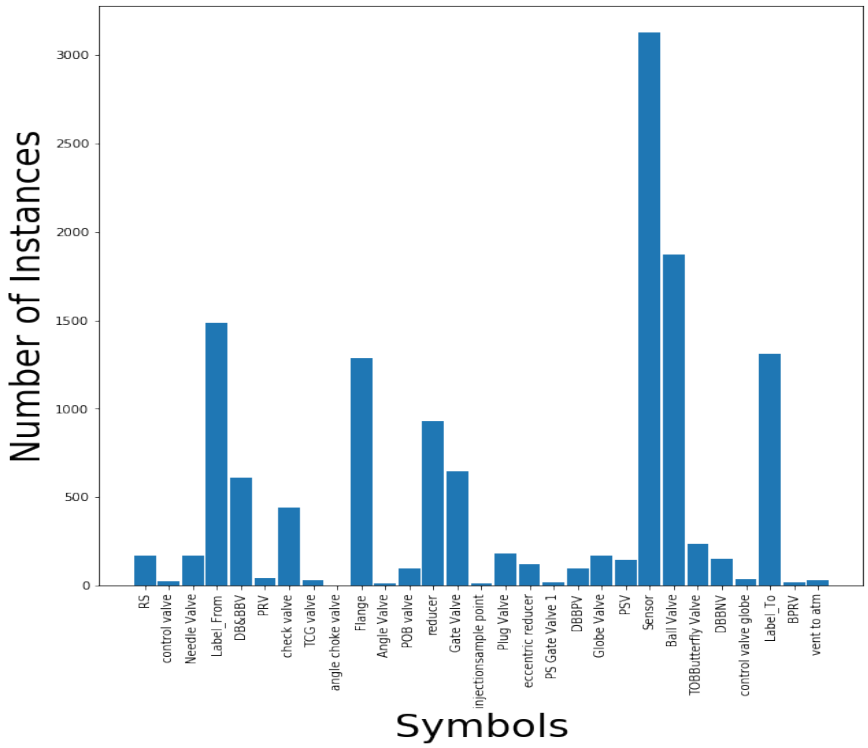


Figure 7.2: *Class distribution in the symbols dataset.*

7.4.2 Symbol Dataset

Symbols were obtained from the P&ID by annotating the diagrams using Sloth [2]. This generates a JSON file containing the bounding box coordinate for each symbol. These symbols were cropped out automatically via a python script and resized to 64×64 grey-scaled images. This dataset is quite small, with 13478 symbols across 29 classes. Moreover, the class distribution of symbols is extremely imbalanced with more than half of the samples in the three largest classes (sensor, ball valve and labelfrom and labelto adding up to 7789 instances or $\sim 58\%$) and the least populated symbol class having just two instances (angle choke valve $\sim 0.01\%$). Figure 7.2 shows the distribution of symbols in the dataset.

7.4.3 Face Generation from Attributes

Control generation was achieved by conditioning the generator on attribute labels. Several experiments were carried out with a different number of samples in the minority classes specifically eyeglasses attribute. For each run, the MFC-GAN model was

[2] https://sloth.readthedocs.io/en/latest/

trained from the scratch and samples were generated after the training is completed. A similar experiment was performed with AC-GAN using eyeglasses attribute. For a fair comparison, a similar generator and discriminator structure was used in AC-GAN. We then examined the quality of the generated samples and how suitable these samples are for augmentation. Samples are good enough if they are of high quality and the required minority attribute appears in the image. The quality of the generated images from the two models is compared using established qualitative measures. We employ visual inspection and Frechet Inception Distance (FID) [75] to evaluate the quality and diversity of MFC-GAN and AC-GAN samples. A lower FID indicates a better sample quality and diversity. Visual inspection reaffirms the presence or absence of the attribute in a generated sample.

7.4.4 Facial Attributes Classification

Our classification model is a CNN with the same structure as the attribute CNN [117]. The attribute CNN has four convolution layers with max pooling layers between them. A fully connected layer follows the last convolution layer with a classifier as the final layer. We used a soft-max classifier, a filter size of 3×3 in all layers and trained the CNN from scratch as against starting from pre-trained weight different from [117]. We performed an initial classification of samples using reduced number of samples in the minority class i.e eyeglasses. We refer to this experiment as baseline. The number of samples in the minority classes is then extended with MFC-GAN generated samples after training on the same number of minority samples, and the classifier is retrained again. In a similar manner, AC-GAN samples were also used to extend the training data, and the CNN is trained from scratch each time. Finally, we report the F1-score and true positive rate of the classifier on each run. We compare the performances of the CNN when MFC-GAN samples are added to when AC-GAN samples are added.

7.4.5 Symbols Generation from Minority Classes

Similar to what was done with faces, the generator was conditioned on the symbol class labels to promote class-specific samples generation. We experimented on the generation of multiple minority class instances i.e the nine least populated classes from the symbols datasets. These classes have 2, 13, 15, 17, 17, 27, 31, 36 and 42 samples. The choice of these classes was because they are the most affected by extreme imbalance ratio. The model was trained only once on this dataset and the samples were generated after training was completed. During training, the minority classes were resampled as described in Algorithm 3. The images were not preprocessed or augmented during

GAN training. We compared the quality of samples generated and the effectiveness of our approach to AC-GAN. Visual inspection was used to determine if the generated symbols are from the required category or not. This also gives information about the quality of the generated symbol.

7.4.6 Improving Symbols Classification

The CNN used has three convolution layers with 32, 64 and 128 filters that are followed by a fully connected and a softmax classification layer. The convolution layers use a kernel size of 3×3 and a 2×2 max-pooling layer after the first and second convolution layers. The fully connected layer has 256 neurons, and no drop-out or batch norm was used. The softmax layer is a 29-way classification layer, and all other layers are Relu activated. Before the CNN training, the dataset was split into a training and a testing set. The test set was used as a hold out set for evaluation and it is worth noting that the same imbalance ratio (as in the original distribution) was maintained in the test set. A baseline performance on the test set was reported after training the CNN on the original training set. Then the minority class instances from MFC-GAN samples were added to the training set and the classification model is retrained from scratch. This procedure is repeated using AC-GAN generated samples and the classification performances were reported. The performances of the CNN are compared using true positive rates, balanced classification accuracy, G-mean and F1-score obtained from each experiment. These metrics were defined in Chapter 6.

7.5 Analyzing Sample Quality and Performance

For the experiment on face datasets, Figure 7.3 shows the sample data generated from each model by conditioning on the minority attribute. More of these samples with varying number of minority instances can be found in section B.2 of Appendix B. Tables 7.2 analyze further the quality of generated images obtained during the experiments using FID metric. The FID was measured by comparing 10k samples with eyeglasses from the training data and generated 10k samples from the models after training using the approach provided in [75].

The classification results on the test set are shown in Figure 7.4 which compares the performance of the baseline classifier and the two models using varying number of minority class instances. Tables 7.1 show the true positive rates (TPR) obtained when the models are used to augment the original dataset with more samples (generated) in a classification task. These results show clearly that when augmenting the dataset

Original sample AC-GAN sample MFC-GAN sample

Figure 7.3: *Comparing original images and AC-GAN and MFC-GAN generated samples. Samples were generated from training on 10k glasses instances.*

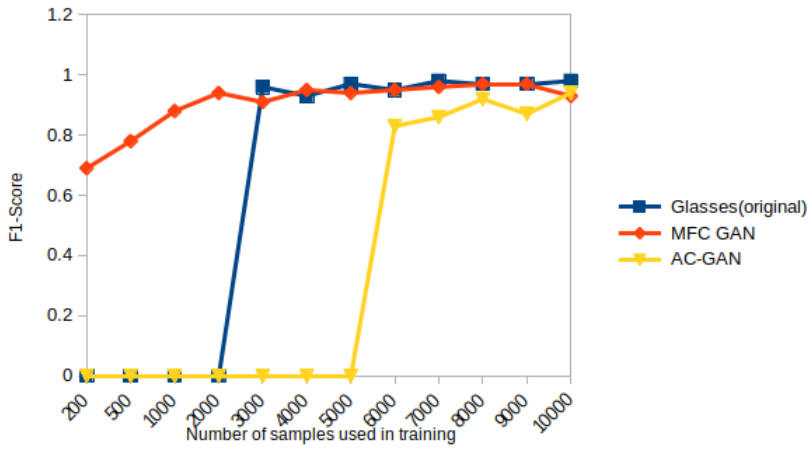
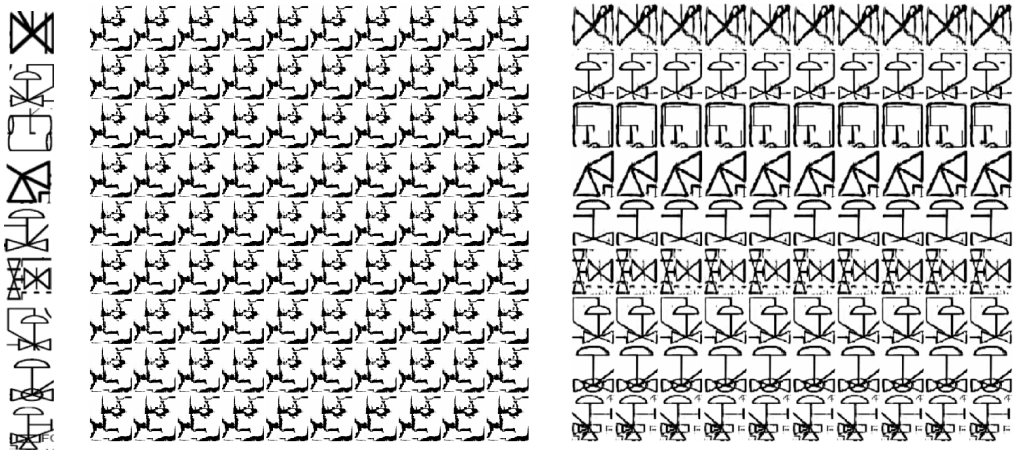


Figure 7.4: *The F1-score of the CNN classifier over varying number of instances with eyeglasses attribute.*

with MFC-GAN generated samples, the TPR was significantly improved in comparison with the baseline, particularly in extreme imbalanced cases (i.e. with 200 to 2000 samples). The results also show that MFC-GAN significantly outperformed AC-GAN in all scenarios.

With regards to the second experiment on P&ID, Figure 7.5 compares the generated samples from AC-GAN and MFC-GAN models with the original symbols from the diagram. We also report the symbols classification results in Table 7.3. Similar to the experiment on faces, our approach demonstrate superior performance in terms of image quality and better overall classification performances on different class imbalance metrics.



P&ID AC-GAN sample MFC-GAN sample

Figure 7.5: *Comparing original P&ID samples with AC-GAN and MFC-GAN generated samples.*

7.6 Significance of Synthetic Samples

As can be seen from the results obtained, the CNN classifier fails to detect minority class instances in extreme scenarios. This is evident from the results shown in tables 7.1, 7.3 and F1-scores from Figure 7.4. However, as the number of minority class instances increases, the CNN performance tends to improve slightly. For instance, in the faces experiment, a reasonable performance was obtained by the CNN when the number of minority class instances reached 3k for eyeglasses attribute. This clearly shows that in extreme cases where the number of minority class instances is minimal, a data augmentation is much needed and MFC-GAN samples becomes useful.

Closely looking at the face generation task, both AC-GAN and MFC-GAN generated realistic samples. However, AC-GAN samples did not fall within the required category as can be seen in Figure 7.3. This rendered the AC-GAN samples inadequate for augmentation. Thereby negatively affecting classification whereas MFC-GAN model performs significantly better when the dataset was extremely imbalanced. Moreover, the samples from MFC-GAN proved to be useful in augmenting the training set to boost classification performance. For instance, with 200 eyeglasses samples, the true positive and F1-score improved from 0 to 53 and ∼70% respectively.

Augmenting AC-GAN samples into the training set did not improve the accuracy of the CNN in identifying eyeglasses attributes. Visual observation of the generated images revealed that the model generated quality samples but was not of the required

Table 7.1: *True positive rate of eyeglass attribute classification and highlighted in bold are the instances where MFC-GAN performed better than the baseline and AC-GAN.*

Number of samples	Baseline	AC-GAN	MFC-GAN
200	0.0	0.0	**0.53**
500	0.0	0.0	**0.64**
1000	0.0	0.0	**0.79**
2000	0.0	0.0	**0.90**
3000	0.92	0.0	0.84
4000	0.86	0.0	**0.91**
5000	0.95	0.0	0.89
6000	0.91	0.71	**0.91**
7000	0.96	0.75	0.92
8000	0.94	0.86	**0.95**
9000	0.95	0.78	0.94
10000	0.95	0.89	0.93

minority classes. Adding these samples to the training data confuses the classification model particularly when the number of original samples is small. These samples over-shadowed the real data and prevented the model from understanding the true discriminative feature/attribute in the samples. However, with significant real samples in the training data, the effect of the spurious samples is minimized. This is in line with the observations by [122] and shows that AC-GAN is inadequate in capturing the true data distribution in an extreme class imbalance scenario.

On the other hand, augmenting MFC-GAN in the training set resulted in better classification results. This improvement was significant in extreme cases where the number of minority class instances is kept to minimal. MFC-GAN model trained on two hundred eyeglasses samples was able to capture the real data distribution and was capable of producing the required minority samples necessary to improve classification results. Visually observing the samples in Figure 7.3 shows the presence of the minority attributes which further explain the improvement in performance. These samples also had better mean FID than the samples generated by AC-GAN as shown in table 7.2. An interesting behaviour of the MFC-GAN model is that it was able to associate eyeglasses with both male and female generated samples.

The symbol dataset is much smaller and less complicated than the face dataset, but similar performances were observed. In this case, the AC-GAN model was unable to generate instances of the minority class at all as can be seen in Figure 7.5. The model collapsed to producing frozen noise samples on all the minority classes. The samples are unrealistic and do not contain any visible symbol features. Consequently, augmenting

Table 7.2: *Mean Freschet Inception Distance (FID) of generated images from experiments on different number of samples with eyeglasses.*

Number of samples	AC-GAN	MFC-GAN
200	72.97	81.26
500	73.51	**72.54**
1000	72.24	**69.65**
2000	75.34	83.36
3000	75.58	81.83
4000	73.18	**65.31**
5000	74.66	**68.02**
6000	75.38	**60.19**
7000	71.90	**59.64**
8000	70.67	**70.57**
9000	74.69	**57.17**
10000	73.96	**59.34**

these samples did not improve the baseline performances in most cases. Whereas MFC-GAN generated far superior and more realistic samples. Visual inspection revealed distinct symbols features and the required categories were generated in each instance. This shows that our model was able to capture salient attributes of each symbol class even in extreme cases. Moreover, MFC-GAN high-quality samples had a positive effect on the performance of the classifier. For example, the G-Mean and sensitivity improved from 0 to 100% on angle choke valve as can be seen in Table 7.3 with just two instances of the classes. This result is consistent in seven of the nine minority classes. However, we observed that the model did not improve the baseline in the other two classes viz control valve and PRV classes. A closer look at Figure 7.5 revealed a high similarity between symbols. There is extreme similarity between angle valve (fifth symbol from the top) with control valve globe (eighth symbols from the top) and PRV(seventh symbol from the top) and BPRV (second symbol from the top). Although symbols were distinctly generated, the similarity of symbols dwindled the classification results in these classes. The low precision in BPRV and control valve globe classes from Table 7.3 further solidifies this observation.

Despite improving classification performance on a reduced number of samples (minority classes), we observed that augmenting more samples could not achieve 100% true positive rate even with 10k real samples. We tried to push the results further by under-sampling the majority class but this did not influence the results much. We infer that this could be related to the classification model chosen because no hyper-parameter search or model tuning was done. In addition, the target of our experiments was to show the usefulness of our GAN generated samples in extreme class imbalanced

Table 7.3: *CNN performance on symbols classification.*

Metric	Model	angle choke valve	Angle Valve	BPRV	control valve	control valve globe	injectionsample point	PRV	PS Gate Valve	TCGvalve
Sensitivity	Baseline	0.00	0.50	0.60	0.88	1.00	0.80	1.00	1.00	0.89
	AC-GAN	0.00	0.75	0.80	1.00	0.91	0.80	0.92	0.80	0.89
	MFC-GAN	**1.00**	**1.00**	**0.80**	0.88	**1.00**	**0.88**	0.77	**1.00**	**0.91**
Specificity	Baseline	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	AC-GAN	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	MFC-GAN	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Precision	Baseline	0.00	1.00	1.00	1.00	0.85	1.00	0.72	1.00	1.00
	AC-GAN	0.00	1.00	0.80	0.73	1.00	1.00	0.86	0.80	1.00
	MFC-GAN	**1.00**	**1.00**	0.67	**1.00**	0.92	**1.00**	**0.91**	0.83	**1.00**
F1-score	Baseline	0.00	0.67	0.75	0.93	0.92	0.89	0.84	1.00	0.94
	AC-GAN	0.00	0.86	0.80	0.84	0.95	0.89	0.89	0.80	0.94
	MFC-GAN	**1.00**	**1.00**	0.73	**0.93**	**0.96**	**0.93**	0.83	0.91	**0.95**
Accuracy	Baseline	0.50	0.75	0.80	0.94	1.00	0.90	1.00	1.00	0.95
	AC-GAN	0.50	0.88	0.90	1.00	0.96	0.90	0.96	0.90	0.95
	MFC-GAN	**1.00**	**1.00**	**0.90**	0.94	**1.00**	**0.94**	0.89	**1.00**	**0.96**
G-Mean	Baseline	0.00	0.71	0.77	0.94	1.00	0.89	1.00	1.00	0.94
	AC-GAN	0.00	0.87	0.89	1.00	0.95	0.89	0.96	0.89	0.94
	MFC-GAN	**1.00**	**1.00**	**0.89**	0.94	**1.00**	**0.93**	0.88	**1.00**	**0.95**

scenarios and classification was only used as an evaluation criterion.

7.7 Conclusion

In this Chapter, we applied Multiple Fake Class Generative Adversarial Networks (MFC-GAN) to generate face images from under-represented attributes. Generating faces with flexible attributes reduces the challenges hindering the use of traditional augmentation approaches. Augmenting facial attributes is crucial in different computer vision domains such as identity management system, cross-age face validation and missing person identification. Furthermore, MFC-GAN was applied to generate symbols from P&ID diagrams. We show that even in extreme cases, MFC-GAN can synthesize quality symbols that improve classification performances. Producing more samples from a few available ones is in demand where labeled data is scarce or difficult to obtain, such as the oil and gas industry.

Chapter 8

Conclusion

In this chapter, we conclude and summarize the main findings resulting from this body of work.

The thesis addressed the problem of learning in the absence of enough labeled data and learning from class-imbalanced data. In particular, we developed novel generative adversarial networks that utilized both the generative and classification capabilities of GANs. These models were used to address two key learning challenges with limited labeled data; namely, image classification with large unlabeled data and data augmentation in class-imbalanced datasets. In addition, the chapter also presents the limitations of the approaches proposed and some potential future works in this regard.

8.1 Summary

In this section, we revisit the objectives from Chapter 1, and summarize the contributions and the major findings from these objectives.

- **A literature review of existing methods that addresses the lack of labeled data in deep learning.** In this thesis, we reviewed existing approaches in image classification where sufficient labeled examples could not be obtained (in Chapter 2). We highlighted the strengths and weaknesses of some of the popular techniques developed such as transfer learning, semi-supervised learning and unsupervised learning. The thesis also looked at the recent advances in GAN models in Chapter 3. In this respect, we explored what GANs can generate, the different flavours of objective functions and architectures, limitations and successful application areas.

- **Evaluate the performances of deep learning models on datasets that are poorly annotated and labeled.** In Chapter 4, we demonstrated how the performances of a deep convolutional model deteriorate in the absence of well annotated data. The experiments carried out showed the need for enough labeled examples before a model could be trained properly. Our findings also highlighted the limitation of simple data augmentation approaches in class-imbalanced data. The experiments also demonstrates that transfer learning in specific domains does not always work and such domains will always require dedicated data collection and annotation process or other label agnostic approaches to achieve the desired performances.

- **To create a novel method to train deep models using minimal number of labeled examples. This will mitigate the problem of insufficient labeled data in image classification.** A novel adversarial learning architecture incorporating latent label embedding, network switchers and multiple fake classes to train a classifier by utilizing large unlabeled data (in Chapter 5). A new method to train the model was created to leverage the changing scenarios of labeled and unlabeled data. One of the greatest appeals of our approach is its label-agnostic property. Also, our model supports a wide range of strategies from fully supervised, semi-supervised to weakly supervised learning.

- **To develop a framework that generates realistic class-specific samples in the absence of enough labeled instances.** The methods proposed utilized the generative property of the proposed model to synthesize more labeled training examples in a multi-classification task. To this end, we developed a new augmentation method using Multiple Fake Class Generative Adversarial Networks (MFC-GAN) in Chapter 6. MFC-GAN demonstrated improvement on the baseline classifier model and when compared to the state-of-the-art AC-GAN model in terms of sample quality and classification performances. The results indicated that by augmenting the training set with MFC-GAN generated samples, performance improved across common metrics used in evaluating imbalance classification. Our approach showed superior performance when compared to common augmentation and oversampling technique (i.e SMOTE).

- **To address class-imbalance in real-world problems with under-represented class instances using the methods and techniques developed.** Finally, we applied MFC-GAN to two challenging real-world domains, namely, legacy engineering drawings and face images with attributes in Chapter 7. Several experiments were carried out on a reduced number of instances in the classes of interest. Results obtained showed that MFC-GAN captured the

underlying data distribution from the class imbalanced datasets and generated realistic samples from the minority class instances. Our approach demonstrated improved performances than the baseline classification in extreme imbalance scenario while out-performing AC-GAN in all cases.

8.2 Limitations and Future Work

The models and techniques developed in this thesis have shown to be useful in areas where there are no enough labeled examples and where the number of instances in the class of interest is low. They have been successful in weakly supervised scenarios and class-imbalance problems. However, the thesis leaves a lot to be desired. The following are some potential future works that could be explored.

- An interesting extension to the methods proposed is automatic image correction. This can be achieved by synthesizing samples with desired properties while preserving the identity of the subject. Automatic generation and augmentation in domains such as face verification, cross-age verification and fashion require the subjects in the images to remain the same. For instance, in face generation from attributes, instead of generating a new face, the new method will add attributes like eyeglass, beard and side-burns to existing faces. Samples identity-preserving could be achieved in future by adding an identity constraining [123] component to the model.

- One of the limitations of MFC-GAN is that the classifier in MFC-GAN was used as a regularizer which ensures efficient gradient flow. The classifier objective promoted early convergence and better samples quality; hence, samples were useful in augmentation. On the other hand, the built-in classifier was kept below optimum and may not be an effectively learned classifier from the adversarial process. Future research could consider other techniques such as noise regularization [151] or gradient clamping [139] in the context of multiple-fake classes to keep the discriminator within touching distance of the generator. A classifier that can identify different categories of synthetic and real samples is crucial in predicting fake images/videos which are a great concern today.

- MFC-GAN was able to capture noticeable image characteristics such as object patterns with sufficient diversity in generated samples but struggles with subtle features. For instance, when the model was trained on facial attributes like goatee, little diversity was observed in the generated images. This is similar to

the problems observed by [7]. Although the model generated the required minority class, the quality and attribute diversity among generated samples was poor. Other interesting aspects were obvious such as the consistent link between male samples and the goatee attribute despite the model being trained on both male and female images. More results and analysis on the goatee experiments are in section B.3 of Appendices B. An exciting future direction in this regard is to use a more descriptive condition such as a scene graph [86]. The description could be used to capture many details such as structure, direction, and labels in order to promote the generation of subtle features.

- An interesting future work will be to consider the inter-relationship between the different label classes in class-imbalanced data. For instance, in CelebA data dataset, some facial attributes occur consistently alongside each other such as male and goatee, attractive and young. While others like beard and sideburns or beard and moustache frequently occur together in the dataset but are independent of one another. Trying to improve the number of samples in such classes using sample generation may indirectly affect the other. The recent work on attention mechanisms [197] in GANs have shown to produce better global coherence in objects. However, the inter-relation among labels was not exploited. Studying the inter-relationship among labels opens up a new discussion on better ways to condition generative models. This could be in form of other parametrized models that statistically describe the right prior (such as an Autoregressive model).

- The proposed methods were successfully used to generate symbols from extreme minority classes in a P&ID, and the samples were used to improve classification performances in the minority classes. An advancement toward this solution will be to produce a complete engineering drawing or a segment of it. Generating a whole diagram is challenging, given the uniqueness of each diagram coupled with noisy text. Moreover, the diagrams are very big and are context-dependent. The recent published ideas on Redrawing of Objects (ReDo) [32] and copy-paste GAN (CP-GAN) [10] opens a new dimension to the problem. Although both pieces of research are in object segmentation, the fascinating thing about these two research is the ability to insert new objects in an image and also extract mask from existing images. P&IDs are unique, given that the target image is a white canvas and the objects to be placed are connected symbols. Again, diagrams can be contextualized with conditions such as connectivity and position in order to generate a meaningful diagram. There are many challenges with using GANs, but it would be interesting to discover how a diagram generation can leverage GAN segmentation techniques and repeated in-painting.

Multiple fake class GANs are a brilliant idea with many applications; they produce high-quality samples and help the class imbalance problem, much more than SMOTE. They also cover data privacy nicely (because they are "fake" people/samples), and produce convincing samples that are not malicious but intended to mitigate problems with a lack of labeled data.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous systems. 2015.

[2] Ali-Gombe Adamu, Elyan Eyad, and Jayne Chrisina. Multiple fake classes gan for data augmentation in face image dataset. In *Neural Networks (IJCNN), 2019 International Joint Conference on*, 2019.

[3] Ali-Gombe Adamu, Elyan Eyad, Savoye Yann, and Jayne Chrisina. Few-shot classifier gan. In *Neural Networks (IJCNN), 2018 International Joint Conference on*. IEEE, 2018.

[4] Ali N Akansu, Paul A Haddad, Richard A Haddad, and Paul R Haddad. *Multiresolution signal decomposition: transforms, subbands, and wavelets*. Academic press, 2001.

[5] Adamu Ali-Gombe and Eyad Elyan. Mfc-gan: Class-imbalanced dataset classification using multiple fake class generative adversarial network. In *Neurocomputing*. Elsevier, 2019.

[6] Adamu Ali-Gombe, Eyad Elyan, and Chrisina Jayne. Fish classification in context of noisy images. In *International Conference on Engineering Applications of Neural Networks*, 2017.

[7] Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Boosting cross-age face verification via generative age normalization. In *International Joint Conference on Biometrics*, 2017.

[8] Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Face aging with conditional generative adversarial networks. *arXiv preprint arXiv:1702.01983*, 2017.

[9] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

[10] Relja Arandjelović and Andrew Zisserman. Object discovery with a copy-pasting gan. *arXiv preprint arXiv:1905.11369*, 2019.

[11] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[12] Yalong Bai, Kuiyuan Yang, Wei Yu, Chang Xu, Wei-Ying Ma, and Tiejun Zhao. Automatic image dataset construction from click-through logs using deep neural network. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 441–450. ACM, 2015.

[13] Donald Bailey, Andrew Norman, and Giovanni Moretti. Electronic schematic recognition. *Massey University*.

[14] Rishi Sharma Shane Barratt and R Sharma. A note on the inception score. 2018. *arXiv preprint arXiv:1801.01973*, 1801.

[15] Christoph Baur, Shadi Albarqouni, and Nassir Navab. Melanogans: High resolution skin lesion synthesis with gans. *arXiv preprint arXiv:1804.04338*, 2018.

[16] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008. Similarity Matching in Computer Vision and Multimedia.

[17] Shabab Bazrafkan and Peter Corcoran. Versatile auxiliary classifier with generative adversarial network (vac+ gan), multi class scenarios. *arXiv preprint arXiv:1806.07751*, 2018.

[18] Bridget Benson, Junguk Cho, Deborah Goshorn, and Ryan Kastner. Field programmable gate array (fpga) based fish detection using haar classifiers. *American Academy of Underwater Sciences*, 2009.

[19] David Berthelot, Tom Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

[20] Kofi Boakye, Sachin Farfade, Hamid Izadinia, Yannis Kalantidis, and Pierre Garrigues. Tag prediction at flickr: A view from the darkroom. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 376–384. ACM, 2017.

[21] Bastiaan J Boom, Phoenix X Huang, Jiyin He, and Robert B Fisher. Supporting ground-truth annotation of image datasets using clustering. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1542–1545. IEEE, 2012.

[22] Ali Borji. Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2019.

[23] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[24] Mohcine Boudhane and Benayad Nsiri. Underwater image processing method for fish localization and detection in submarine environment. *Journal of Visual Communication and Image Representation*, 39:226–238, 2016.

[25] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 7, 2017.

[26] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[27] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

[28] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1):321–357, June 2002.

[29] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016.

[30] Liang-Chieh Chen, Sanja Fidler, Alan L Yuille, and Raquel Urtasun. Beat the mturkers: Automatic image labeling from weak 3d supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3198–3205, 2014.

[31] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[32] Mickaël Chen, Thierry Artières, and Ludovic Denoyer. Unsupervised object segmentation by redrawing. *arXiv preprint arXiv:1905.13539*, 2019.

[33] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-supervised generative adversarial networks. *arXiv preprint arXiv:1811.11212*, 2018.

[34] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016.

[35] Yu-Sheng Chen, Yu-Ching Wang, Man-Hsin Kao, and Yung-Yu Chuang. Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6306–6314, 2018.

[36] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.

[37] LI Chongxuan, Taufik Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 4088–4098, 2017.

[38] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. Emnist: Extending mnist to handwritten letters. In *Neural Networks (IJCNN), 2017 International Joint Conference on*, pages 2921–2926. IEEE, 2017.

[39] Brendan Collins, Jia Deng, Kai Li, and Li Fei-Fei. Towards scalable dataset construction: An active learning approach. *ECCV*, 2008.

[40] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Ruslan R Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *Advances in Neural Information Processing Systems*, pages 6510–6520, 2017.

[41] Ayushman Dash, John Cristian Borges Gamboa, Sheraz Ahmed, Muhammad Zeshan Afzal, and Marcus Liwicki. Tac-gan-text conditioned auxiliary classifier generative adversarial network. *arXiv preprint arXiv:1703.06412*, 2017.

[42] Ramanath Datta, Paramita De Sekhar Mandal, and Bhabatosh Chanda. Detection and identification of logic gates from document images using mathematical morphology. In *Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), 2015 Fifth National Conference on*, pages 1–4. IEEE, 2015.

[43] Konstantinos Demertzis and Lazaros Iliadis. Detecting invasive species with a bio-inspired semi-supervised neurocomputing approach: the case of lagocephalus sceleratus. *Neural Computing and Applications*, pages 1–10.

[44] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[45] Jia Deng, Olga Russakovsky, Jonathan Krause, Michael S Bernstein, Alex Berg, and Li Fei-Fei. Scalable multi-label annotation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3099–3102. ACM, 2014.

[46] Emily Denton, Sam Gross, and Rob Fergus. Semi-supervised learning with context-conditional generative adversarial networks. *arXiv preprint arXiv:1611.06430 and under review by ICLR2017 conference*, 2016.

[47] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a[U+FFFC] laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015.

[48] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782,ICLR2017*, 2016.

[49] Qi Dong, Shaogang Gong, and Xiatian Zhu. Class rectification hard mining for imbalanced deep learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1851–1860, 2017.

[50] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 766–774, 2014.

[51] Georgios Douzas and Fernando Bacao. Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with Applications*, 91:464–471, 2018.

[52] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[53] Max Ehrlich, Timothy J Shields, Timur Almaev, and Mohamed R Amer. Facial attributes classification using multi-task representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 47–55, 2016.

[54] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2154, 2014.

[55] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

[56] Alberto FernáNdez, Victoria LóPez, Mikel Galar, MaríA José Del Jesus, and Francisco Herrera. Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches. *Knowledge-based systems*, 42:97–110, 2013.

[57] Maayan Frid-Adar, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Synthetic data augmentation using gan for improved liver lesion classification. *arXiv preprint arXiv:1801.02385*, 2018.

[58] Luoting Fu and Levent Burak Kara. From engineering diagrams to engineering models: Visual recognition and applications. *Computer-Aided Design*, 43(3):278–292, 2011.

[59] Jon Gauthier. Conditional generative adversarial nets for convolutional face generation. *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester*, 2014(5):2, 2014.

[60] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *ICLR2018*, 2018.

[61] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[62] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

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

[64] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

[65] Guillermo L Grinblat, Lucas C Uzal, and Pablo M Granitto. Class-splitting generative adversarial networks. *arXiv preprint arXiv:1709.07359*, 2017.

[66] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[67] Swaminathan Gurumurthy, Ravi Kiran Sarvadevabhatla, and V Babu Radhakrishnan. Deligan: Generative adversarial networks for diverse and limited data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, 2017.

[68] Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361, 2012.

[69] Mohamed Farouk Abdel Hady and Friedhelm Schwenker. Semi-supervised learning. In *Handbook on Neural Information Processing*, pages 215–239. Springer, 2013.

[70] Emily M Hand, Carlos D Castillo, and Rama Chellappa. Doing the best we can with what we have: Multi-label balancing with selective learning for attribute prediction. In *AAAI*, 2018.

[71] Emily M Hand and Rama Chellappa. Attributes for improved attributes: A multi-task network utilizing implicit and explicit relationships for facial attribute classification. In *AAAI*, pages 4068–4074, 2017.

[72] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[73] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[74] Xiangteng He and Yuxin Peng. Weakly supervised learning of part selection model with spatial constraints for fine-grained image classification. In *AAAI*, 2017.

[75] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6629–6640, 2017.

[76] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[77] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[78] Ekram Hossain, SM Shaiful Alam, Amin Ahsan Ali, and M Ashraful Amin. Fish activity tracking and species identification in underwater video. In *Informatics, Electronics and Vision (ICIEV), 2016 5th International Conference on*, pages 62–66. IEEE, 2016.

[79] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5375–5384, 2016.

[80] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, volume 1, page 3, 2017.

[81] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. 2008.

[82] Mark J Huiskes and Michael S Lew. The mir flickr retrieval evaluation. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, pages 39–43. ACM, 2008.

[83] Daniel Jiwoong Im, Chris Dongjoo Kim, Hui Jiang, and Roland Memisevic. Generating images with recurrent adversarial networks. *arXiv preprint arXiv:1602.05110*, 2016.

[84] Hiroshi Inoue. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.

[85] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[86] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[87] Pamela Johnston and Eyad Elyan. A review of digital video tampering: from simple editing to full synthesis. *Digital Investigation*, 2019.

[88] Piotr Dollar Ross Girshick Kaiming He, Georgia Gkioxari. Mask r-cnn. *arXiv preprint arXiv:1703.06870*, 2017.

[89] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196 ICLR2018*, 2017.

[90] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1857–1865. JMLR. org, 2017.

[91] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv*, 2014.

[92] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[93] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[94] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[95] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

[96] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, 2017.

[97] Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

[98] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[99] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[100] Abhishek Kumar, Prasanna Sattigeri, and P Thomas Fletcher. Improved semi-supervised learning with gans using manifold invariances. *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[101] Yevhen Kuznietsov, Jorg Stuckler, and Bastian Leibe. Semi-supervised deep learning for monocular depth map prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6647–6655, 2017.

[102] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[103] Ekaterina Lantsova, Tatiana Voitiuk, Tatiana Zudilova, and Arto Kaarna. Using low-quality video sequences for fish detection and tracking. In *SAI Computing Conference (SAI), 2016*, pages 426–433. IEEE, 2016.

[104] Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404, 1990.

[105] Yann LeCun, Corinna Cortes, and Christopher JC Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

[106] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013.

[107] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616. ACM, 2009.

[108] Minhyeok Lee and Junhee Seok. Controllable generative adversarial network. *arXiv preprint arXiv:1708.00598*, 2017.

[109] Erich L Lehmann and Joseph P Romano. *Testing statistical hypotheses*. Springer Science & Business Media, 2006.

[110] Xiu Li, Min Shang, Hongwei Qin, and Liansheng Chen. Fast accurate fish detection and recognition of underwater images with fast r-cnn. In *OCEANS 2015-MTS/IEEE Washington*, pages 1–5. IEEE, 2015.

[111] Min Lin. Softmax gan. *arXiv preprint arXiv:1704.06191(submitted to nips2017)*, 2017.

[112] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[113] Dawei Liu and Zhihua Yu. A computationally efficient algorithm for large scale near-duplicate video detection. In *International conference on multimedia modeling*, pages 481–490. Springer, 2015.

[114] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 469–477, 2016.

[115] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[116] Xinhua Liu, Chengjuan Xie, Hailan Kuang, and Xiaolin Ma. Face aging simulation with deep convolutional generative adversarial networks. In *Measuring Technology and Mechatronics Automation (ICMTMA), 2018 10th International Conference on*, pages 220–224. IEEE, 2018.

[117] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3730–3738, 2015.

[118] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[119] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[120] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 700–709. Curran Associates, Inc., 2018.

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

[122] Giovanni Mariani, Florian Scheidegger, Roxana Istrate, Costas Bekas, and Cristiano Malossi. Bagan: Data augmentation with balancing gan. *arXiv preprint arXiv:1803.09655*, 2018.

[123] Richard T. Marriot, Sami Romdhani, and Liming Chen. Intra-class variation isolation in conditional gans. *arXiv preprint arXiv:1811.11296*, 2018.

[124] J Matai, R Kastner, GR Cutter Jr, and DA Demer. Automated techniques for detection and recognition of fishes using computer vision algorithms. In *NOAA Technical Memorandum NMFS-F/SPO-121, Report of the National Marine Fisheries Service Automated Image Processing Workshop, Williams K., Rooper C., Harms J., Eds., Seattle, Washington (September 4–7 2010)*, 2010.

[125] Ajkel Mino and Gerasimos Spanakis. Logan: Generating logos with a generative adversarial neural network conditioned on color. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 965–970. IEEE, 2018.

[126] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[127] Ishan Misra, Abhinav Shrivastava, and Martial Hebert. Watch and learn: Semi-supervised learning for object detectors from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3593–3602, 2015.

[128] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957 and ICLR2018*, 2018.

[129] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. *arXiv preprint arXiv:1802.05637*, 2018.

[130] Carlos Francisco Moreno-García, Eyad Elyan, and Chrisina Jayne. Heuristics-based detection to improve text/graphics segmentation in complex engineering drawings. In *International Conference on Engineering Applications of Neural Networks*, pages 87–98. Springer, 2017.

[131] Carlos Francisco Moreno-García, Eyad Elyan, and Chrisina Jayne. New trends on digitisation of complex engineering drawings. *Neural computing and applications*, pages 1–18, 2018.

[132] Saeid Motiian, Quinn Jones, Seyed Iranmanesh, and Gianfranco Doretto. Few-shot adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 6673–6683, 2017.

[133] Youssef Mroueh and Tom Sercu. Fisher gan. In *Advances in Neural Information Processing Systems*, pages 2513–2523, 2017.

[134] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011.

[135] Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *Advances in Neural Information Processing Systems*, pages 3387–3395, 2016.

[136] Tu Nguyen, Trung Le, Hung Vu, and Dinh Phung. Dual discriminator generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2670–2680, 2017.

[137] Dong Nie, Roger Trullo, Jun Lian, Caroline Petitjean, Su Ruan, Qian Wang, and Dinggang Shen. Medical image synthesis with context-aware generative adversarial networks, 2017.

[138] Augustus Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.

[139] Augustus Odena, Jacob Buckman, Catherine Olsson, Tom B Brown, Christopher Olah, Colin Raffel, and Ian Goodfellow. Is generator conditioning causally related to gan performance? *arXiv preprint arXiv:1802.08768*, 2018.

[140] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. *International conference on machine learning,page 2642-2651*, 70:2642–2651, AUG 2017.

[141] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *British Machine Vision Conference*, volume 1, page 6, 2015.

[142] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016.

[143] Anthony Perez, Swetava Ganguli, Stefano Ermon, George Azzari, Marshall Burke, and David Lobell. Semi-supervised multitask learning on multispectral satellite images using wasserstein generative adversarial networks (gans) for predicting poverty. *stanford cs321n, not published yet.*

[144] The-Anh Pham, Mathieu Delalandre, Sabine Barrat, and Jean-Yves Ramel. Accurate junction detection and characterization in line-drawing images. *Pattern Recognition*, 47(1):282–295, 2014.

[145] Guo-Jun Qi. Loss-sensitive generative adversarial networks on lipschitz densities. *arXiv preprint arXiv:1701.06264*, 2017.

[146] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[147] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766. ACM, 2007.

[148] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.

[149] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[150] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 3, 2016.

[151] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. In *Advances in Neural Information Processing Systems*, pages 2018–2028, 2017.

[152] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 41(4):801–814, 2019.

[153] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1):157–173, 2008.

[154] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016.

[155] Swami Sankaranarayanan, Yogesh Balaji, Carlos D Castillo, and Rama Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. *ArXiv e-prints, abs/1704.01705*, 2017.

[156] Kevin Schawinski, Ce Zhang, Hantian Zhang, Lucas Fowler, and Gokula Krishnan Santhanam. Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit. *Monthly Notices of the Royal Astronomical Society: Letters*, 467(1):L110–L114, 2017.

[157] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[158] Alexander K Seewald. Digits-a dataset for handwritten digit recognition. *Austrian Research Institut for Artificial Intelligence Technical Report, Vienna (Austria)*, 2005.

[159] Insaf Setitra and Slimane Larabi. Sift descriptor for binary shape discrimination, classification and matching. In *International Conference on Computer Analysis of Images and Patterns*, pages 489–500. Springer, 2015.

[160] Hoo-Chang Shin, Holger R Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.

[161] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. How good is my gan? In *ECCV 2018-European Conference on Computer Vision*, pages 1–20, 2018.

[162] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, volume 2, page 5, 2017.

[163] Rui Shu, Hung Bui, and Stefano Ermon. Ac-gan learns a biased distribution. *workshop paper :Second workshop on Bayesian Deep Learning (NIPS 2017), Long Beach, CA, USA.*, 2017.

[164] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[165] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4080–4090, 2017.

[166] Jingkuan Song, Jingqiu Zhang, Lianli Gao, Xianglong Liu, and Heng Tao Shen. Dual conditional gans for face aging and rejuvenation. In *IJCAI*, pages 899–905, 2018.

[167] Nasim Souly, Concetto Spampinato, and Mubarak Shah. Semi supervised semantic segmentation using generative adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5688–5696, 2017.

[168] Concetto Spampinato, Daniela Giordano, Roberto Di Salvo, Yun-Heh Jessica Chen-Burger, Robert Bob Fisher, and Gayathri Nadarajan. Automatic fish classification for underwater species behavior understanding. In *Proceedings of the first ACM international workshop on Analysis and retrieval of tracked events and motion in imagery streams*, pages 45–50. ACM, 2010.

[169] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

[170] Kumar Sricharan, Raja Bala, Matthew Shreve, Hui Ding, Kumar Saketh, and Jin Sun. Semi-supervised conditional gans. *arXiv preprint arXiv:1708.05789*, 2017.

[171] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems*, pages 3308–3318, 2017.

[172] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.

[173] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. In *Advances in Neural Information Processing Systems*, pages 2553–2561, 2013.

[174] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[175] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[176] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

[177] Harri Valpola. From neural pca to deep unsupervised learning. *Advances in Independent Component Analysis and Learning Machines*, pages 143–171, 2015.

[178] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016.

[179] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[180] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[181] Lipeng Wan, Jun Wan, Yi Jin, Zichang Tan, Stan Z Li, et al. Fine-grained multi-attribute adversarial learning for face generation of age, gender and ethnicity, 2018.

[182] Chaoyue Wang, Chang Xu, Xin Yao, and Dacheng Tao. Evolutionary generative adversarial networks. *arXiv preprint arXiv:1803.00657*, 2018.

[183] Shoujin Wang, Wei Liu, Jia Wu, Longbing Cao, Qinxue Meng, and Paul J Kennedy. Training deep neural networks on imbalanced data sets. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 4368–4374. IEEE, 2016.

[184] Xiaolong Wang and Abhinav Gupta. Generative image modeling using style and structure adversarial networks. In *European Conference on Computer Vision*, pages 318–335. Springer, 2016.

[185] Zongwei Wang, Xu Tang, Weixin Luo, and Shenghua Gao. Face aging with identity-preserved conditional generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7939–7947, 2018.

[186] Yuanwang Wei, Zhijiang Zhang, Wei Shen, Dan Zeng, Mei Fang, and Shifu Zhou. Text detection in scene images based on exhaustive segmentation. *Signal Processing: Image Communication*, 50:1–8, 2017.

[187] Lior Wolf, Tal Hassner, and Itay Maoz. *Face recognition in unconstrained videos with matched background similarity*. IEEE, 2011.

[188] Sitao Xiang and Hao Li. On the effects of batch and weight normalization in generative adversarial networks. *stat*, 1050:22, 2017.

[189] Zhongwen Xu, Linchao Zhu, and Yi Yang. Few-shot object recognition from machine-labeled web images. In *Computer Vision and Pattern Recognition*, 2017.

[190] Shengke Xue and Xinyu Jin. Robust classwise and projective low-rank representation for image classification. *Signal, Image and Video Processing*, 2018.

[191] Wankou Yang, Karl Ricanek, and Fumin Shen. Image classification using local linear regression. *Neural Computing and Applications*, 25(7-8):1913–1920, 2014.

[192] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE international conference on computer vision*, pages 2849–2857, 2017.

[193] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[194] Yuhong Yu, Ashok Samal, and Sharad C Seth. A system for recognizing a large class of engineering drawings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8):868–890, 1997.

[195] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[196] Dengsheng Zhang, Md Monirul Islam, and Guojun Lu. A review on automatic image annotation techniques. *Pattern Recognition*, 45(1):346–362, 2012.

[197] Han Zhang, Ian Goodfellow, dimitris Metaxas, and Augusto Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.0818v1*, 2018.

[198] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5907–5915, 2017.

[199] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[200] Kaipeng Zhang, Lianzhi Tan, Zhifeng Li, and Yu Qiao. Gender and smile classification using deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 34–38, 2016.

[201] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.

[202] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.

[203] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[204] Xinyue Zhu, Yifan Liu, and Zengchang Qin. Data augmentation in classification using gan. *arXiv preprint arXiv:1711.00648*, 2017.

Appendix A

List of Publications

The contributions from this work lead to the following publications.

A.1 Journal Article

- Ali-Gombe, Adamu, & Elyan, Eyad (2019, July). MFC-GAN: Class-imbalanced Dataset Classification using Multiple Fake Class Generative Adversarial Networks. In 2019 NeuroComputing journal . Elsevier. DOI: https://doi.org/10.1016/j.neucom.2019.06.043

A.2 Conference Proceedings

- Ali-Gombe, Adamu, Eyad Elyan, and Chrisina Jayne. Fish classification in context of noisy images. 18th International Conference on Engineering Applications of Neural Networks. Springer, Cham, 2017. DOI: https://doi.org/10.1007/978-3-319-65172-9_19

- Ali-Gombe, A., Elyan, E., Savoye, Y., & Jayne, C. (2018, July). Few-shot Classifier GAN. In 2018 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE. DOI: https://doi.org/10.1109/IJCNN.2018.8489387

- Ali-Gombe, Adamu, Elyan, Eyad & Jayne, Chrisina (2019, July). Multiple Fake Classes GAN for Data Augmentation in Face Image Dataset. In 2019 International Joint Conference on Neural Networks (IJCNN) . IEEE.

Appendix B

Supplementary Results

B.1 Goatee Attribute Classification Results

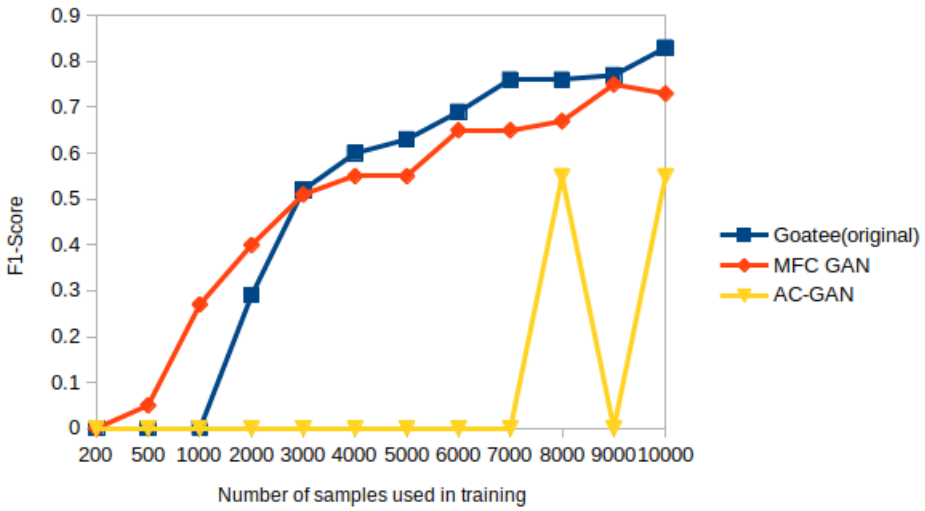
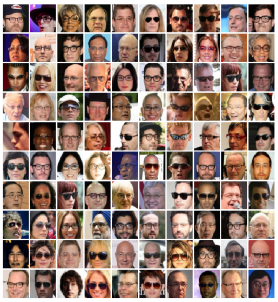
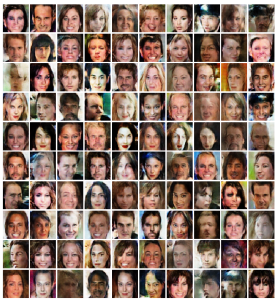


Figure B.1: Goatee classification performances

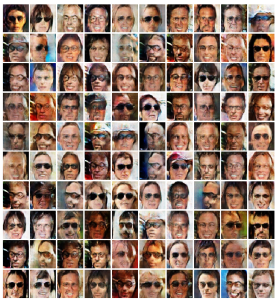
B.2 MFC-GAN Results on Facial Glasses Attribute



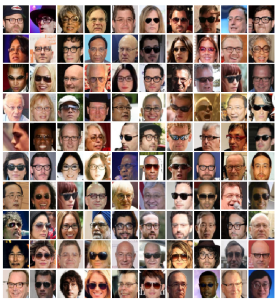
Original samples



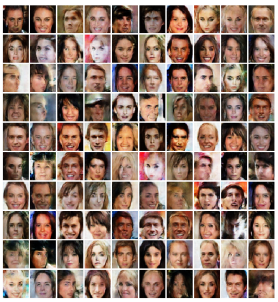
AC-GAN 2h



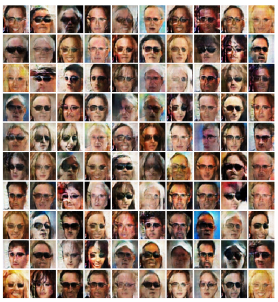
MFC-GAN 2h



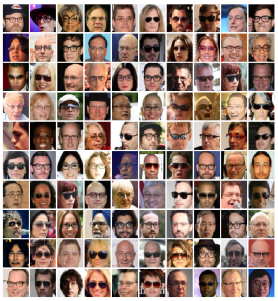
Original samples



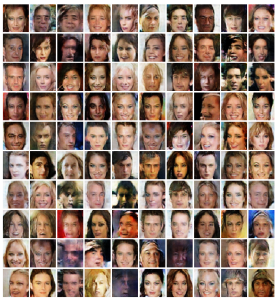
AC-GAN 5h



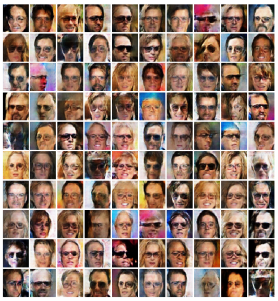
MFC-GAN 5h



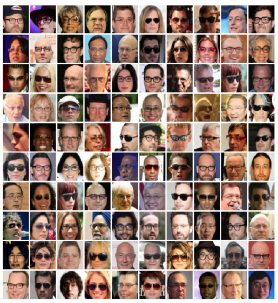
Original samples



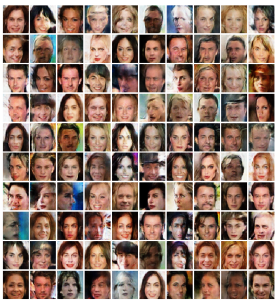
AC-GAN 1k



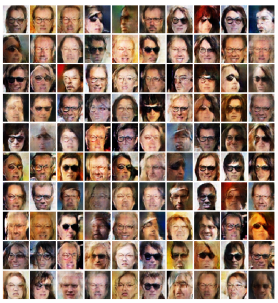
MFC-GAN 1k



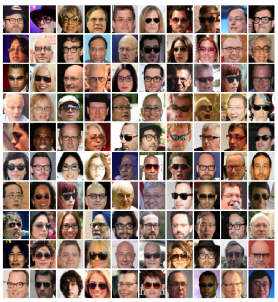
Original samples



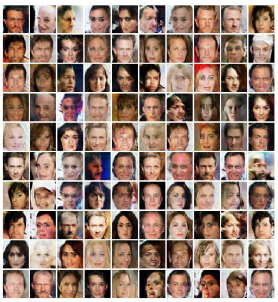
AC-GAN 2k



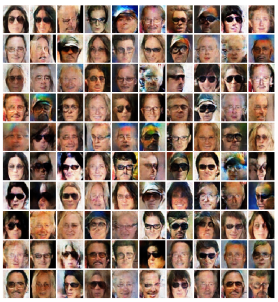
MFC-GAN 2k



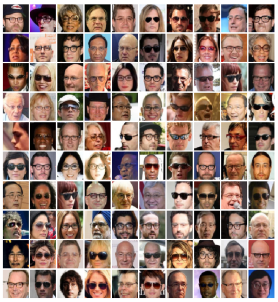
Original samples



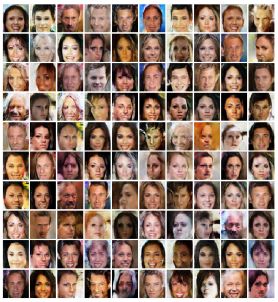
AC-GAN 3k



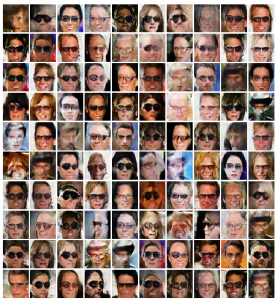
MFC-GAN 3k



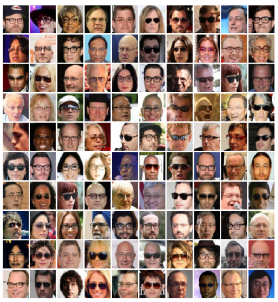
Original samples



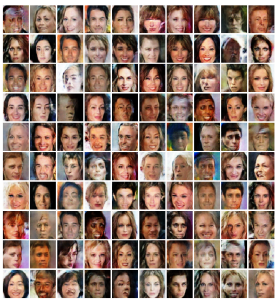
AC-GAN 4k



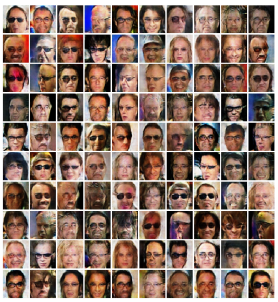
MFC-GAN 4k



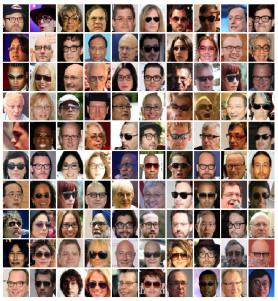
Original samples



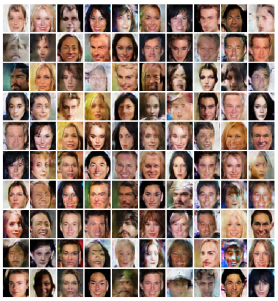
AC-GAN 5k



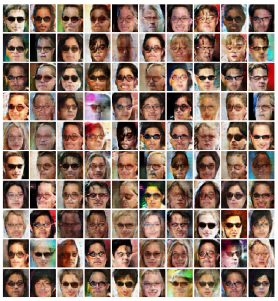
MFC-GAN 5k



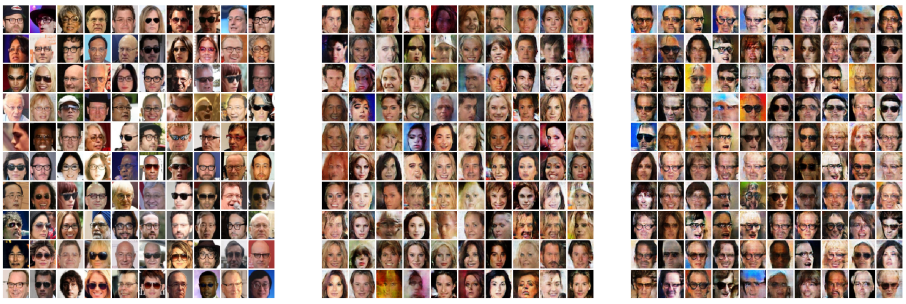
Original samples



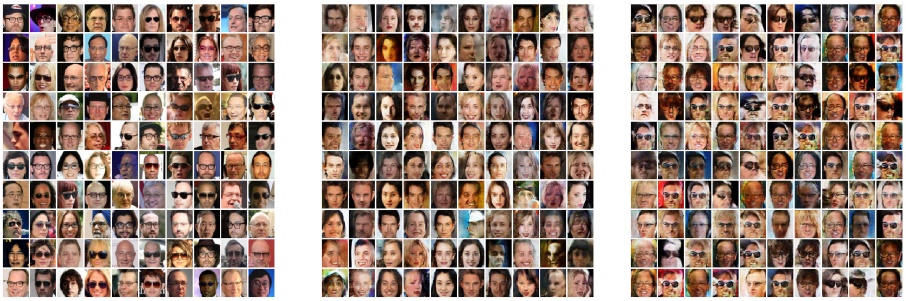
AC-GAN 6k



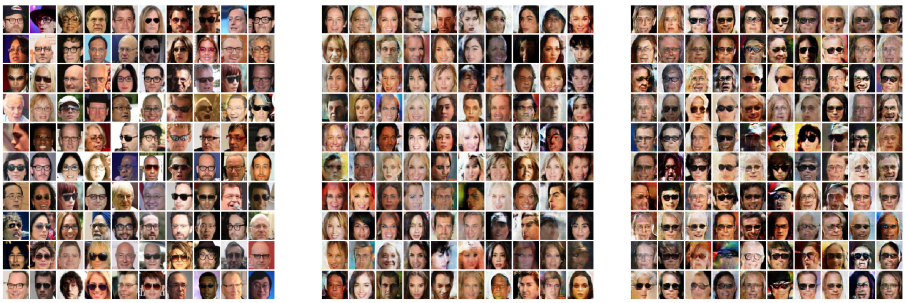
MFC-GAN 6k



Original samples AC-GAN 7k MFC-GAN 7k



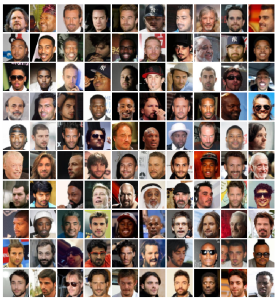
Original samples AC-GAN 8k MFC-GAN 8k



Original samples AC-GAN 9k MFC-GAN 9k

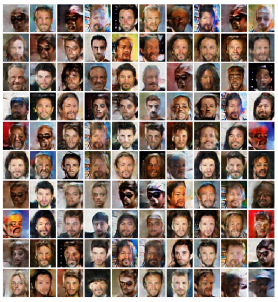
Figure B.2: *Comparing samples AC-GAN and MFC-GAN samples from experiments with varying number of glasses instances in the.*

B.3 MFC-GAN Results on Face Goatee Attributes

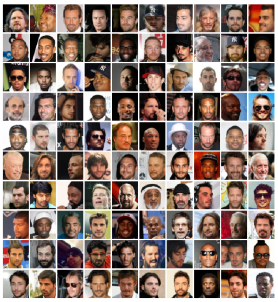


Original samples

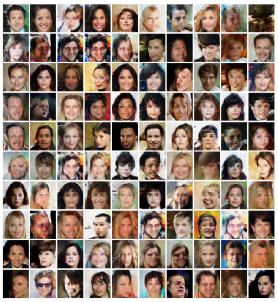
AC-GAN 2h



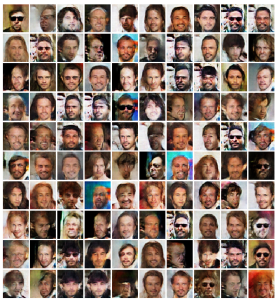
MFC-GAN 2h



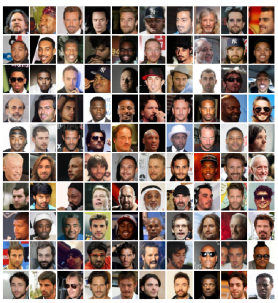
Original samples



AC-GAN 5h

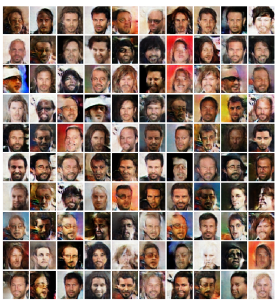


MFC-GAN 5h

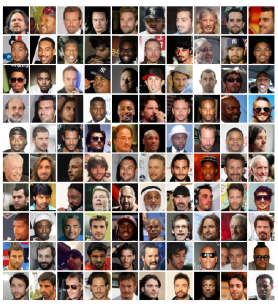


Original samples

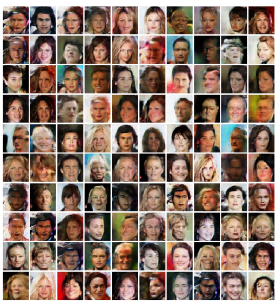
AC-GAN 1k



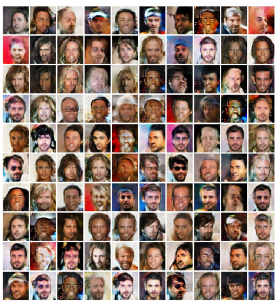
MFC-GAN 1k



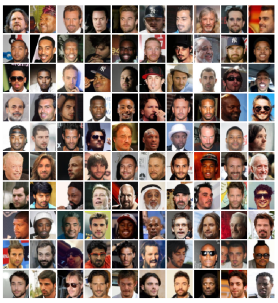
Original samples



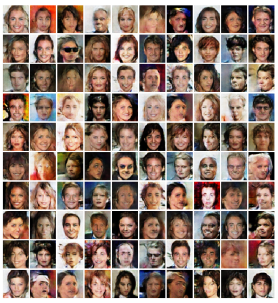
AC-GAN 2k



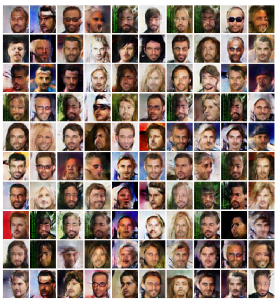
MFC-GAN 2k



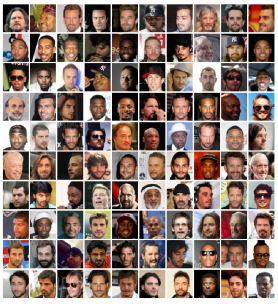
Original samples



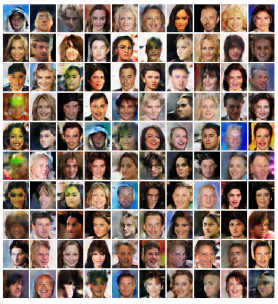
AC-GAN 3k



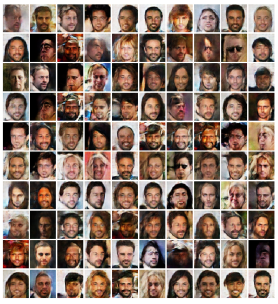
MFC-GAN 3k



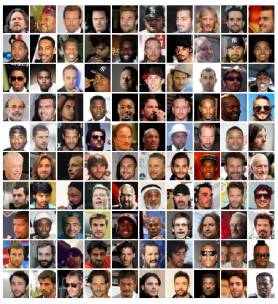
Original samples



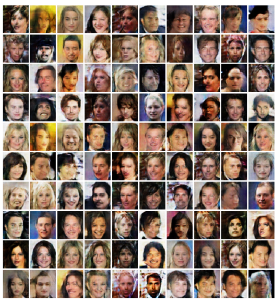
AC-GAN 4k



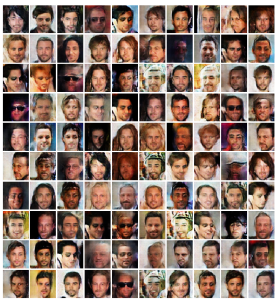
MFC-GAN 4k



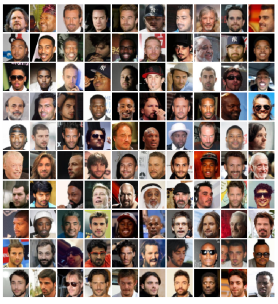
Original samples



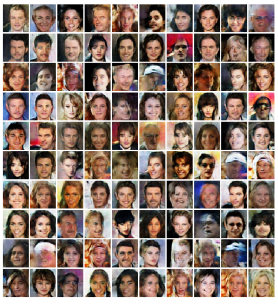
AC-GAN 5k



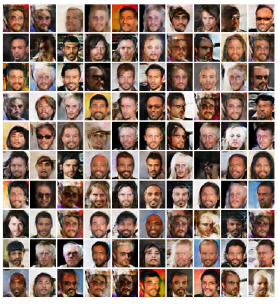
MFC-GAN 5k



Original samples



AC-GAN 6k



MFC-GAN 6k

Figure B.3: *Comparing samples AC-GAN and MFC-GAN samples from experiments with varying number of goatee instances in the.*

Table B.1: *Mean Freschet Inception Distance (FID) of generated image from experiments on different number of samples with goatee.*

Number of samples	AC-GAN	MFC-GAN
200	71.45	**65.64**
500	68.15	**65.80**
1000	69.14	**69.67**
2000	66.62	**66.79**
3000	58.72	62.56
4000	69.48	**62.16**
5000	57.06	61.25
6000	70.55	71.35
7000	92.97	**61.37**
8000	73.29	**68.51**
9000	60.02	**59.69**
10000	70.44	**61.69**

Table B.2: *True positive rate report on goatee attribute classification and highlighted in bold are the instances where MFC-GAN performed better than both the baseline and AC-GAN.*

Number of samples	Baseline	AC-GAN	MFC-GAN
200	0.0	0.0	0.0
500	0.0	0.0	**0.03**
1000	0.0	0.0	**0.16**
2000	0.17	0.0	**0.25**
3000	0.35	0.0	0.34
4000	0.44	0.0	0.38
5000	0.47	0.0	0.39
6000	0.53	0.0	0.49
7000	0.62	0.0	0.49
8000	0.62	0.38	0.51
9000	0.63	0.0	0.60
10000	0.72	0.38	0.58

Appendix C

List of Algorithms