Executive Summary

This body of work is split into two sections to reflect the requirements of a SIPS thesis. The first section contains the main body of work, the thesis itself. This thesis focuses on reducing the knowledge barrier for implementing machine learning solutions in the industry. This was achieved by implementing and evaluating a novel two-step transductive transfer learning pipeline embedded into a proposed "ML-as-a-Service" platform. This pipeline is composed firstly of a layer which takes a domain adaptation approach in order to create large data sets from limited reference data provided by the user. This is followed by a retraining layer which after retraining a pretrained model, produces an optimal classifier for that's specific to the environment the model will be deployed in. This model has a higher performance than a traditionally trained model, that required more data than what the user could initially provide.

The second part of this work covers the other requirements for a SIPS thesis including two case studies, a WHS report and a project experience report. The case studies cover two subjects that have been replaced by this industry thesis, Professional Engineering 2 and Introduction to Biomechatronics. While unable to fulfil the learning attributes of these subjects through the traditional means (i.e. Accenture does not have any biomechatronic accounts), the learning outcomes were instead fulfilled by other projects that I was tasked with while working at Accenture of equal relevance.

UNIVERSITY OF SYDNEY

THESIS

Using Transfer Learning to Produce a 'Machine Learning-as-a-Service' Platform.

Author:
Tys LOWDE

Supervisors:
Dr. Mitch BRYSON
Luke HIGGINS

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Engineering(Mechatronics)

in the

School of Aerospace, Mechanical and Mechatronic Engineering

January 21, 2019

Declaration of Authorship

I, Tys LOWDE, declare that this thesis titled, “Using Transfer Learning to Produce a 'Machine Learning-as-a-Service' Platform.” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a degree at this University.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. This includes the list of software packages included in Chapter 3.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.
- I consulted with my supervisor Mitch Bryson at the beginning of this project, offering advice on a satisfactory direction.
- Luke Higgins provided both direction and support throughout this thesis.
- The automated selection of hyper-parameters for machine learning models used in this thesis is based strongly on code and principles developed in Max Schultz's thesis "Optimal Control for the Automated Design of machine learning Models".

Signed:

Date:

Acknowledgements

I would first like to acknowledge my parents Simon and Melanie. Without their love and unconditional support none of this would exist.

Thank you Luke for providing the opportunity and platform to do this thesis.

To HK, without you this thesis would be half of what it is. I owe you a debt you can collect anytime.

Thank you to everyone who I worked with at Accenture while doing this thesis; Alex, Rumanshu, Riya, Jonas and Tarryn. You were the best people I've ever had the pleasure of working with.

To Max, Josh, Jack and the friends that are and have been, without you I would have nothing.

And finally to my love Mandy. Thank you.

Abstract

Using Transfer Learning to Produce a 'Machine Learning-as-a-Service' Platform.

by Tys LOWDE

This thesis proposes a novel two-step transductive transfer learning pipeline, composed of a domain adaptation and retraining layer. The pipeline addresses a prohibitive prerequisite of traditional machine learning approaches, by reducing the amount of data from the targeted environment (domain) that the user will be required to provide. Alongside solving this problem, this body of work lays the foundations of an "ML-as-a-Service" platform capable of significantly reducing the barrier to entry for non-experts.

The data scarcity problem is addressed in the pipeline's first layer, using generative adversarial networks (GANs) to transform an extensive data set using a similar, but much smaller representative data set. These GANs take a feature representation approach to solving this domain adaptation problem. Because of the fundamental limitations of GANs, it was necessary to employ two types of GANs (*pixel-wise* and *SeqGAN*). These GANs were improved upon and then applied to the domain adaptation of continuous and discrete data types respectively. The output from this first layer is a synthetic data set that is used in the second layer to retrain a pre-trained model. This retraining approach applies a novel re-imagining of an algorithm initially used for *FreezeOut* training. An annealing equation freezes the layers at different rates in the retraining process, producing models that are more robust, due to having retained some of the relevant knowledge from the previous domain.

Across all experiments, the use of the retraining layer in the pipeline lead to a $0-7\%$ F1 score increase. This improvement has an inverse relationship to the success of the domain adaptation layer. For continuous data types, among other successful experiments, this pipeline beat the current MNIST to MNISTM accuracy benchmark by 1.4%. However, for discrete data types, *SeqGAN*s failed to learn the target domain and create new synthetic media. This failing, also seen in some experiments with the *pixel-wise* approach, is being proposed to be due to a lack of range and variance in the provided target data set. Using variance as a metric to gauge the success of GANs when completing complex domain adaptation tasks, is a significant finding that can be used in future applications.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Topic Motivation

Accenture is a world leading software consultancy company, in an industry that is rapidly adopting machine learning (ML) solutions to replace or enhance tasks that are either currently fulfilled by humans or not possible at all. However, when Accenture delivers these ML-based applications, they can only be applied in extremely narrow environments. The machine learning components within the applications, must be retrained when new use cases or environments are identified, with the prior learning and resources from previous work failing to be transferred. These retraining processes require large amounts of data from the new environment. This typically requires an extensive data collection process to be undertaken. A process that is overseen by experts in both the data science and machine learning fields. These resources are typically unavailable to clients within their own companies.

These constraints are not exclusive to Accenture's operations. The current limitations that are imposed by the traditional machine learning pipeline have lead to other software giants such as Microsoft and Google to investigate the possibility of releasing 'inexpert friendly' ML services. If these new platforms are not built, companies are likely to move away from ML-based services, in favour of other solutions that do not have high data or specialist knowledge requirements. Accenture and other software companies must begin to build their products to accommodate non-experts or risk losing having industries lose interest in the machine learning space completely.

1.2 Problem Statement

The proposed platform will be fully accessible to non-experts and remove the onus on the user to produce large amounts of new data to define their new environment. In order for this platform to become the basis of Accenture's new approach to machine learning, the approach must be as generic as possible. Therefore, the platform must be data agnostic, modular and make use of data sources from marginally different environments.

This thesis will propose a pipeline which will create classifiers that are optimal in the target environment ($\mathcal{D}_T$) from limited data sets. The data provided by the user will need to have the same feature space as one of the larger data sets already sourceable by the platform ($\mathcal{X}_S = \mathcal{X}_T$). The distribution of that data will be different ($P(X)_S \neq P(X)_T$) and correcting this dissonance will be the first step of the two-step pipeline. These requirements are those of a transductive transfer learning problem ($\mathcal{D}_T \approx \mathcal{D}_S$ where $\mathcal{D}_S = \{\mathcal{X}_S, P(X_S)\}$).

Two distinct use cases will be conducted to prove the validity and optimality of the proposed pipeline, a natural language processing and an image recognition problem. These use cases will evaluate a two-step transductive transfer learning pipeline. The suitability of this pipeline will be judged using metrics common to the ML space F1 Score, Precision and Recall.

1.3 Differentiating Between the Platform and the Pipeline

There are two core elements to this body of work, an "ML-as-a-Service" platform and a "Two-Step Transductive Transfer Learning" pipeline. The purpose of this thesis is to produce an "ML-as-a-Service" platform. However, major components of such a platform exist as an API development exercise and lack novel or additive achievement. Instead, the core contribution towards this platform, that this body of work will put forward is the "Two-Step Transductive Transfer Learning" pipeline. This pipeline will form the backbone of the proposed "ML-as-a-Service" platform.

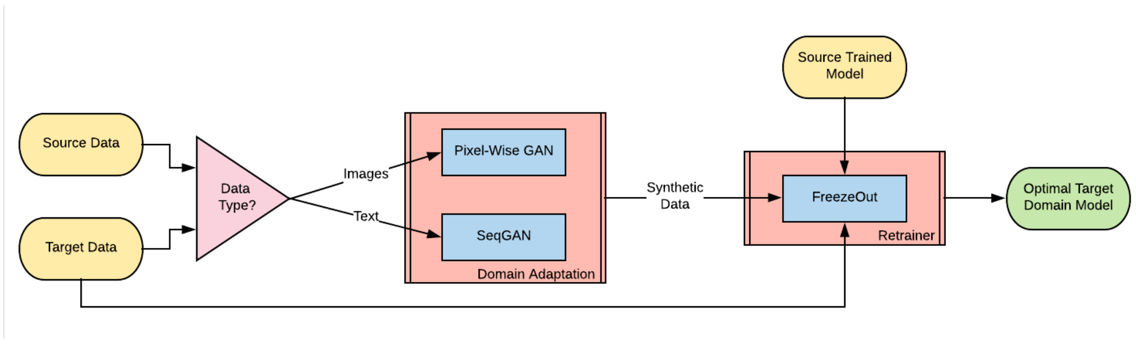


FIGURE 1.1: 2-Step Transductive Transfer Learning Pipeline

In order to build and test this pipeline, modules from the perspective "ML-as-a-Service" platform were required. These modules were researched, built and tested as a part of the development of this thesis. These tests were conducted to ensure the validity of the pipeline's results by first confirming the validity of the modules that support it (experiments are located in the appendix). The minimal platform that was built is by no means a deployable platform, but it does form the basis of one that could be built for industrial application.

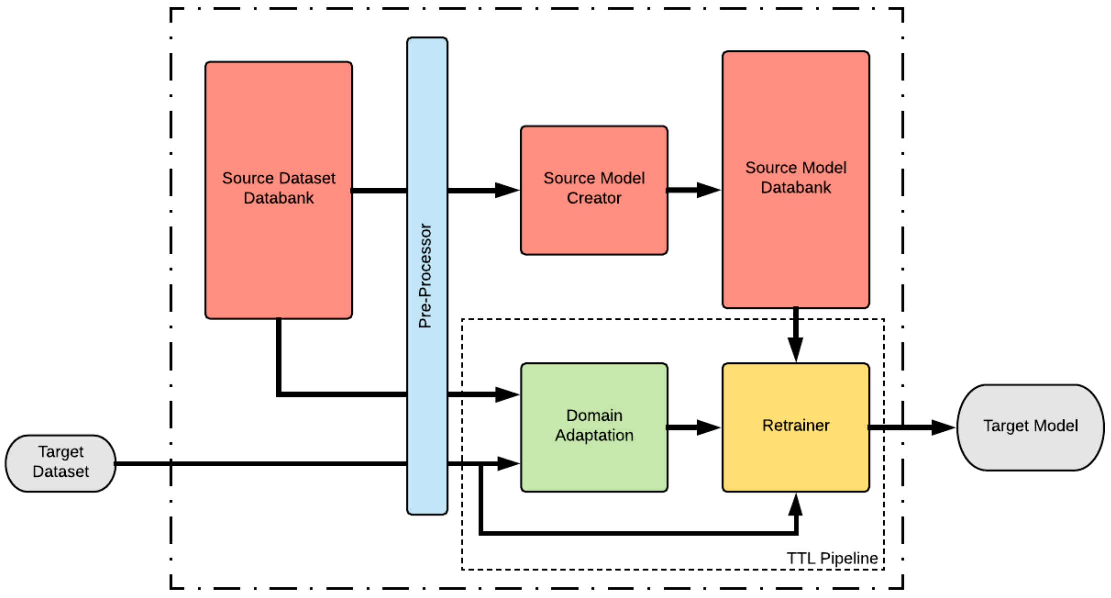


FIGURE 1.2: Minimal "ML-as-a-Service" Platform

A more in-depth review of this platform and how the pipeline will satisfy its requirements will be located in Chapter 8.

1.4 Thesis Outline

This thesis will include all facets of the creation, integration and testing of a novel two-step transductive transfer learning pipeline. The breakdown of this thesis follows a chronological process of research, module design, modular testing, integration, integration testing through two use cases:

- **Literary Review** - *Chapter 2*: Contains a review of prior works in order to achieve an understanding of the historical and current day context of machine learning. Also covered is an evaluation of the work done in the areas of transfer learning, image recognition and natural language processing.
- **Software and Hardware** - *Chapter 3*: An overview of the software packages that were used in this thesis, either directly or as a reference. Also is a description of the hardware that was used to test these solutions.
- **Module Design** - *Chapter 4 & 5*: An overview of the platform's modules; including their function, the approach that was taken to achieve their purpose and the software and code base particular to that approach.
- **Module Experiments** - *Chapter 6 & 7*: Experimental methods, results and evaluations of the approaches taken for the modules directly related to the two-step transductive transfer learning pipeline. These experiments will ensure that the individual purpose of each module is met within the context of the wider proposed platform. Only modules that were novel and directly contained within the pipeline were evaluated within this section.
- **System Integration** - *Chapter 8*: A detailed review of the platform after integration of the modules, with careful consideration from the perspective of the core persona (the inexpert user).
- **Use Case 1: Image Recognition** - *Chapters 9 & 10*: Proof of the platform's performance for an image recognition use case. This use case covers a challenging real world problem using crowded image data with a progressively blurred, sepia overlay. This section is split into two chapters to distinguish the experiment and the discussion of the pipeline's performance in the image recognition space.
- **Use Case 2: Sentiment Analysis** - *Chapters 11 & 12*: Proof of the platform's performance for a NLP sentiment analysis use case. This use case used real ticket data from Accenture to evaluate the two-step's pipeline to create synthetic tickets, to create a better classifier for ticket sentiment. This section is

split into two chapters to distinguish the experiment and the discussion of the pipeline's performance in the NLP space.

- **Conclusion** - *Chapter 13*: Provide a brief analysis of the contributions of this thesis, work that can be derived from its findings and a summary of the success of the pipeline.

Chapter 2

Literature Review

When considering what a "ML-as-a-Service" platform is and why it is required, an understanding of the technical and industry context that it is designed for needs to be investigated. This literature review will evaluate the currently available technologies, how they are being applied and have been applied in the past in order to build an "ML-as-a-Service" platform that is capable of removing the barriers to entry for the main user of this platform, the non-expert. In order to do this, several questions need to be answered:

- What are the limitations of the current traditional machine learning pipeline and what effect do they have on the industry?
- How severe are these limitations given the historical context of ML?
- What advancements in the ML field including model types, transformation, training and optimisation algorithms and data augmentation approaches can be applied to overcome these limitations?
- How will this improved pipeline be applied to common use cases the industry will likely require.

2.1 Machine Learning in the Industry

Machine learning is making its way into almost all industries due to its ability to improve the quality of work and reduce the cost and manpower associated with current jobs [1]. As well as opening up new fields particularly in the areas of human computer interaction (HCI) [2], complex predictive analysis [3] and the medical industry [4]. Unfortunately there exists significant misconceptions about the abilities of ML that have led to the over-estimation of the ability for machine learning applications [5].

By far, the greatest danger of Artificial Intelligence is that people conclude too early that they understand it.

– *Yukdowsky 2008* [6]

The belief that ML/AI is capable of completely replacing humans in the workforce is a misconception [7] [8]. That doesn't consider that current ML implementations, whilst impressive, can only operate within the environments they were originally trained within.

Despite this, machine learning is a huge market disruption. It is predicted by the Bank of America that the AI market will grow to $157.2 billion by 2020, increasing current productivity in all industries by 35% [9]. With such interest, it's imperitive that growth in this market is not stagnated by poor application due to misinformation.

2.1.1 Machine Learning's Troubled History of Adoption

Understanding the history of machine learning is important when identifying and solving the issues currently plaguing ML's adoption in today's markets. After it's inception in the 1950's, early ML/AI had reasonable growth up until the 70's when the first "AI Winter" occured [10] . Which was brought about by a drastic drop in funding from major investors including DARPA and notable US universities. The drop in funding was accredited to slow results in a poorly understood market [11]. Investors in AI/ML companies and researchers noted that ML/AI at the time had poor performance and was plagued by technical errors. Eventually the pitching of a product that did not meet the reality of what it was evenutally caught up with the industry.

This cycle of over-promising and under-delivering is known as a "hype-cycle". However, unlike other consumer based industries (e.g. the video games industry's "ATARI crash" in the 80's [12]) the ML/AI industry appears to have an abnormal frequency of these hype-cycles. "AI Winters" occurred twice again, the most recent being the during the period where the rule-based machines known as 'Expert Systems' failed to meet market expectations [13].

Today's current explosion of interest and rapid commercial adoption of AI/ML has come from the inception of data-centric machine learning approaches. This form of ML uses extensive, labelled and maintained databases such as *ImageNet* [14], that have became accessible as the internet came online. Acting as a catalyst, these data sets have been attributed to dramatic improvements in the results of bench-marking competitions, such as the "Large Scale Visual Recognition Challenge" [14].

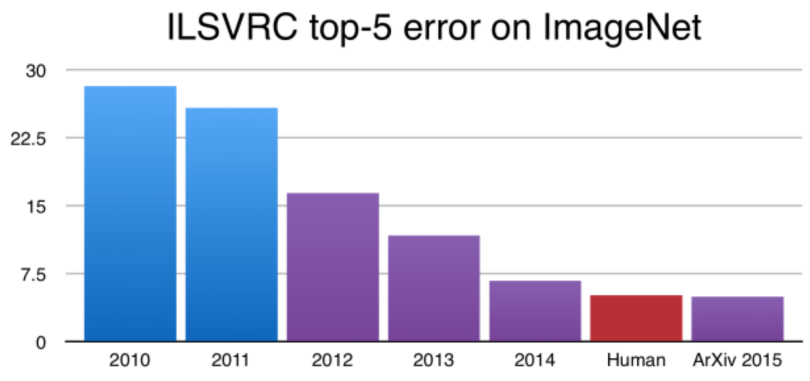


FIGURE 2.1: ImageNet Large Scale Visual Recognition Challenge [14]

However, it's likely that during this period of adoption that another hype-cycle is being created. It is then inevitable that when these solutions under perform the reaction will always be negative [10] [11]. The only solution to this is to allow businesses to understand the technologies they work with, either by reducing the technical barrier or by improving the education that already exists. This thesis will be primarily focused on reducing that learning barrier through the implementation of a black-box "as-a-Service" platform.

2.1.2 Current Attempts at Removing Barriers to Entry

While ML is currently still only being deployed to specific use cases, its uptake has been broadened by the advent of new technologies. Of note is the rise of cloud-computing. Cloud computing has been pioneered and commercialised by companies such as Amazon, Microsoft and Australia's Macquarie Telcom [15]. The proliferation of these platforms has enabled the low-cost, rapid development of a range of machine learning centred platforms [16] [17], including Microsoft's ML.NET and Google's Cloud AI. These platforms have significantly reduced the raw technical barrier to entry for ML services, by allowing anyone to connect to an ML platform without the need to install and debug a local implementation. However these platforms are still lacking fundamental accessibility requirements, focusing heavily on providing tools for machine learning experts in the form of trainers, models and clustering algorithms.

A renewed focus on human computer interaction (HCI) has assisted in this area. The recent failures of IBM's Florida Health oncology treatment tool (*Elekta*) Siwicki has shown that disregarding the human element of any ML implementation is detrimental to its success. In this case *Elekta* failed to consider the prevalence and range of acronyms, jargon and incorrect labelling in the medical field. It also did not consider that a large portion of the data it scraped (research papers from journals) could not be relied on without significant peer consultation. After this, the industry's focus included the handling and robustness of the collection of data from people i.e. in the NLP voice assistants [19]. Considerations for the physical environment and the concept of a 'growing' platform are now incorporated into the knowledge base for these algorithms through concepts such as Incremental and Continuous learning [20]. With this mentality of considering the human element of an ML implementation, the industry is avoiding another AI winter by addressing their biggest faults. However from here, a renewed focus must be made on how the traditional machine learning pipeline as a whole is itself preventative and in need of improvement.

2.2 Machine Learning

2.2.1 Defining Machine Learning

Machine Learning has a range of descriptions and how this body of work defines it, is as such:
An ML algorithm is a mathematical model that maps a set of input data $\mathcal{X} = \{x_1, ...x_i, ...x_n\}$ *to a label, from a set of possible labels* $\mathcal{Y} = \{y_1, ...y_i, ...y_m\}$ [21].

Target Domain
some transformation function
Label Space

FIGURE 2.2: Generalisation of the ML transformation

In this thesis, a core focus is the domain that the data is from. A Domain $\mathcal{D}$ consists of two components, a feature space $\mathcal{X}$ and a marginal probability distribution of P(X) where $X = \{x_1, x_2, ..., x_n\} \in \mathcal{X}$. For example, if the learning task is sentiment analysis from a set of social media posts, x_i would be an individual language element of those posts, X would be a set of posts, and $\mathcal{X}$ would be all possible x_i elements. For any given Domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$ there exists a task $\mathcal{T}$ which consists of a label space $\mathcal{Y}$ and a transformation function (sometimes referred to as a predictive function) $f(\cdot)$ such that $\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}$.

Instances x that are from the data input after deployment belong in the target domain $\mathcal{D}_T$, whereas the domain that was used to train the predictive model existed in the source domain $\mathcal{D}_S$. In the traditional machine learning pipeline the assumption is made that the source and target domains are identical, such that from the probabilistic viewpoint $P(y|x_T) = P(y|x_S)$.

2.2.2 The Traditional Machine Learning Pipeline - Classifiers

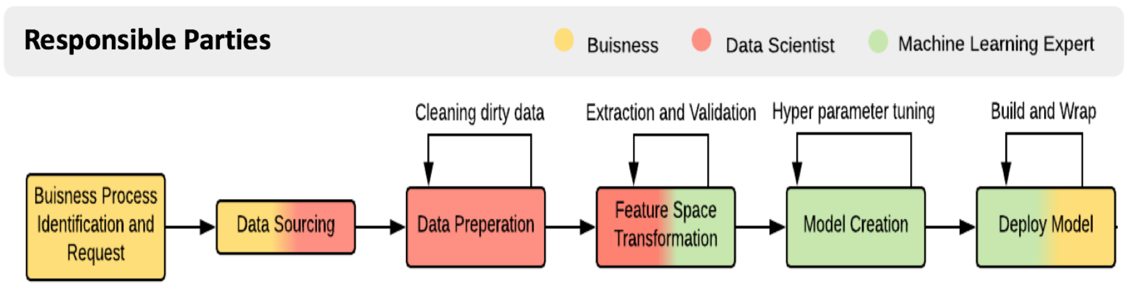


FIGURE 2.3: The Traditional Machine Learning Pipeline and Associated Responsible Parties

1. **Business Understanding/Review:** The first and rarely covered step of the pipeline is the business development stage [22]. This stage considers questions such as "Will the problem be solved by a machine learning application?", "Is there a simpler solution?" and "Are we as business capable of implementing this solution?". Both experts from both business and the ML field are required to engage at this stage, which can drastically reduce the accessibility of ML as a business option.

2. **Data Sourcing and Preparation:** Once the solution has been determined to require machine learning, the training data needs to be gathered, cleaned and labelled. This can be an intensive process for the business as the amount and quality of labelled data that can be provided is inherently linked to success of the classifier [23]. Here size is not as important as ensuring that a complete representation of the targeted environment across all labels is gathered [24].

3. **Feature Extraction:** After the data is made ready, it's then necessary to extract the feature space that will be used to train the model. This extraction is unique to the data type. For example in the feature extraction of text data, the sentence data can be simply reduced into "Stems and Leaves" which can then be converted into a numerical vector that is better handled by ML algorithms [25]. In

deep learning applications this feature extraction layer has been made effectively redundant, as deep learning approaches allow the model to learn and build its own feature extraction layer [26].

4. **Model Training:** With the feature space built, a model needs to be selected and trained on the sample data. Typically the choice of model would be made at the beginning with the Business Review stage. The selection of hyper-parameters for the model is generally a ballpark estimate made by a machine learning expert, and then tuned based until an appropriately successful classifier is found [27]. The success of the classifier is judged on a validation set of data that is separate to the training data. Stratification of the validation and training data set is important in reducing the capacity of the model to over fit to the training data, making it ineffective when applied to the target data set [28].

5. **Deployment:** After the model has been trained it will need to be encapsulated such that the user is able to input in new data to classify and receive those classifications appropriately. Typically in cloud based applications this takes the form of RESTful API frameworks that work over HTTP requests [29], and for embedded models traditional design patterns are still applicable.

2.2.3 Where Machine Learning Exists on the AI Curve

In order to understand the purpose of ML, it's important to consider where it fits in terms of the larger AI spectrum. As despite its reputation as a cure-all, ML is still in it's infancy especially when considering its integration with Business Intelligence systems [30] [31].

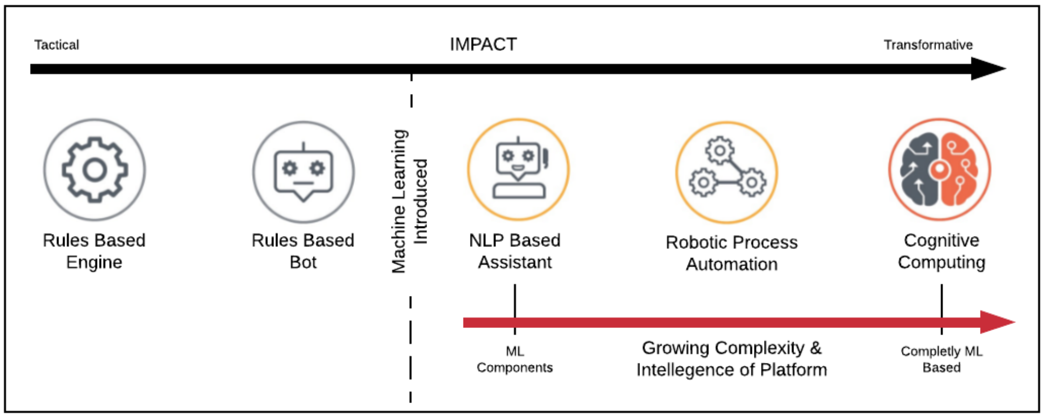


FIGURE 2.4: The Growing Complexity of AI

All products that market themselves as AI or ML based, are not fully cognitive. They use ML components that sit in explicit domains and satisfy set tasks. In the case of NLP based digital assistants, the task is sentiment analysis and the domain is human voice. Accents and slang of the user introduce perturbations into this domain that will affect the performance of the NLP program [32]. True cognitive computing is able to overcome these domain changes given that it's a basic requirement for true AI to learn naturally [33].

2.3 Common Machine Learning Algorithms

There are an impressive amount of machine learning algorithms available, each with their own strengths and weaknesses.

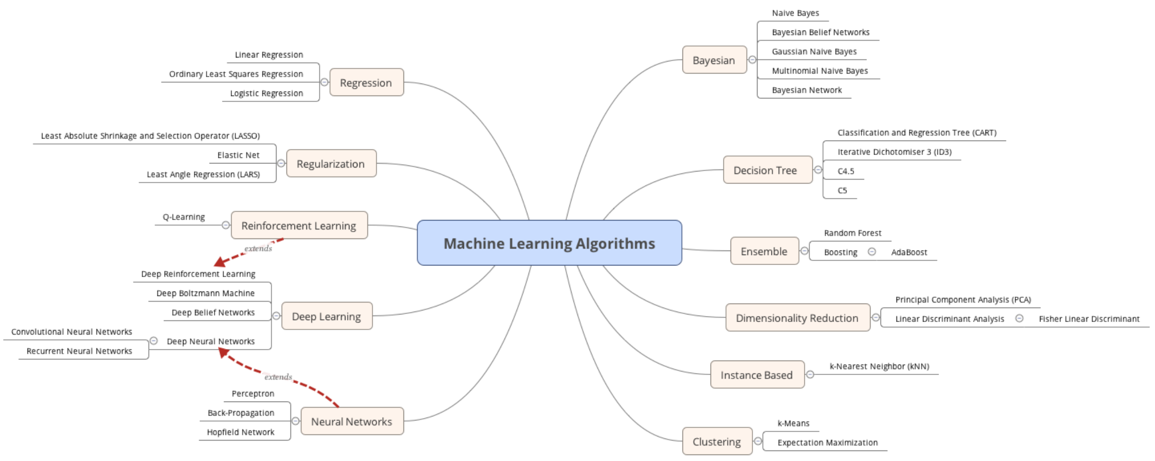


FIGURE 2.5: The Wide Array of Machine Learning Algorithms [34]

Selection of the most suitable algorithm is dependant on the purpose of the machine learning application. There are 3 main types of ML algorithms supervised, unsupervised and reinforcement; with some models being able to perform in multiple areas. The focus of this thesis is in the supervised theatre where the algorithm learns from a labelled data set for the purpose of prediction. Supervised learning algorithms fulfil 3 main use cases classification, regression and anomaly detection. The selection of an algorithm is made by considering the purpose of the algorithm, constraints such as training time v. accuracy, linearity of the data set, ease of implementation and possible business limitations [35].

2.3.1 Applications of Supervised Machine Learning

Supervised learning is a type of machine learning that uses a known, labelled data to train an ML model to predict future outcomes.

Anomaly Detection

Anomalies are defined as points of data that exist outside the normal distribution of the set [36]. Also refer to as outliers, these points are important in many applications as they can exist as warning that something outside the normal is happening within a system such as credit card fraud and cyber-security. The most common machine learning approach to detecting these anomalies, groups normal data together and returns any point that does not fall within these regions [37].

Regression

Regression is the task of relating a data domain to some continuous model. Regression problems are focused on identifying trends and patterns in known data and making predictions on future data [38]. Linear Regression problems are typically easier to solve, but in high complexity data environments, non-linear trends are much more likely. ML applications are typically applied to solve non-linear problems [39]. Examples of these problems include predictions on stock prices, housing markets, medical trials or engineering applications.

Classification

Classification is task of relating a data domain to a discrete set of class labels [38]. There is some overlap between a regression and classification problem, as classification algorithms are still able to map to a continuous output variable. As the output of the classifier takes the form of a probability to some class label [40], e.g. the likelihood the image contains a dog. Classification problems are either a binary (two-class or binomial classification) or n-categorical (multi-class) [41]. ML models are capable of building non-linear transformations that most other non-ML approaches are only capable of doing after extensive manual tuning.

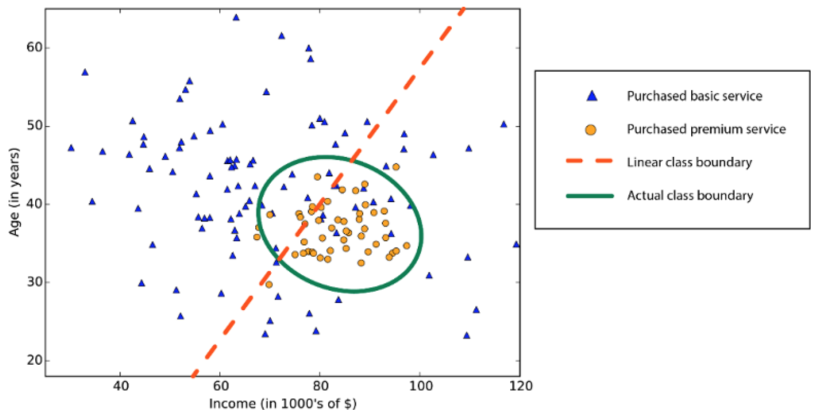


FIGURE 2.6: Solving non-linear scenarios using a SVM [41]

ML models that are typically applied to this task include SVMs, neural networks, probabilistic algorithms such as Naive-Bayes, hierarchy models such as decision trees and kNNs [42]. The choice of algorithm (and sometimes the hyper-parameters of those algorithms) in a classification task has additional considerations unique to the problem. That falls under the umbrella of a 'No-Free Lunch' optimisation problem [43]. These considerations include:

- **Bias and Variance:** The trade-off between these two can typically be referred to as the 'flexibility' of the algorithm. The flexibility is also known as under or over fitting. A highly biased system will not react to neither noise or new legitimate data. The inverse is true for a highly variant model [44]. Some algorithms such as Decision Trees [45] and Neural Networks [44] are known to be sensitive to this trade-off, while algorithms such as SVMs perform better [46].

- **Complexity Vs Availability:** Higher complexity algorithms with greater depth are typically capable of mapping more complex data domains against correct labels [47]. However there is precedent that complex models when trained on small or noisy data sets are unreliable and generally perform worse than simpler, better trained models [48]. This is a significant consideration with data from the medical and advanced sciences [49].

- **Curse of Dimensionality:** Refers to the problems that arise in handling data in the higher dimension space (1000s) that don't occur at lower dimensions [50]. Data must have a distribution that covers the all possible mappings for each dimension combination. An extension of the 'Hughes Phenomenon' dictates that for each dimension, there should exist at least 5 training examples against each other dimension [51] [52]. SVMs and kNNs can make use of a mathematical 'Kernel Trick' to reduce the impact that higher dimensions have on their performance [53].

2.3.2 Deep Learning Supervised Machine Learning Algorithms

Deep Learning ML Algorithms are a subset of algorithms that break down the feature space at each layer to interpret and learn new and more abstract elements at each layer [54]. This layered method abstracts away a large proportion of the feature space engineering [26]. Of particular note in this field are Deep Neural Networks (DNNs), which include Multi-Layer Perceptrons and Deep CNNs. In the context of this body of work's proposed pipeline, these models are important, as they are particularly known for their ability to handle all data types.

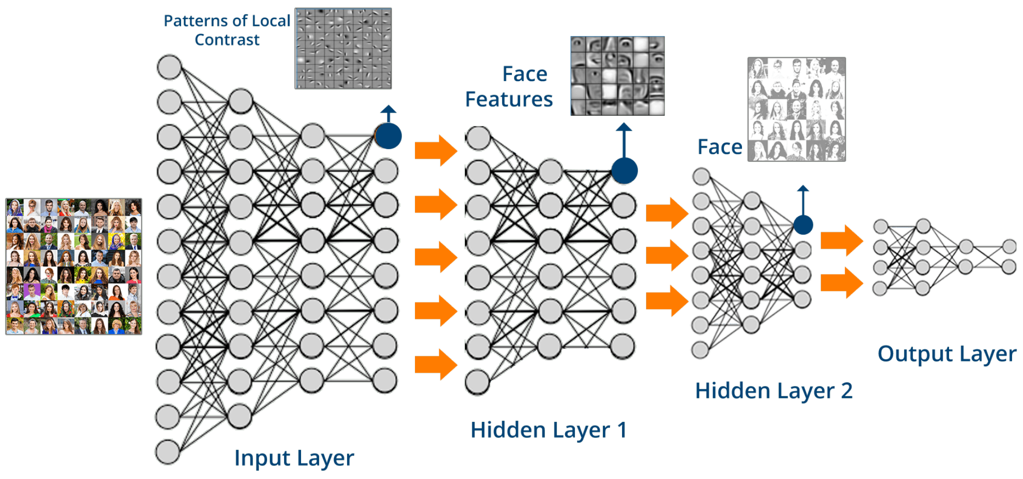


FIGURE 2.7: Progressive feature abstraction in DNNs [55]

Multi-Layered Perceptrons

A perceptron is the most basic neural network and consists of three levels of nodes; an input (can be many nodes), hidden (single node) and output (single node). Each node in the input layer is connected to the hidden node with some weight. The sum of these weighted inputs plus a constant is then used as the input to a non-linear 'activation' function [56]. The output of this activation function is passed on.

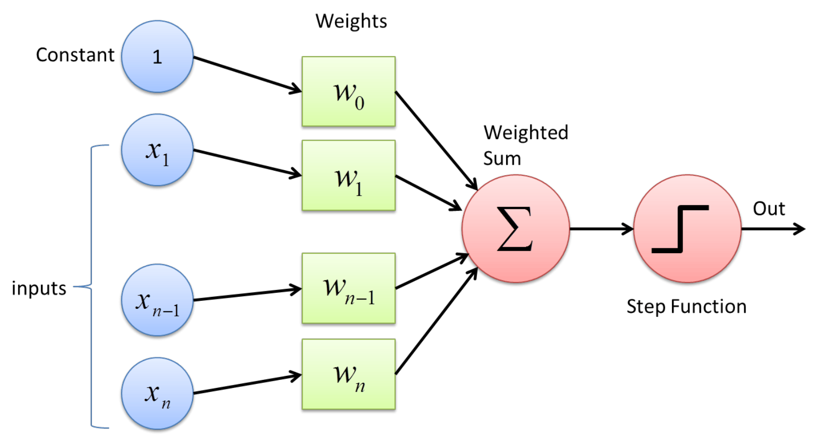


FIGURE 2.8: Simple Perceptron [57]

In MLPs these perceptrons are layered adjacently and vertically, creating a strong statistical cumulative distribution function [58]. The selection of MLP's hyper-parameters includes the depth and width of the hidden layers, the choice of activation function, starting weights and constants.

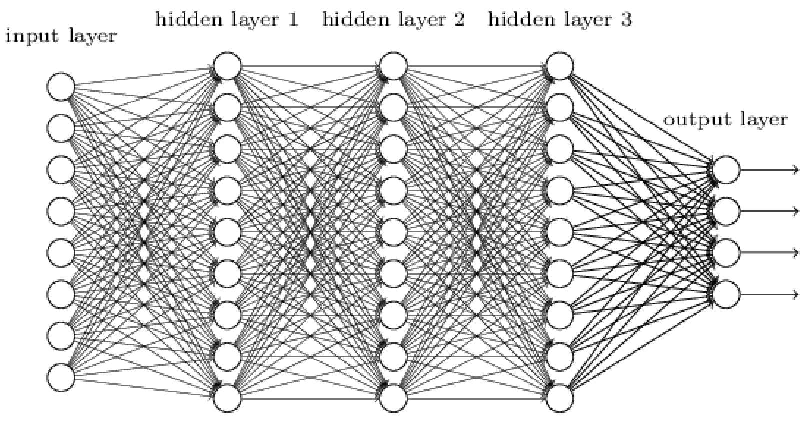


FIGURE 2.9: Deep Neural Network (Multi-Layered Perceptron)

Convolution Neural Networks

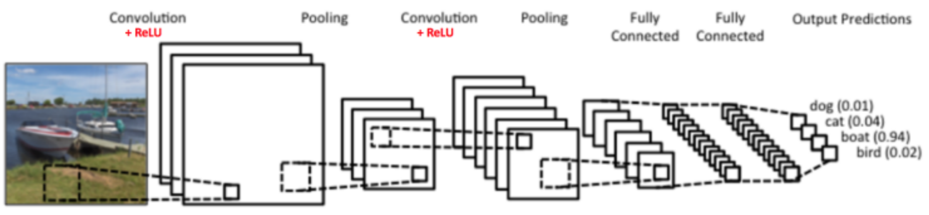


FIGURE 2.10: Convolution Neural Network[59]

CNNs gained significant popularity in the 80s and 90s due to their success in the computer vision and acoustic modelling fields. CNNs are heavily inspired by the behaviour of visual cortex neurons, reacting only to stimuli that occurs within a restricted field and not the whole environment [60]. In practical terms this presents itself as a layered network that pulls in data from nodes within its 'visual range' (convolution layer), applies some mathematical activation and weight (ReLU layer), generalises the outcome (pooling layer) and then after multiple applications of these layers the heavily transformed feature set is passed onto a fully connected layer that acts as a typical MLP network [61] [62] [63].

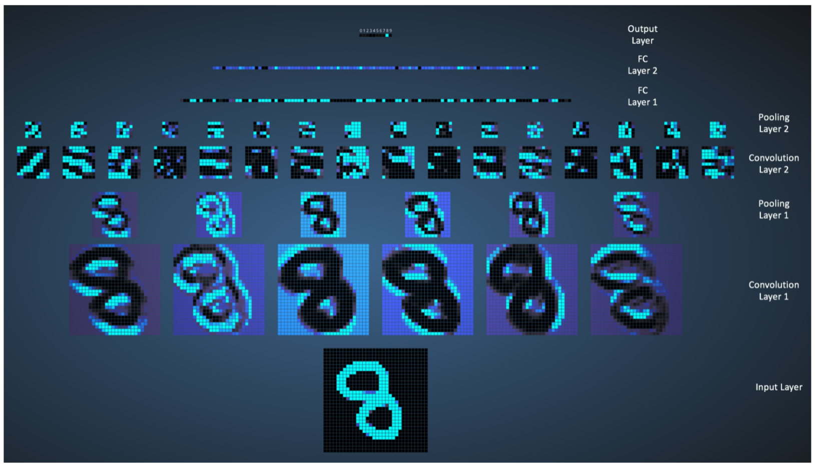


FIGURE 2.11: Convolution Neural Network - Practical Representation [59]

Residual Networks and Shortcut Connections

Recently CNNs have had excellent success in the Visual Recognition field, in 2012 when applied to the MINIST handwriting database they achieved an error rate of 0.23% [64]. In 2015, using the extensive *ImageNet* database, a CNN was able to make a commercial facial recognition product that located faces at orientations previously incapable by ML recognition applications with a 91.79% success rate [65].

This success is attributed to the solving of one of the major problems with all deep networks, but are notable in the CNN space. The problem was known as the 'vanishing gradient' issue. Where as the feature space moved from one layer to the next, the feature space that was passed on would become more and more abstract and begin to lose relevance. To counter this, the residual learning framework was built. Where the output of the node and its input (or an input further up the layers) are summarised before passing on: $H(x) = F(x) + x$. These are typically referred to as "Residual Blocks'.

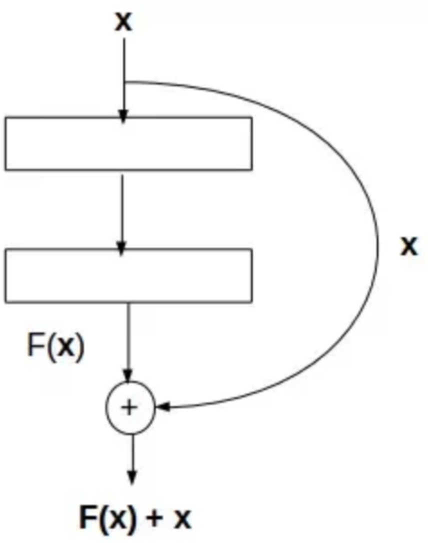


FIGURE 2.12: Residual Layer

The current leader using the shortcuts and residual blocks (known as dense blocks in this model) is the *DenseNet* model type. This model shortcut concept to the extreme, within each dense block the output of each layer is passed as an input to all layers below it.

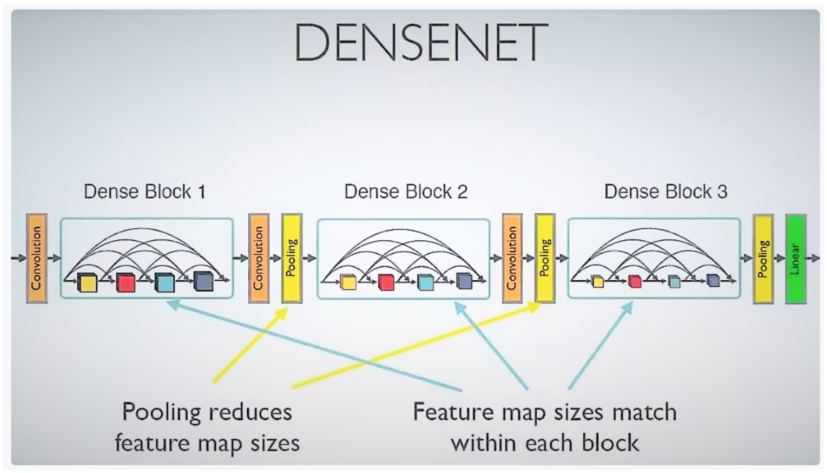

DENSENET
Dense Block 1
Dense Block 2
Dense Block 3
Convolution
Pooling
Linear
Pooling reduces feature map sizes
Feature map sizes match within each block

FIGURE 2.13: DenseNet Layout

2.3.3 Training Deep Neural Networks

The most popular form of training and the one that is implemented in this body of work, is known as back-propagation. When instance x from the training data is pushed through through the net, the result at the outputs (which is a confidence score 0-1) is then compared against the known label [66]. The error is pushed back down the model, adjusting the weights between nodes and layers using a stochastic gradient decent algorithm [67].

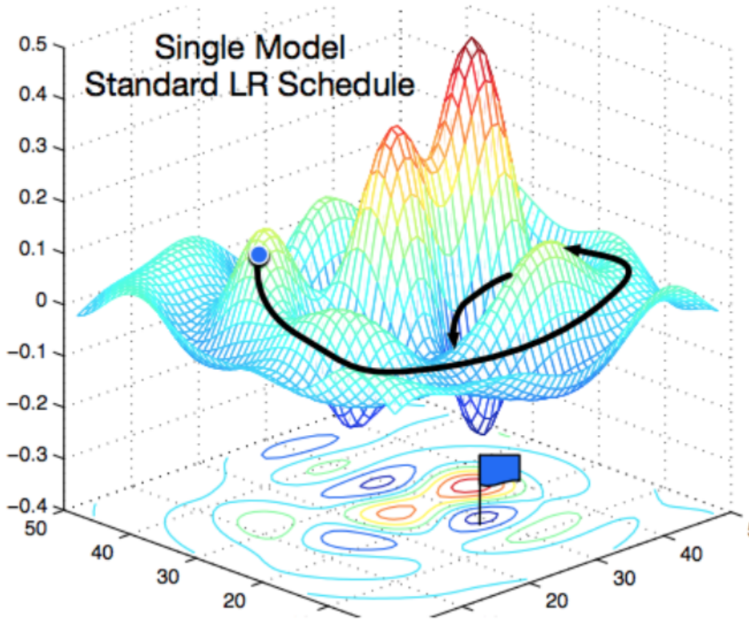


FIGURE 2.14: Stochastic Gradient Descent Visualisation [68]

Other training techniques include dropout [41]. Dropout is the process of randomly closing off nodes between training epochs in order to reduce the likelihood of over-fitting to the training set by reducing the reliance of certain nodes to form consensus [69]. The discovery of training techniques such as drop-out [70], depth over width hyper-parameter tuning [71] and data augmentation (such as noise jittering) [72] have all significantly improved the use of deep learning in all applications.

2.4 Sentiment Analysis and Natural Language Processing

Before the advent of data-driven ML platforms, the use of 'classical' NLP solutions utilises sets of known phrases, patterns and rules [73]. This method of 'symbolic reduction' was both time and space exhaustive and did not lend itself well to situations other than that where careful control of the domain and application context was possible.

Statistical based models that arrived in the late 80's / early 90's took advantage of the new wave of growing data availability. Models were built with a statistical understanding of how sentences and phrases were constructed as opposed to any pre-determined set of rules [74]. The issue of domain constraints were not addressed in this manner, however the flexibility of the systems were greatly enhanced.

NLP statistical models are split into two categorises described as either generative or discriminative. Discriminative models such as Neural networks will attempt to statistically classify a pre-known feature space $P(y|X)$ [75]. Generative models are capable of mapping new, unknown features by accounting for their similarities to known features [76]. Generative models are more robust to noise and over-fitting, but are overall harder to train and require large amounts of higher-quality training data. Gaussian Discriminant Analysis is an example of a generative model which builds a probabilistic space from training data that allows for the new features to be analysed without necessarily having to match known features. Notably some discriminative models can be extended to be generative, these include the improvements made to predictive neural networks used in semantic word representation such as the extension from *Word2Vec* to *Doc2Vec* [77].

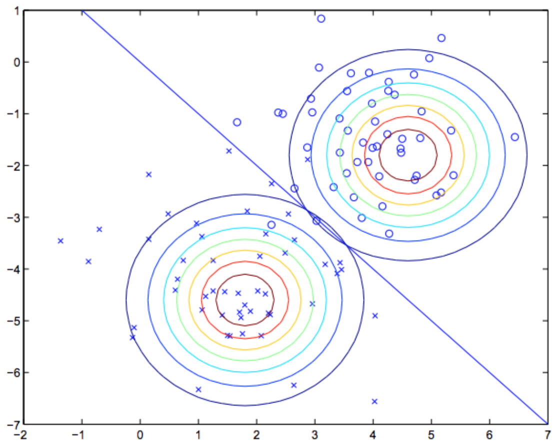


FIGURE 2.15: Generative Model - Gaussian Discriminant Analysis [76]

2.4.1 Word Representation

All statistical models regardless if they're generative or discriminative require that the words imputed be done so in some numerical way. A popular method is to break down sentences into it's 'stems and leaves' creating a 'bag of words' [25]. This model which converts a dense text into a single sparse vector ignores the semantics and syntax of a sentence, limiting the contextual impact of individual words. In order to improve the robustness of the model, a frequency modifier is attached to each word, to ensure that words such as 'the' and 'in' are given less importance over words specific to some desired mapping (i.e. relating offensive language to aggressive sentiment).

Newer methods for word embedding which take into account the semantic context of the words have been shown to be much more successful. Two common approaches with equally positive results are Hyperspace Analogue to Language techniques such as Glove Embedding and Predictive Neural Networks such as Word2Vec [77]. These models convert the initially poor representation of the 'bag of words' into a dense, multi-dimensional dimensional vector. With semantically similar words sharing closer proximity to one another [78].

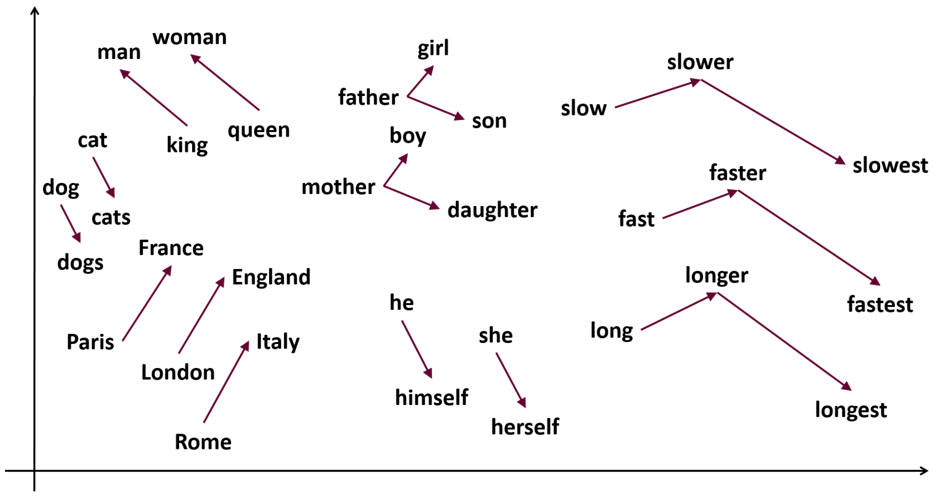


FIGURE 2.16: Word2Vec representation [79]

Semantic techniques consider the plurality and tense as well as other morphisms of words and can rationalise their similar counterparts. Of interest is the ability to take the vectors of words and process them algebraically by taking the cosine distance between vectors:

$$vector(ruler) + vector(woman) \approx vector(queen) \tag{2.1}$$

These techniques will be used in any NLP application to some degree. 'Bag of Words' while basic is quite popular due to its simplicity. However, the platform that this body of work will produce, will be deployed in any environment. Because of this, the complexity of the problem is significantly increased and an approach that considers semantic representation that can be extended to include new, unseen words will need to be deployed.

2.5 Image Recognition

Image recognition is a broad genre that includes any application where the classification of features within an image space is required. Notable applications include facial recognition, optical character recognition and content-based image searches. For object identification, steady progress has been made in the yearly *ImageNet* classification challenges. Since its inception, new approaches in the IR field have delivered exponential increases to model accuracy to the point where classification on this database currently sits on a 0.0225 error rate [14].

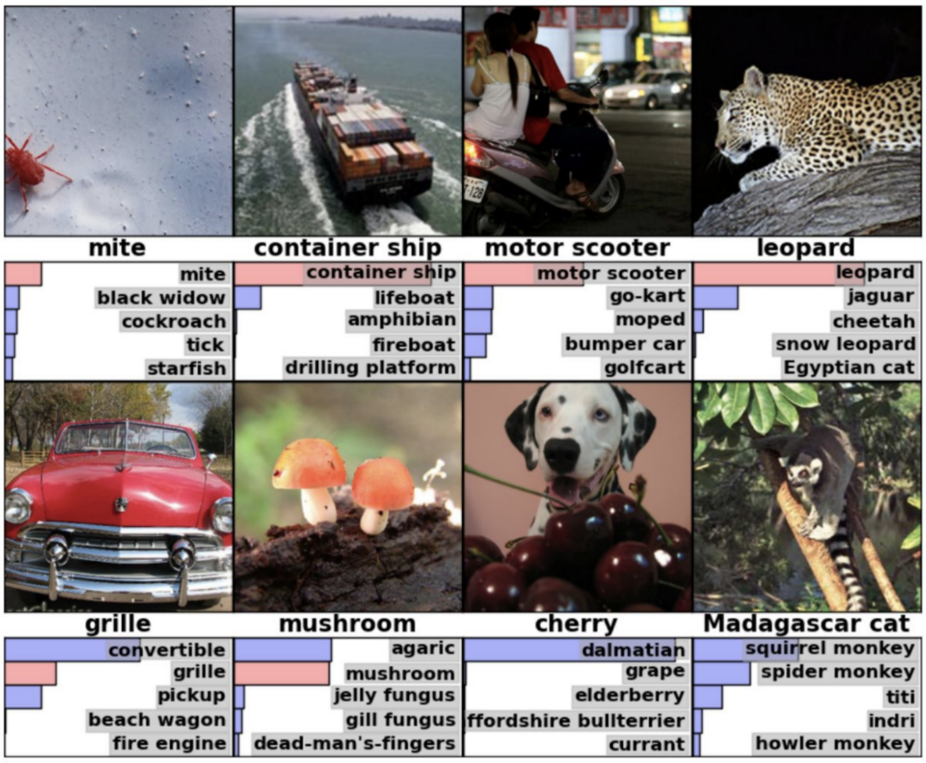


FIGURE 2.17: ImageNet Probabilistic Results

2.5.1 Salience Detection

Object recognition, one of the more popular IR tasks is a typical classification problem. Salience detection is a pre-processing step in the object recognition process that will isolate the most ′important′ object in an image. The following steps of the process will then classify that isolated object [80]. Salience detection is required for complex tasks such as medical imaging, where abnormalities are typically defined by their juxtaposition to their surroundings [81]. In CNNs, salience detection is typically covered within the initial layers and is one of the reasons for their better results with the image recognition problem space [82]. These approaches are still decided upon by the engineer and not natively learned by the classifier.

Simple salience detection solutions will take a set of background seeds and create a map that will be formed from pixels spreading out of that seed that to similar pixels. Algorithms to accomplish this include; Markov Chain [83], Cellular Automata [84] and Normalisied Cut [85]. However in applications such as medical imaging, the challenges of imagery artefacts (smudge on the lens, blurring, lens distortion, image noise etc.) leads to foreground and background blending that require a more complex approach.

These solutions are addressed by either a top-down or bottom-up approach. A top-down approach requires the system to have the benefit of already knowing what they should be looking for, by being given a filter bank beforehand [86]. This approach is not well suited to a generic approach such as that sought in a "ML-as-a-Service" platform and instead a bottom-up approach is required. The bottom-up approach extracts low-level features such as HSL components, texture and edges. Then using modified versions of typical spacial algorithms such as Cellular Automata [84] removes outlining and anomalous features. This leads to significant improvements in most image recognition problems [87].

2.5.2 Deep Learning and Image Recognition

Deep CNNs have been hailed for their breakthroughs in image recognition, being responsible for the 50% reduction in error rates on the *ImageNet* competitions [14] and achieving significant success in industry benchmarks such as CIFAR100, CIFAR250 and MNIST [88].

However, as IR goes from a research piece to an industry application, accuracy must be balanced with computation time and spacial considerations [82]. The size of the net becomes a significant consideration when todays advancements with residual layers and "short-cuts" in models such as *DenseNets* which hold the current records [89]. These networks that allow for all layers to accept each layer below as an input, improve feature propagation while also exponentially adding to the space required by the model with each layer. For example in the most recent *ImageNet* competition, the best entry that didn't contain extensive use of short-cuts was a basic ResNet that had 1.4 Million parameters, where as *DenseNets* had 24.8 Million.

Because of this large data overhead other neural network architectures have been also investigated. MLPs are an attractive option given their scalable, parallelisation-friendly and well performing nature. MLPs can still make use of images preprocessing pooling layer while significantly reducing the size their nets take up [90]. Several studies seeking to compare CNNs to MLPs have produced mixed results, likely due to fundamental differences in the methodology of the experiments. While CNNs are still favourable, the gap between the performance of the two model types is such that MLPs still remain a favourable avenue of research.

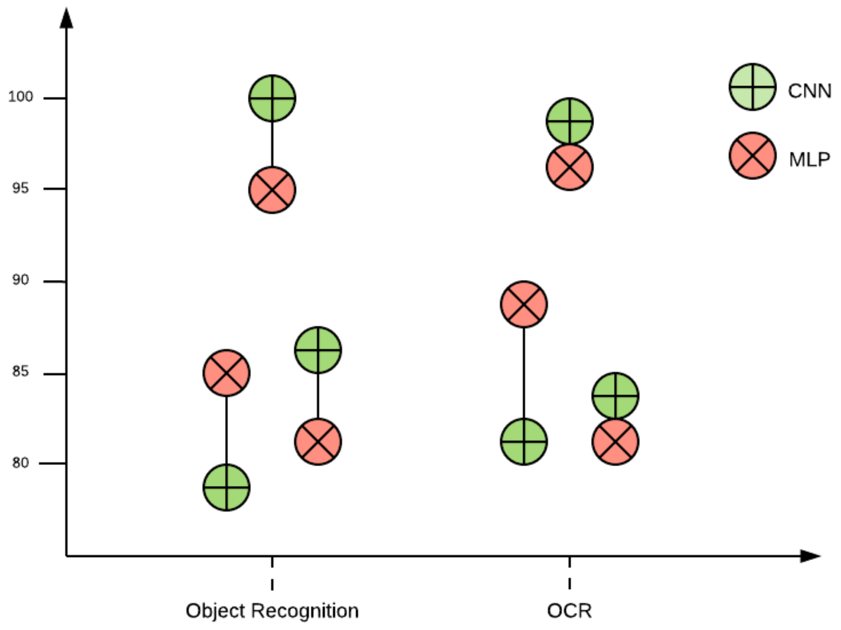


FIGURE 2.18: CCNs Vs MLPs on Image Recognition Applications

	[91]	[92]	[93]	[94]	[95]	[96]
MLPs						
OCR	94.69				89	81.7
Object Recognition		95.86	84.7	81.62		
CNNs						
OCR	96.44				80	83.9
Object Recognition		99.32	79.3	85.39		

The outcome from these results show that CNNs are still favourable for their accuracy. However, this thesis' "ML-as-a-Service" platform will be built to ensure that, because of the continual improvements made to ML models, the platform is built to be modular and capable of swapping out model types. Ensuring that should MLPs eventually become more desirable by the user, the platform is still relevant.

2.6 Transfer Learning

Machine learning has been commonly referred to as an application of uniform convergence theory. As for any given input instance x_i which is known by the optimal predictive model $f(\cdot)_{opt}$, it will be mapped correctly to some label y_i [97]. The current proof of traditional machine learning uses this uniform convergence theory, stating that should the model be provided enough data, then it will provide an optimal model.

This proof of optimally is the basis of the traditional ML pipeline, because it is a balanced equation. However it is a naive and incorrectly assumes that the domain the model was trained in (D_S), is the same as the domain that it will be applied in (D_T). This will only hold true if the data the model is performed against was captured at the same time and environment as the data it will be applied on. In any other situation these models exist in environments that are always changing. It is only the degree of change or the size and quality of the data selection that will stretch the domain dissonance enough to be noticeable. Sudden changes to the domain is the most common case from where this problem will arise.

Transfer learning is a field of machine learning which takes into account the context of how the model will be applied. Accounting the possible change of the expected domain and task upon deployment [24]. This thesis will focus on a transductive transfer learning approach to solve the problem of changing domains.

2.6.1 The Challenges and Rewards of Transfer Learning

The goal of transfer learning is to improve the overall machine learning pipeline, by taking advantage of knowledge already sourcable by the platform and extending it for new domains and tasks [24]. The benefits of transfer learning comes in three ways:

- **Initial Performance:** The initial performance of the model, compared to an 'ignorant' agent (a completely untrained model).
- **Learning Time:** The speed of which the asymptotic error rate is reached compared to a classifier trained using source data.
- **Final Performance:** The final error rate of the model against a model trained with source data.

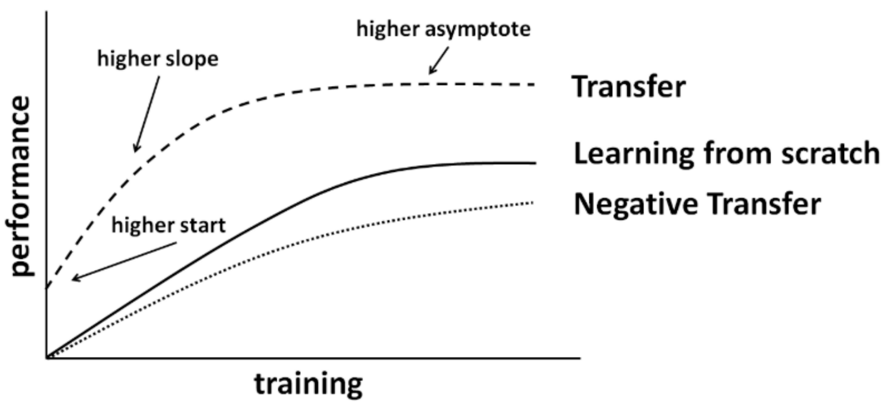


FIGURE 2.19: The improvement and risk of transfer [98]

All transfer learning problems need to be approached with 3 fundamental questions, failing to address these questions while designing the approach can lead to the problem of Negative Transfer. Negative transfer occurs when the transfer learning process is destructive to the learning process as a whole and creates models with poorer performance as compared to a traditional training pipeline [24]. The 3 Questions are:

- **What to transfer:** Considers which knowledge available in the source domain is applicable in the target domain. All transfer methods address this problem differently, so before deciding on which approach to use, a decision must be made on the domain that the model will operate in.
- **How to transfer:** The 'how' concerns the process and algorithms in order to achieve that. The approaches that are generally considered include *Instance Transfer*, *Feature Transfer*, *Parameter Transfer* and *Relational Feature Transfer*. To perform these approaches data boosting methods, clustering, graphical representation or end-to-end learning (GANs) all considered.
- **When to transfer:** A holistic question that takes into account the context of the application, evaluating the trade-offs created by applying the transfer. In applications of *Relational Feature Transfer*, sometimes computationally expensive algorithms need to be applied, which may not be appropriate for a given hardware implementation.

Overall transfer learning is about building a platform that is able to use the knowledge of other domains and tasks. This goal is often referred to as "life long learning" where ML platforms can be put in place and expected to grow to accommodate the changing environments and purpose they exist in [99].

As a new area of ML, transfer learning has many hurdles to overcome. Companies may be hesitant to save extremely large amounts of source data and models to be used in computationally extensive training processes. However on the inverse, transductive transfer learning approaches have been shown to, if they have a significantly apt amount of target data, fair well when there is bias or noise in the source data set. Because of this, already existent data retention programs may have significantly less operational overhead.

2.6.2 Transductive Transfer Learning - Domain Adaptation

When the aim of the process is to reconcile the dissonance between the source and target domain, the process is referred to as 'Transductive Transfer Learning'. Two scenarios exist for transductive transfer learning; the feature space between the two domains are not the same ($\mathcal{X}_D \neq \mathcal{X}_T$) or while the feature space is the same ($\mathcal{X}_D = \mathcal{X}_T$) the proportional distributions of the feature space are not equal ($P(X)_D \neq P(X)_T$). Many of these transductive transfer learning approaches do not require a labelled target data set, avoiding the expensive process of labelling data sets [24].

For the transductive transfer learning case there exist another case where the feature space is the same ($\mathcal{X}_D = \mathcal{X}_T$), but the probability distribution is not the same ($P(X_D) \neq P(X_T)$) and while the tasks have identical label sets ($\mathcal{Y}_D = \mathcal{Y}_T$), the conditional distribution (the mapping function) is similar but not the same ($P_D(Y|X) \sim P_T(Y|Y)$). In this case the solution takes the form of a semi-supervised retraining that uses a method known as 'Domain Adaptation' [100]. This is the scenario that will be the focus of this thesis.

A notable application of domain adaptation is in addressing the sample distribution bias problem that occurs when the ML pipeline doesn't have full control of the data presented to it [101]. In this application, the ML model must be able to from the gathered source data, select or map source data that is inline with an unbiased target sample set.

There are two typical transfer learning approaches to domain adaptation; *Instance Transfer* and *Feature Representation Transfer*. An *Instance Transfer* approach is typically referred to as data boosting. Where data from the source domain is weighted based on it's similarity/distance to the target domain. This approach does not produce any new data and is typically employed if the width of the source domain puts the target domain as a significant subset within it. On the other hand a *Feature Representation Transfer* approach which seeks to build a representative data set of the target domain using the source domain as a starting point does produce a new data set. This approach provides a far superior amount of data to be available during model training. However, there is the risk that the new data set is not an accurate representation.

2.6.3 Domain Adaptation - Feature Representation Transfer

Feature representation transfer aims to create a mapping between the source domain and the target domain. A good visualisation of the domain disparity that causes models to suffer is by comparing the MNIST handwritten numbers database against the USPS coloured handwritten number data set.

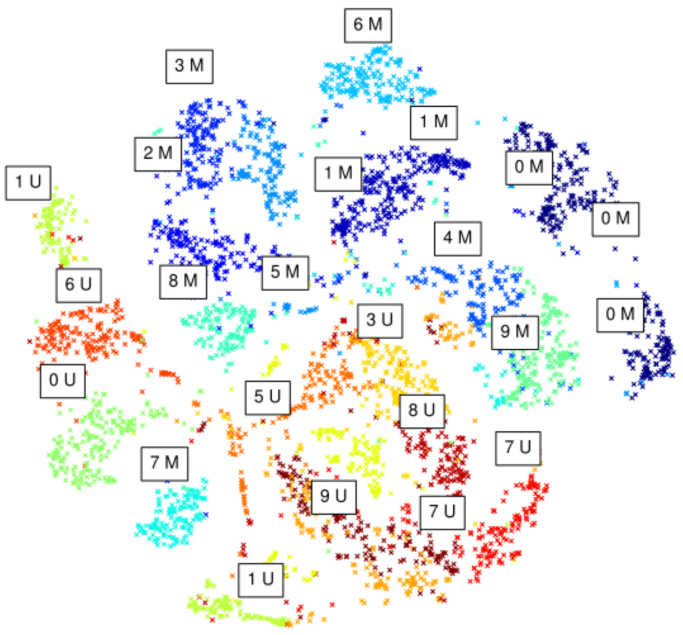


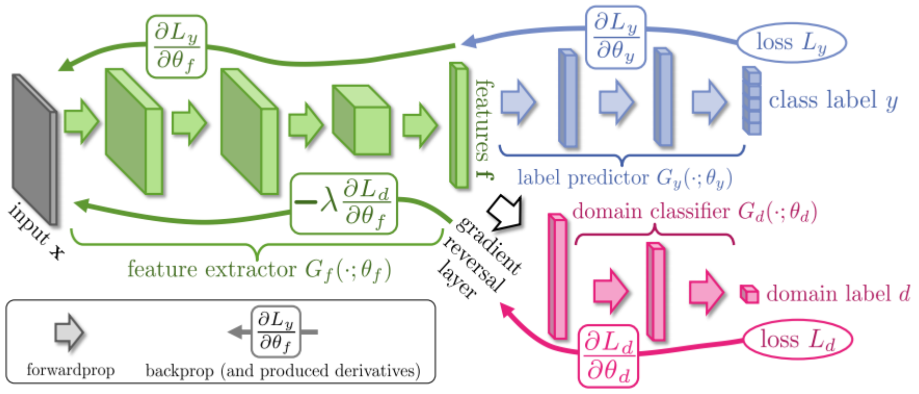
FIGURE 2.20: 2D visualisation of the MNIST-USPS data-sets. [102]

The disparity of the two data sets means that when the USPS data is classified on a model trained with MNIST, the results are not accurate.

The feature adaptation process can be treated as unsupervised or semi-supervised. When the target domain lacks labels, an unsupervised clustering method is used to find a joint distribution between the source and target domains. In the semi-supervised process, the domain invariance is ignored and target data is used to select the source data with the same categories. After either method, the model is then retrained (using the above discussed method and others) on the source data that has been transferred into the target domain [103].

Generative Adversarial Networks

Generative adversarial networks are two tiered deep learning neural networks that have the ability to produce 'new' media. The training process is adversarial in nature with a 'generator' model that creates media and a discriminator' model that evaluates the new content against some known content [104]. A gradient reversal layer flips the loss of the discriminator to be the gain of the generator [105]. When the discriminator is successful in differentiating the synthetic data produced by generator from the target data, the loss is transferred down to the generator.



$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
loss L_y
class label y
features $\mathbf{f}$
label predictor $G_y(\cdot;\theta_y)$
$-\lambda \frac{\partial L_d}{\partial \theta_f}$
domain classifier $G_d(\cdot;\theta_d)$
input $\mathbf{x}$
gradient reversal layer
feature extractor $G_f(\cdot;\theta_f)$
domain label d
$\frac{\partial L_d}{\partial \theta_d}$
loss L_d
forwardprop
backprop (and produced derivatives)

FIGURE 2.21: GANs - Gradient Reversal [105]

The Min-Max function defined in the original GAN paper that is commonly employed today, albeit with minor deviations is:

Equation 2.2 The overall Min-Max value function

$$\min_G \max_D \mathbb{E}_{x\ p_{data}(x)}[logD(x)] + \mathbb{E}_{z\ p_z(z)}[log(1 - D(G(z))] \tag{2.2}$$

where:

- z is the input vector into the generator (Source data set + Noise)
- x is the target data set
- D is the confidence of the discriminator
- G is the output of the generator

Equation 2.3 The overall Min-Max value function

$$\nabla_{\theta(G)} \frac{1}{m} \sum_{i=1}^{m} log(1 - D(G(z^{(i)}))) \tag{2.3}$$

When the data is continuous the gradient reversal is described as:
GANs due to the complexity of the task are tough to train [106]. In the Domain Adaptation case the generator which traditionally begins learning from a random noise vector, instead uses the source data as the starting point. Because of the enforced similarity between the two domains at the start, the training time for the generator is significantly smaller than those trained from noise. Similar to the process of 'jittering', the slight difference between the target and transferred data-set can lead to better performing models than those trained on purely the target data set of equal size [107].

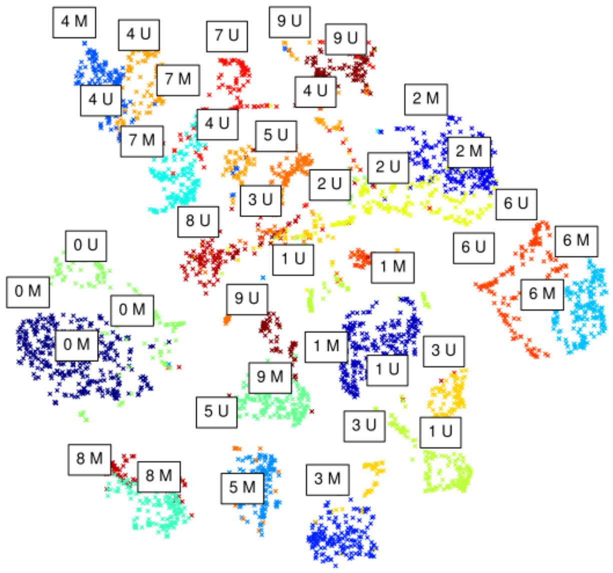


FIGURE 2.22: MNIST-USPS data-sets after Domain Adaptation [105]

The traditional GAN as described by Ian Goodfellow is not data agnostic and can only operate on continous data types [108]. Continuous data such as image data can produce loss functions between two images on a pixel to pixel or convolution basis [109]. However, NLP sparse vectors are discrete data types, with the distance between two sentences incalculable. Several approaches such as *seqGANs* [110] and *MLE-GANs* [111] aim to bridge the gap by sampling and inferring. Since the outputs of the discriminator are non-differential, gradient reversal is not possible between the discriminator to the generator. Instead an estimation of the loss to the generator is found and applied.

Sequence GANs:

SeqGANs as described by Yu and his team in their paper *Sequence Generative Adversarial Nets with Policy Gradient* [110] is an extension of a "encoder"-"decoder" network where a sequence is encoded and then decoded by the generator. That decoding is the new synthetic data i.e. The encoded sentence "Wǒ shì Tys" is decoded as "I am Tys" or a more apt example for the use case of this thesis "I am Tys" would be decoded as "Hello Tys, I am Bot". In this case the encoder is the semantic preprocessing layer such as *word2vec*. The sequence that is created by the decoder (generator) is built using a series of states. For each state (the sequence of words built so far) the next likely word is found.

$$\begin{aligned} C &= \sum log(P(x|h)) \\ &= \sum \sum_t P(x_t|x_{1:t-1}, h) \\ &= \sum log[P(x_1|h) * log[P(x_2|x_1 : 2, h) * log[P(x_3|x_1 : 3, h) * ... * log[P(x_T|x_1 : T, h)] \end{aligned} \tag{2.4}$$

Where h is the input sequence and x is the output of the previous layer in the decoder. The output of the generator x is passed onto the discriminator with a sample set of target sentences and the synthetic sentence is evaluated. The pipeline can be simplified to:

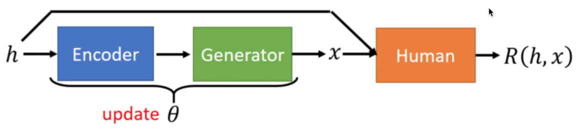


FIGURE 2.23: Simplified SeqGAN pipeline

An MLE optimisation function exists for this pipeline where through modification made in the parameter space of the generator θ, the minimum loss in the discriminator is found.

$$\begin{aligned} \theta &= argmax_\theta \bar{R} \\ \bar{R} &= \sum_h P(h) \sum_x R(h, x) P_\theta(x|h) \end{aligned} \tag{2.5}$$

What makes sequence GANs different is that it is composed of many encoder-decoder networks. For each state that is generated, a set of the next most likely words is then found.

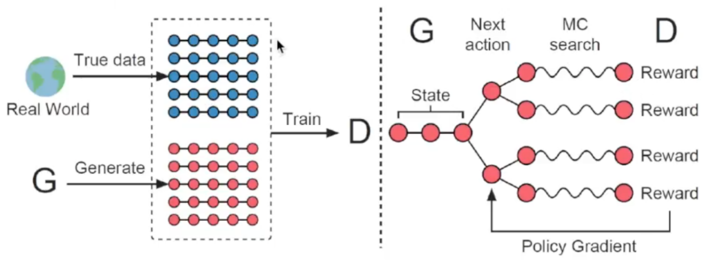


FIGURE 2.24: Sequence GANs [110]

A monte-carlo search is then conducted on all the possible outcomes from the discriminators to build a policy gradient. This gradient is then passed back down to the generator to modify its parameter space. This policy gradient is an extension of the MLE optimisation function as performed above.

Equation 2.6 Sequence generator reward maximisation function

$$J(\theta) = \mathbb{E}[R_T|s_0,\theta)] = \sum_{y_1 \in \mathcal{Y}} G_\theta(y_1|s_0) \cdot Q_{D_\phi}^{G_\theta}(s_0,a) \tag{2.6}$$

where:

- R_T is the reward for a complete sequence, there are no partial rewards for incomplete sequences.
- $Q_{R_\phi}^{G_\theta}(x,h)$ is the action-value function, also known as the cumulative award across all the discriminators or the policy gradient.

2.6.4 Inductive Transfer Learning - Multi Task Learning

In the scenario where there exists a difference between the task of the source and target $\mathcal{T}_S \neq \mathcal{T}_T$, but the difference between the domains is negligible $\mathcal{D}_S \sim \mathcal{D}_T$ ($\mathcal{X}_S = \mathcal{X}_T$ but $P(X_S) \sim P(X_T)$), an inductive transfer learning approach is applicable. The proposed "ML-as-a-Service" platform could make good use of this application of transfer learning. Enabling it to use of models from the same domain, but retrained to perform new tasks, benefiting from the transition [24].

2.7 Conclusion

By reviewing the work of others in this field, a final conclusion and direction that addressed all the proposed questions at the start of this literature review was decided upon.

The proposed "ML-as-a-Service" platform must be data agnostic to satisfy machine learning's largest consumer, the non-expert. However, similar platforms have been slow to accept the needs of the non-expert user. If they continue to not to be considered, then eventually when another hype-cycle is created in the industry and poorly formed expectations are not met, the ML field will again collapse.

The most severe problem that was identified in this review, occurs when models are forced to operate in domains they were not trained for. A problem that generally occurs when the data used to train the model is not the same as what will then be captured once deployed. To overcome the lack of representation, a transductive transfer learning pipeline will be built and embedded within an "ML-as-a-Service" platform. This platform will ultimately deliver optimal classifiers, based on up-to-date literature the most optimal models for image recognition are *DenseNet*'s, which have also shown capability in the NLP space.

TO address the domain dissonance, this two-step pipeline will take a *Feature Representation Transfer* approach to domain adaptation, a subset of transductive transfer learning. The approach will utilise generative adversarial networks for both continuous and discrete data types, to create new data sets.

Implementing this pipeline to create a data agnostic platform will remove all barriers preventing non-experts from creating their own ML solutions. In the context of the history of AI/ML adoption in the industry, the removal of these barriers is necessary in order to prevent another freeze as those described in the late 20^{th} century.

3

Software and Hardware Setup

This section will include software and hardware components such as python packages, GPU setup and management the OS environment that will need to be replicated in order to ensure the results covered within this thesis can be repeated.

3.1 Software Packages

The following packages are used in this body of work. For those packages that are implemented directly into the 2-step TL pipeline, alterations were made to improve and extend their ability. These alterations are briefly described below, with more detail available in the module overview chapter. The code base for this thesis was written in Python 3.6.

- *PyTorch v0.41* - Provides the framework for handling ML specific variables, data augmentation, ML objects/functions used in the different models (i.e. layers, parsers and optimisers) and training (back-propagation) functions. *PyTorch*'s main benefit over its cousin *TensorFlow*, is its deployable optimaisation patterns. In the machine learning context, correctly implemented optimisers significantly reduce the training overhead of model creation and retraining. The second major difference is that PyTorch dynamically allocates its memory, allowing objects to be stored and recoverable outside of the sessions defined in frameworks such as *TensorFlow*. This is important in a research context where results and objects need to be captured and used by other ill defined code structures.

- *DenseNet* - Published by Zhuang Liu along with his and his collegue's paper *Densely Connected Convolutional Networks* [89]. This model is a deep CNN with shortcuts from each layer to every layer below it in the same "dense layer". This setup removes an issue known as the 'vanishing gradient' which has severly limited the success of deep neural nets in the past. Limited modifications have been made to this package which include:

  - *DenseNet* functions for model creation, optimisation and training have been modified to allow for dynamic adjustment of the learning rate used in *FreezeOut* approach detailed later.
  - Training and validation functions have been rewritten to capture extra results.
  - Updated *Adam* optimiser implemented from *PyTorch* library.
  - Updated syntax for PyTorch v0.41.

- *HyperOpt* - HyperOpt is an optimisation pack that can operate across both discreet and continuous hyper-parameter spaces. In this thesis it is used to allow for model creation to be repeated, altering a given set of hyper-parameters to achieve a supplied objective function. The framework for using *HyperOpt* was supplied in Max Schultz's work *Optimal Control for the Automated Design of Machine Learning Models*. Modifications were made to use PyTorch's optimisation Parzen Tree Estimator function, as opposed to the *Optunity* package. Algorithm escape clauses were added to ensure the search of the hyper-parameter space does not extend beyond what is necessary. [112]

- *Paragraph-Vectors* - A *PyTorch* interpretation of the *Doc2Vec* approach that utilises a two-tiered neural net that learns from a large lexiconic data-set the semantic mappings between words. The platform (after training) provides a set of functions that converts paragraphs into decimal vectors of set size, either by padding or aggregating. This code base is used as is, within the platform. [113]

- *PixelDA* - Release by Bousmalis and his colleges alongside their paper *Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks*. A pixel level GAN will be implemented in the domain adaptation module for continuous data sources (images). Significant modifications were made to their approach in terms of the model and how it is implemented. These changes are covered in length within the module's chapter. [114]

- *SeqGAN* - due to the nature of traditional GANs inability to function when given a discrete learning space, *SeqGAN* will be implemented to fill this space for NLP use cases. Modifications to this package concern how the loss and gradient are managed, improvements to pretraining and optimisation to *SeqGAN*'s implementation within the *PyTorch* infrastructure. [110]

- *TrAdaBoost* - A package initially designed to boost 'diff-distribution' data from the source domain and include it in the target data set in order to produce a larger, more diverse data set. Modifications were made to this code base to use it as a retraining method, in a post-domain adaptation context and updating to be compatible with the new *PyTorch* infrastructure.

- *FreezeOut* - Initially designed to accelerate the training of layered ML models such as MLPs and CNNs, by applying a scaled annealing learning rate to the layers. Only the algorithmic approach was retained in this thesis body.

3.2 Hardware Setup

While precision, recall and the associated F1 Score are the primary metrics for evaluating model types, there are secondary considerations. Model types with higher complexity overhead such as *DenseNet*s may not be suitable for all platforms. Capturing the training speed and memory usage of the source model creation and retraining modules is therefore required.

In order to ensure that results are tangible across tests, the experimental setup is conducted in a controlled environment. As GPU priority is not guaranteed and will fluctuate throughout training, GPU CUDA (NVIDIA native) libraries are not used. In applications where speed and resource management is measured, only CPU libraries will be used. GPU CUDA libraries are enabled in the domain adaptation module experiments, where speed of convergence is not measured.

Experiments were run on a DELL PowerEdge R330 Rack Server running the Ubuntu 18.02 OS, with Linux kernel version 4.15. Using 32 hyper-threaded Dual X5560 Xeon Processers (2.8 GHz turbo boosted to 3.2GHz), 96GB of DDR3 800MHz RAM. GPU resources were made avaliable for the domain adapation module. This included 2 GTX 1080Ti's with 3584 CUDA cores and 11GBs of GDDR5X RAM. This setup was local, with no network overhead.

An *Anaconda* environment is given control of 32 CPUs locking them to the environments profile, preventing their use by any other processes. In order to avoid power throttling by the servers load-balancer, the environment is run at admin level. Results from these experiments cannot be used to determine the real life speed of any of the platform's pipelines. This is due to the overrides made to hardware and security overheads, that are not practical for an in-production platform.

4

Prerequisite Modules

The following chapter will cover two modules that are not included in the two-step transductive transfer learning pipeline, but their creation and validation is important in ensuring the validity of the pipeline. This chapter will include the requirements and approach of each module. As proof of their validity, included in Appendix A and B are experimental setups, results and evaluations of the modules.

4.1 Source Model Creation

4.1.1 Problem Statement

The final "ML-as-a-Service" platform could be applied in three scenarios.

1. The traditional pipeline where the domain $\mathcal{D}$ and Task $\mathcal{T}$ are equal ($\mathcal{D}_S = \mathcal{D}_T$ and $\mathcal{T}_S = \mathcal{T}_T$).
2. The inductive transfer learning pipeline, where the domain is equal, but the label space ($\mathcal{Y}$) is different. Requiring a new mapping ($f(\cdot)$) between feature set ($X = \{x_1, x_2, ..., x_n\}$) to label $y \in \mathcal{Y} = \{y_1, y_2, ..., y_n\}$ such that $\mathcal{D}_S = \mathcal{D}_T$ and $\mathcal{T}_S = \{\mathcal{Y}_S, f(\cdot)_S\} \neq \mathcal{T}_T$.
3. The transductive transfer learning pipeline where the user provides a data set to represent the target domain ($\mathcal{D}_T = \{\mathcal{X}_S, P(X)_S\}$) that is similar to a data set already held by the platform ($\mathcal{D}_S$). In this scenario the task similarity between data sets in irrelevant.

While this work will only focus on the transductive transfer learning scenario, all three scenarios would be covered by a complete "ML-as-a-Service" platform and this module would be cruical in all 3 scenarios.

4.1.2 Requirements

The following is a set of requirements and assumptions that will ensure the validity of the *source model creator* module. The priority placed on each requirement will be applied through a MoSCoW style of requirement prioritisation, which ranks requirements as 'Must, Should, Would and Could' by order of priority.

Assumptions made by Module

- *Availability*: The data must be available within the platform or made available by the user. In order to provide this, some form of a controller and database system would need to be implemented. Implementing this would be beyond the scope of the thesis.
- *Completeness*: Referring to the trade off of the 'curse of dimensionality', all features must be fully represented in each label, else they should not be included.

Requirements imposed on Module

- *Completeness*: The model produced by the module must have converged before training has been completed.
- *Optimal*: The source model $\mathcal{M}_S$ created by the module, must be optimal $\mathcal{A}_{opt}$ for the source domain. This optimum must be within both the context of the platform and against other approaches $\mathcal{A}_{opt} = \mathcal{A}_S \geq \mathcal{A}_{other}$.
- *Data Agnostic*: In order to meet the requirements of the whole "ML-as-a-Service" platform, the source model must be invariant to the data input.
- *Dynamic*: The models produced must be layered, with each layer capable of being dynamically adjusted.

4.1.3 Approach

Approach From Literature

Based on the readings covered in the literature review, *DenseNets* were selected as this modules' output model. *DenseNets* are data agnostic CNNs with so far unbeaten results in the *ImageNet* competitions while also holding high benchmarks in NLP tests. These results have been achieved by significantly reducing the vanishing gradient problem, allowing for strong feature propagation through the deep learning architecture. As covered in the literature review, this is achieved by creating shortcuts for features to directly propagate from one layer's output to all layers of the same dense block below. Where before deep neural networks would with each layer, progressively learn more abstract and irrelevant concepts, this shortcut method keeps each layer in the dense block grounded and learning relevant features [14] [115].

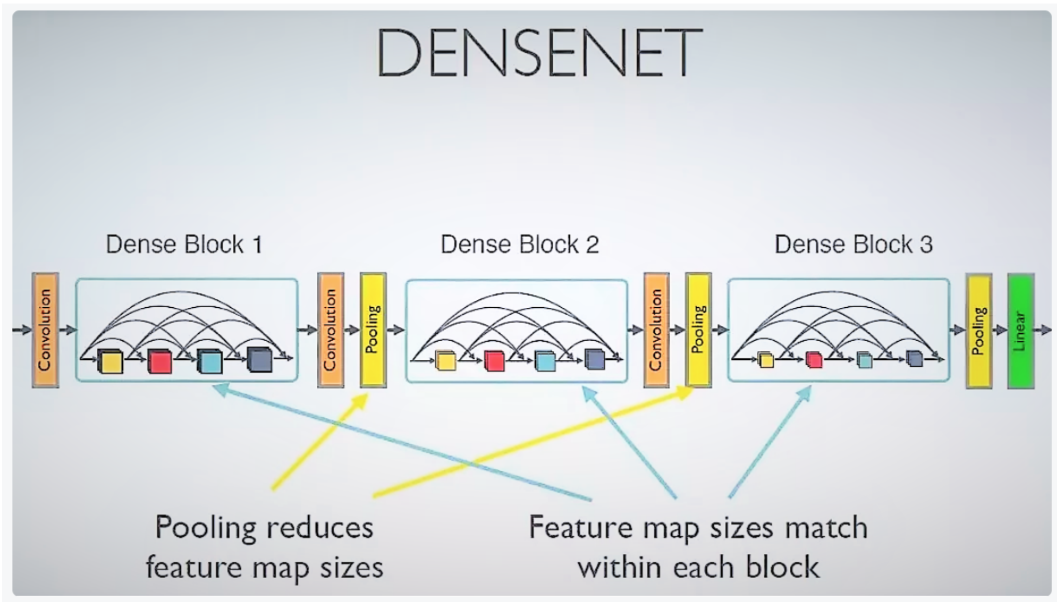

DENSENET
Dense Block 1
Dense Block 2
Dense Block 3
Convolution
Pooling
Linear
Pooling reduces feature map sizes
Feature map sizes match within each block

FIGURE 4.1: DenseNet Layout

In the feed-forward state, the models lower layers are able to receive low and high level feature activations. The flow of the data down the layers is as follows:

1. The input layer takes in i embedded dimensions.
2. The input layer then calculates i feature maps which are added to the original i vector.
3. The 2nd layer therefore receives a vector of $2i$ features.
4. This continues on through the layers.

However, this escalating dimension space is one of the major drawbacks of using *DenseNet*s. Many applications can not afford to save the large amount of space required by a *DenseNet* using larger batch sizes. Capturing the memory requirements (RAM) of training the model will be an equally important outcome as a speed and precision.

Auto-Tune Wrapper

The *Auto-Tune* framework as described in Schultz's thesis *Optimal Control for the Automated Design of Machine Learning Models* will be implemented within this module. Results of Schultz's thesis proved that the implementation of an auto-tuner was always necessary in any black box ML platform to create optimal models. As such, the *Auto-Tune* pipeline will be implemented without testing its effectiveness. [112]

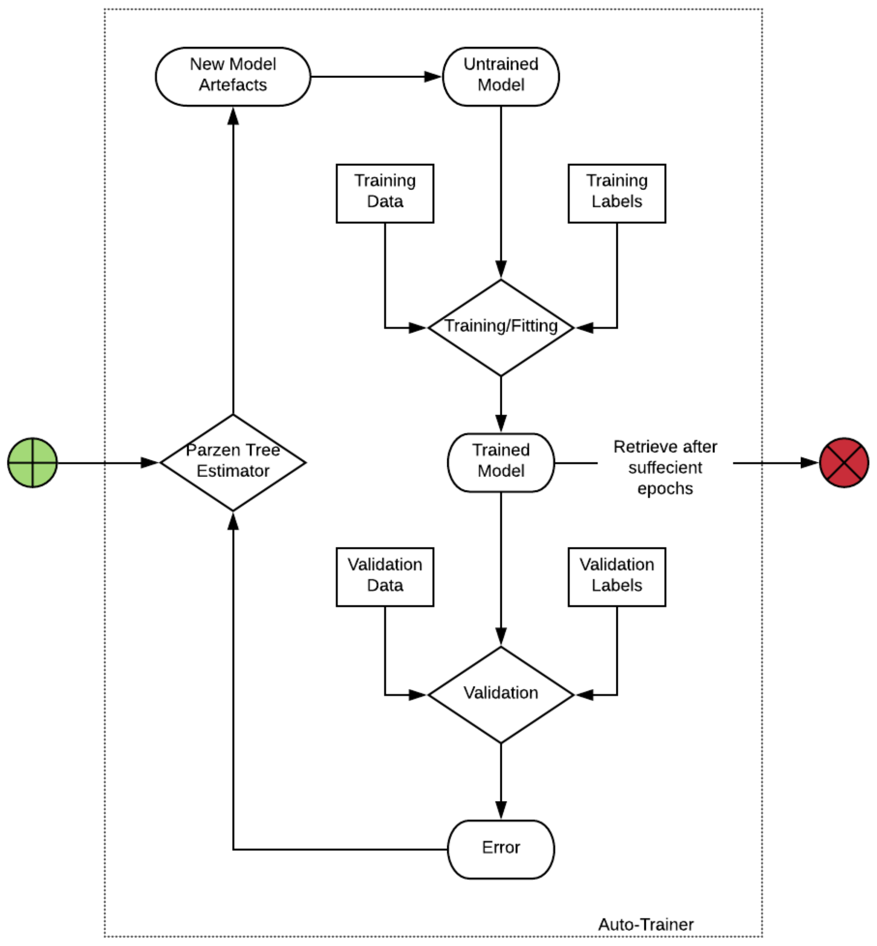


FIGURE 4.2: *AutoTune* Pipeline

The *Auto-Tune* pipeline is powered by the *HyperOpt* package. This implementation of *HyperOpt* makes use of a *PyTorch* tree-structure parzen estimator, where areas of the configuration space that produce contextually better results are more aggressively tested. This search method significantly reduces the optimisation time required to find an optimal model.

Modifications to base *DenseNet* code

The modifications to the *DenseNet* package used are mostly relate to its implementation in other modules. When updating to the most recent syntax for *PyTorch* 4.1 newer optimisers (*Adam*) and data collection layers were added to improve performance and practicability in an experimental setup and when embedded in the *Auto-Tune* wrapper. The learning rates of each layer was made dynamic, to be adjustable during training. To accomodate this change of learning rate on each epoch, the old optimisation packages was switched out with *PyTorch*'s *Adagrad* optimiser.

Which compiles after each training epoch and not at the models creation.

4.2 Pre-Processing

4.2.1 Problem Statement

Data preprocessing historically takes 5 steps:

1. **Extraction:** The required data typically does not sit in the platform's folder structure and must be accessed from local or remote repositories, typically using APIs and manual downloads.

2. **Formatting:** All standard ML models require numeral vectors as inputs. Data that's comes from sources such as SQL databases, documents or IoT devices will require formatting into usuable vectors.

3. **Cleaning:** This can include the removal of data from known bad sources, handling of missing data or the obfuscation of sensitive data.

4. **Sampling:** Representative sampling is a pre-processing technique to reduce large data sets into smaller sets of high-entropy. This can be effective when prototyping ML solutions. Shuffling and stratification is also included in this step.

5. **Transformation:**

    - **Normalisation** - Data from the same set can have divergent distributions where features can have different denominations (i.e. cents and dollars) or in image sets, using different cameras can introduce different equipment occlusions. Normalisation assists to bring feature distributions within a 0 to 1 range and remove noise, making them relative between each other.
    - **Decomposition** - In some cases breaking down data can make it easier for machine learning to understand i.e. breaking down an overall active time period into on-off periods.
    - **Aggregation** - Bringing together individual events into a single variable can assist in creating more meaningful data i.e. count of login attempts as opposed to a list of log ins.

The *Pre-Processing* module will focus mostly on transformation. It will do this at a manual level, as opposed to an automatic pipeline that would generally be expected from an "ML-as-a-Service" platform. These concessions form the basis for why these supporting modules are not considered as a part of the true contributions this thesis seeks to provide.

4.2.2 Requirements

The following is a set of requirements and assumptions that will ensure the validity of the *Pre-Processing* module. The priority placed on each requirement will be applied through a MoSCoW style of requirement prioritisation.

Assumptions made by Module

- *Integrity*: The cleanliness including missing data and incorrect feature assertions should be handled by the user before it is passed to this module.
- *Extraction and Formatting*: The form and structure of the data must be in the form that is the expected output of the *Pre-Processing* module, excluding batch creation.
- *Manual*: This module is not expected to be an automated solution, though implementing such would be a needed extension to the "ML-as-a-Service" platform.

Requirements imposed on Module

- *Standardisation*: The module must ensure all data that can be considered of similar domain in the transductive translation pipeline is in the same shape and size to be used at will by the other modules without re-transformation.
- *Data Scope*: Three forms of data could be considered by this platform: image data, multivariate sets (i.e. csv with categorical or numerical features) and lexicon data (i.e. Email bodies). This body of work and as such the module should only consider imagery and text data sets.
- *Batches*: Sub-samples of equal label and entropy should be created by the module.
- *Normalised*: All data should be normalised when passed on by the module.
- *Stratification*: Data in batches should stratified and no bias is introduced due to the distribution of data.

4.2.3 Approach:

Transforming Image Data:

All 3 modules responsible for the creation of synthetic data and models have inputs that require standardisation of the data before processing. The platform will implement a rules based pre-processing pipeline that will transform and standardise before sampling and delivery to the platforms other modules.

- **Channel Size:** - Greyscale images exists in a single intensity based 8 bit channel, however coloured imagerey can exist in 3 (RGB/YUV/HSL) or 4 (CMYK). In order to ensure standardised input, all images will be converted into the HSL colour interpretation using *PyTorch*'s Tensor transformation functionality.
- **Spatial Normalisation:** - A process focusing on expanding the dynamic range of intensity in order create deeper contrast within the images. Better contrast in images assists in feature identification. However since this platform will use CNNs, a rule-based normalisation layer isn't necessary. Instead a *BatchNorm* layer will be used instead. This layer which is included in the *DenseNet* model implemented in this thesis, will normalise each image against the spread of the images in the same batch.
- **Shuffling:** Some data sets will arrive with interpretable features to the order the images are delivered. Example a series of screen still from a cameras. In order to ensure randomisation and prevent over fitting the data sets will be shuffled within their labels using *PyTorch*'s Tensor transformation functionality.
- **Stratification:** The batches used to train models will need to ensure that an even split of classes are available in each batch run. This is done to ensure the model is not bias towards any one class. Stratification will be done during shuffling.

Transforming Text Data:

NLP requires pre-processing in order to convert document paragraphs into numeral vectors, while still retaining the semantic relevance between words and phrases. Language artefacts such as idioms and slang can significantly confuse the intent of a phrase, a problem humans have as well when communicating. A complete NLP solution that includes vectorisation would typically follow 3 steps; tokenisation, normalisation and lemmatisation. The *Doc2Vec* package will be implemented to take responsibility for this component.

The *Doc2Vec* package will reduce documents (in this case sentences) into vectors. In training, a vector 'W' for each word and a vector 'D' for each document is produced. When used as the preprocessing layer, documents can then be evaluated for their cosine closeness to the documents and words already known by the preprocessor.

This platform will use a distributed memory model form of *Doc2Vec*. This is a higher fidelity approach that takes in both the document vector and a window around each word to produce a set sized vector [113]. Noise in the form of fake words is introduced in order to prevent over-fitting. The size of the vectors will be set at 100, however for larger data sets such as reports this would in future need to be scaled. The vectors are passed along into other ML models which learn to predict sentiment.

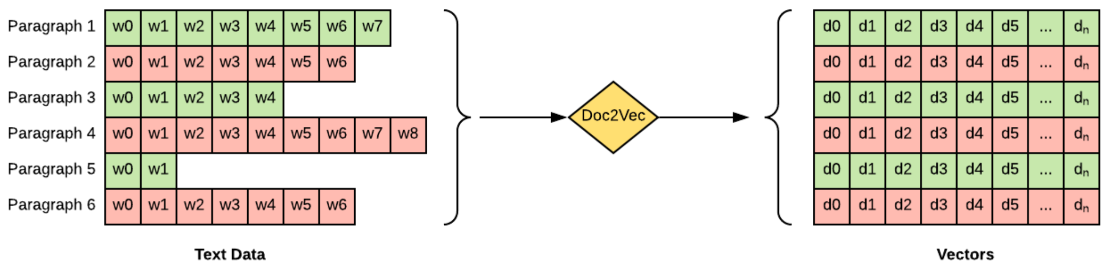


FIGURE 4.3: Doc2Vec Pipeline

When new data is introduced into the platform i.e. target data, the *Doc2Vec* can only make an approximation of the vector. This package is still in some development and how to expand the known vector space incrementally is still debated. For this platform *Doc2Vec* makes an approximation of the input.

5

Pipeline Modules

This chapter will cover the purpose, context, algorithmic and software implementation for each module in the pipeline. In some modules, multiple approaches were evaluated, however only the final implemented approach will be discussed. An alternate approaches to the retraining module is included in the appendix.

5.1 Domain Adaptation

5.1.1 Problem Statement

A domain adaptation approach to transductive transfer learning seeks to address the dissonance between domains through the creation of a new data set that sits within the target domain. This is accomplished by transforming these larger data sets in near but not adequate domains, large amounts of synthetic target data can be created. Due to a precondition of machine learning, larger data sets are able to teach models a more robust representation of the domain.

GANs which were selected to fulfil this problem statement come with their own limitations. The most significant problem introduced, is the inability to operate in a non-continuous data space (i.e. Text Data). The loss generated by the discriminator can not be passed back down to the generator. As the generator is producing, non-real valued data and can't then pass down a different data type in back propagation. Because of this, two approaches will be developed that are specific to continuous and discrete problem sets.

5.1.2 Requirements

The following is a set of requirements and assumptions that will ensure the validity of the *Domain Adaptation* module. These requirements extend over both the continuous and discrete approaches and are sourced from the higher requirements imposed on platform as a whole. The priority placed on each requirement will be applied through a MoSCoW style of requirement prioritisation.

Assumptions made by Module

- *Conformity*: Both the source and target data set, must have the same shape and size. Normalisation within and between the data sets should exist to ensure performance.
- *Representative*: The target data must supply a complete representation of the target domain.
- *Equivalence*: The source and target data set must have the same feature space (i.e. image to image). The similarity of the distribution will affect the speed of convergence and final validity of the synthetic data. ($\mathcal{D}_S \sim \mathcal{D}_T$).

Requirements imposed on Module

- *Erudite*: GANs must learn the target domain, and not just learn to copy the target data. Making them able to produce new synthetic data in the target domain, and not copies of the already available data.
- *Optimal*: The target domain must be a significant subset of the synthetic domain after the GAN converges.
- *Completeness*: The GAN must have reached convergence in training before synthetic data is passed along the pipeline.

5.1.3 Approach - Continuous Data

The research community has produced a large amount of papers providing to improvements atop the traditional GAN architecture that was covered in the literature review. As a starting point, this paper would take significant note of Bousmalis' et al. work in *Unsupervised Pixel–Level Domain Adaptation with Generative Adversarial Networks* [114]. The core components of this approach centres around building a loss function based on the pixel-by-pixel dissonance between the synthetic data and the target date. Typically in the continuous domain, these loss functions are built based on the image as a whole.

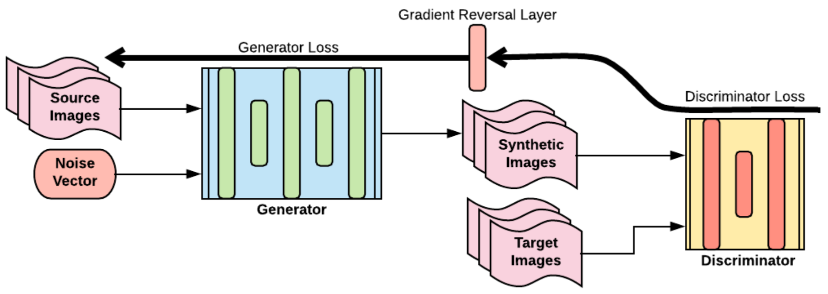


FIGURE 5.1: The traditional GAN learning pipeline.

The field of study that GANs exists in sees cited improvements released almost monthly and Bousmalis's approach is now over three years old. The following changes were made incrementally and retained after a basic MNIST to MNISTM (covered in subsequent chapters) test did not see a depreciation in performance. All these improvements come from peer reviewed papers, alluding to their validity across all applications.

- All inputs are made between -1 and 1. Then a tanh layer is used in the last layer of the generator. [108]
- A maximum log loss function is used, as opposed to a minimum log loss function to reduce the vanishing gradient effect early on. [108]
- *BatchNorm* is used, where all synthetic data or all real samples from the batches is bundled together to the discriminator. [116]
- Does not use sparse gradients, in place of layers such as ReLU, *LeakyReLU* layers are used instead. [117]
- An ADAM optimiser is used. [116]
- Decay noise vector during training. [118]

With these alterations in place, the following models were built. It's important to note at this point that in terms of the pipeline and model's individual layers, little beside the use of a residual block and the overall structure remained of Bousmalis's work [114].

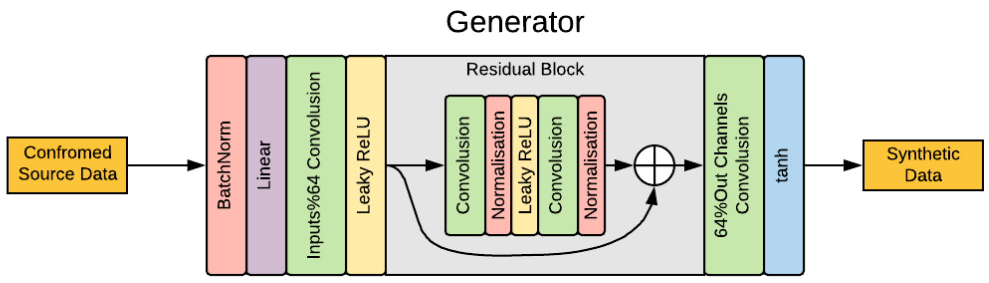


FIGURE 5.2: Modified *pixel-wise* Generator.

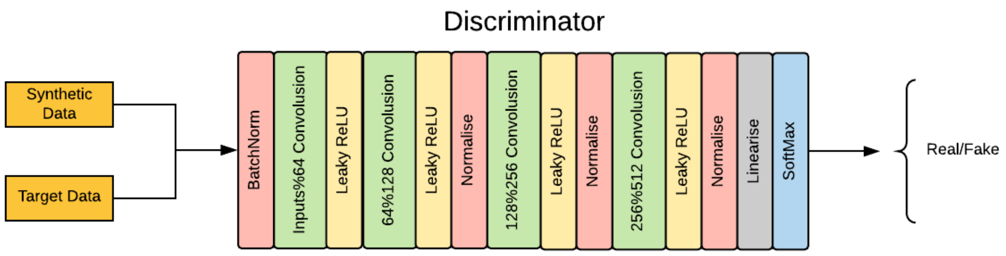


FIGURE 5.3: Modified *pixel-wise* Discriminator.

As with most machine learning models, the loss that the discriminator generates and passed back back to the generator takes the form of an optimisation function. The loss here however is calculated as a masked pairwise mean squared error (PMSE) of the pixels and not the mean different of the images as a whole.
For a binary masked $\mathbf{m}$, the PMSE is:

$$\mathcal{L}_D(G) = \mathbb{E}_{x^s,z}[\frac{1}{k}||(x^s - G(x^s, z; \theta_G)) \circ \mathbf{m}||_2^2 - \frac{1}{k^2}((x^s - G(x^s, z; \theta_G)^T \mathbf{m})^2] \tag{5.1}$$

The above states, that the loss of the discriminator is for any number of pixels k of input x, the mean difference L^2 of the surrounding pixels '$||_2^2$' between the synthetic and target images. This approach focuses on the shape and form, before it considers colour and intensity.
This pixel level approach yields the following advantages:

- **Task Invariant Architecture:** Conventional domain adaptation solutions require that the task model (the final classifier provided by the platform to the user) embedded within the feature representation transfer layers. This approach will provide a pipeline that will allow for the substitution of different task models.

- **Label Space Semi-Invaraince:** The GAN learns the form and pose within the image, aspects that are not entirely bound by the label space. As multiple aspects can be inclusive across multiple labels. This allows for a slight label imbalance, as long as the disadvantaged labels are similar to a more defined label space.

- **Inferred Learning:** The GAN built in this approach does not learn how to convert the source data into target data, instead it learns the transformation between the source and target domain. This is especially important in a continuous learning space where new source data can be introduced to update the target model.

5.1.4 Approach - Discrete Data

A constraint on the previous GAN and other conventional GANs, that prevents their implementation as generic solutions for all data types, is that the data must exist in a continuous numerical space. GANs are not able to generate the required gradient from loss in discrete space. This is a problem when considering the NLP problem space.

As discussed in the prepossessing module, the GAN will be receiving a numerical data set. However, as a word only exists for a single coordinate within this real value space, all non-word coordinates are empty space (typically received as "<PAD>"). The traditional GAN approach can still operate in this space, albeit with negligible success. As the space between valid coordinates is so large that no gradient can be successfully formed. An approach of inflating coordinates into CI spaces was tested and found to be unsuccessful. As unique words with a large distance from others were given much larger CI spaces, over-biasing the model's training towards them.

Instead a three tiered GAN, that uses a policy gradient to travel loss back down to the generator from discriminator was implemented. Policy gradients make a real-valued estimation of the loss, bridging the gaps in the discrete space and thus emulating a continuous data space to work in. The approach implemented is heavily based on the *SeqGAN* package as described in the paper from Yu et al. *SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient* [110]. Improvements have been made on the model, but its core framework remains the same.

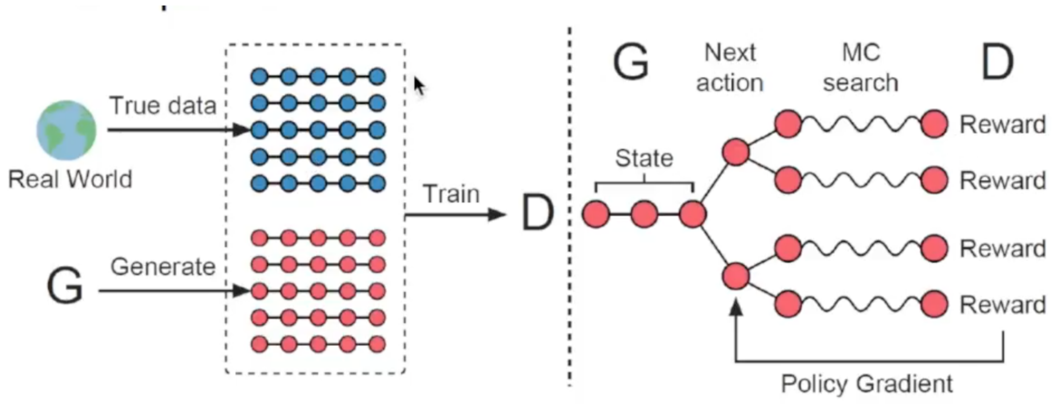


FIGURE 5.4: *SeqGAN* model

Given a list of real world sequences, the generative model G_θ will produce its own set of of sequences $Y_T = (y_1, ..., y_t, ..., y_T), y_t \in \mathcal{Y}$ where $\mathcal{Y}$ is the known vocabulary ($\mathcal{Y} = \mathcal{Y}_S + \mathcal{Y}_T$). This generation process is based on the "encoder-decoder" setup as discussed in the literature review. For a given time-step t, a state s_t is produced by the generator through an implementation of the REINFORCE algorithm, a set of the next likely states s_{t+1} from the previous state.

$$Q_{D_\phi}^{G_\theta}(s = Y_{1:t-1}, a = y_t) = \begin{cases} \frac{1}{N}\sum_{n=1}^{N} D_\phi(Y_{1:T}^n), Y_{1:T}^n \in \mathbf{MC}^{G_\beta}(Y_{1:T}; N) & t < T \\ D_\phi(Y_{1:T}) & t = T \end{cases} \tag{5.2}$$

$Q_{D_\phi}^{G_\theta}(s_0, y_1)$ is an action-value function used to describe the the reward from taking action a from starting state s.

A Monte-Carlo search on all possible states as created by the generator is conducted and passed to the discriminator.

$$\{Y_{1:T}^1, ..., Y_{1:T}^N\} = \mathbf{MC}^{G_\beta}(Y_{1:t}; N) \tag{5.3}$$

Where $Y_{1:T}^i$ is sampled at each state. After each state is searched and passed to the discriminator, its appropriateness is evaluated to an optimisation function.

$$\min_\phi \text{ is } \mathbb{E}_{Y\ P_data}[\log \mathcal{D}_\phi(Y)] \text{ is } \mathbb{E}_{Y\ G_\theta}[\log(1 - \mathcal{D}_\phi(Y))] \tag{5.4}$$

After each state is passed through the discriminator and a reward is built, a policy gradient is then built from the losses. This gradient is travelled back down to the generator and used to train it.

$$\delta J(\theta) = \mathbb{E}[R_T|s_0, \theta] = \sum_{y_1 \in Y}^{X} G_\theta(y_1|s_0) \cdot Q_{D_\phi}^{G_\theta}(s_0, y_1) \tag{5.5}$$

The maths of the *SeqGAN* has not been modified beyond the implementation of a tanh layer in the discriminator as opposed to this models log softmin layer. This tanh layer follows the same rationale as the pixel-wise GAN, in order to select the "best-of" as opposed to remove the "worst-of" [108]. However in order to promote parallel processing and implementation in a GPU CUDA environment, the REINFORCE algorithm and MC search was altered to run per state as opposed to iteratively along each possible branch before being collected at the end. This improvement theoretically should've reduced the processing time by T^T. However, only a 20% reduction was observed, indicating a significant overhead that had not been addressed.

Algorithm 1 Sequence Generative Adversarial Nets

```
INPUT: generator policy $G_\theta$; roll-out policy $G_\beta$; discriminator $D_\phi$; a sequence data set $S = X_{1:T}$
  Initialise $G_\theta$, $D_\phi$ with random weights $\theta, \phi$.
  Pre-train $G_\theta$ using MLE on $S$
  $\beta \longleftrightarrow \theta$
  Generate negative samples using $G_\theta$ for training $D_\phi$
  Pre-train $D_\phi$ via minimising the cross entropy
  repeat
    for $i$ in g-steps do
      Generate a sequence $Y_{1:T} = (y_1, ..., y_T)$ $G_\theta$
      for $t$ in $1 : T$ do
        Compute $Q(a = y_t \; s = Y_{1:t\text{-}1})$
      end for
      Update generator parameters via policy gradient
    end for
    for $j$ in d-steps do
      Use current $G_\theta$ to generate negative examples and combine with given positive examples S
      Train discriminator $D_\phi$ for k epochs
    end for
    $\beta \longleftarrow \theta$
  until SeqGAN converges
```

The pretraining of the algorithm was also an area of improvement. The initial implementation of the *SeqGAN* used a naive MLE solution to train both the generator and discriminator. This was modified to include the policy reward gradient from the discriminator to act as an mitigating element, that when the discriminator loss began to significantly decrease, the pre-training was halted. This was done in order to prevent the same over-training problem identified in the initial paper that was not addressed [110].

5.2 Retrainer

5.2.1 Problem Statement:

In this half of the two-step pipeline, with the domain adaptation module having successfully created a sizeable representation of the target domain, the retrainer module will create models for this new domain by retraining models previous trained on the source domain. This process of retraining is based upon how ML models (especially CNN models) store their knowledge on a layer by layer basis of different abstractions. The lower layers of models trained on similar domains will share a lot in common as they are have to learn almost identical features and as the layers go higher and the features become more abstract it is these layers that need to be retrained.

Then, by the same principles of dropout and jittering, a model that is trained on the source domain and then retrained for the target domain will have a larger domain of knowledge, having retained some of the knowledge from the source domain, but now being fully trained for the target domain. This will mean that validation and real world target data, that was not caught in the initial training, will likely still be caught by this more robust model that has been built.

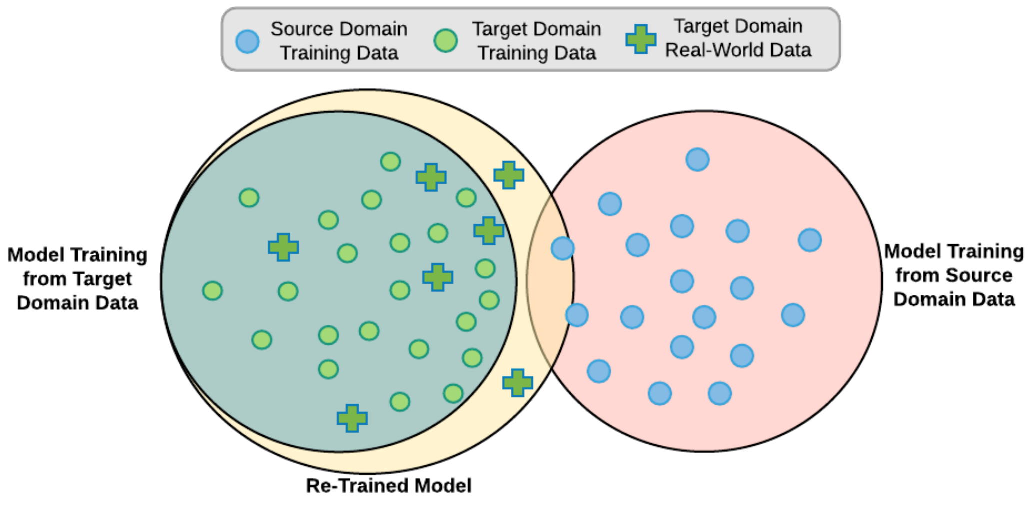

Source Domain Training Data
Target Domain Training Data
Target Domain Real-World Data
Model Training from Target Domain Data
Model Training from Source Domain Data
Re-Trained Model

FIGURE 5.5: Area of knowledge of source, target and retrained models.

While not applicable to this pipeline, this retrainer module is also capable of acting as a solution to the inductive transfer learning problem. This is still relevant to the transductive transfer learning pipeline. As while the inductive problem is concerned about the capturing of task relevant data when retraining, the transductive problem is concerned about capturing both domain and task relevant data. A solution that is able to capture both types of data will have a better outcome.

5.2.2 Requirements

The following is a set of requirements and assumptions that will ensure the validity of the *Retrainer* module. The priority placed on each requirement will be applied through a MoSCoW style of requirement prioritisation.

Assumptions made by Module

- *Pretrained*: The source models passed to this module must have previously been trained to convergence in the source domain.
- *Equivalence*: The effectiveness of retraining will be correlated to the divergence of the source and target domain. As such the domains of the two data sets should not be excessively different.
- *Ratified*: The synthetic data passed to the module, should be a close representation of the target domain.

Requirements imposed on Module

- *Optimal*: The final model produced by this module must have reached convergence.
- *Extension*: The total domain knowledge of the final retrained module, should be greater or equal to the domain knowledge of a from-scratch trained model.
- *Speed*: As a hallmark of positive transfer learning, the retraining process should be faster than a from-scratch approach.

5.2.3 Approach:

There are two core approaches to retraining models in the transductive transfer learning context. Either a data boosting or model augmentation process is conducted.

Data Boosting - *TrAdaBoost*

Data boosting involves reweighting the data set to prioritise data that positively improves the knowledge of the domain, and reduce data that is negative and such is likely noise. A data boosting approach known as *TrAdaBoost* was explored as the first option for this module [119]. The original purpose of *TrAdaBoost* was to train the model on data from multiple data sets, but only using data that is within the same domain as that of the target data. This allows the training process to focus on high entropy data for the target task and domain. Data level approaches are also agnostic to the model type. Which in the context of the platform's generic pipelines, enables model types to be swapped out without effecting the performance of the transfer learning pipelines. A full account of the implementation, experimental setup and results for this approach can be found in Appenedix C. To summarise, the experiments discovered a key issue with this approach that was exasperated when the source and target domain were too alike. The original *TrAdaBoost* paper tested on diverging data sets that did not start with a spread that was closely related. Because of this a model augmentation approach was instead favoured.

Model Augmentation - *FreezeOut*

Model augmentation involves changing elements of the model in order to produce a better outcome. The hyper-parameter auto-tuner pipeline contained within the *Source Model Creator* module is a model augmentation approach to hyper-parameter tuning. Here the *FreezeOut* algorithm was used [120]. Initially designed to quickly train models without significant decreasing that model's accuracy. It was instead altered for to become apart of a transfer learning approach.

In a layered ML model, the lower layers learn predominately domain invariant features. These layers can capture both domain and task relevant features that are common between different data sets. By 'freezing' these layers early in the training cycle, the model retains knowledge that is relevant for the both the new target domain and old source domain. Features that are highly abstract and specific to the source domain are found in the higher layers. *FreezeOut* leaves these layers longer in the training space, allowing them to be retrained using the target domain data. This model augmentation process, is not agnostic to the model type. However, in the context of this platform where all models will be *DenseNet*s, this is acceptable.

The only component taken from the original *FreezeOut* package was the annealing equation iterated within each layer at the beginning of every epoch.

Equation 5.6 Annealing Learning Rate - Scaled and Cubic

$$\alpha_i(t) = 0.5 * \alpha_i(0)(1 + cos(\pi t / t_{i_{(cubed)}})) \tag{5.6}$$

$\alpha_i(0) = \alpha/t_i$ is the initial learning rate of each layer (i), where α is the base learning rate. In the original implementation of *FreezeOut* cubic scaling was performed to ensure each layer travelled equally in the training space.

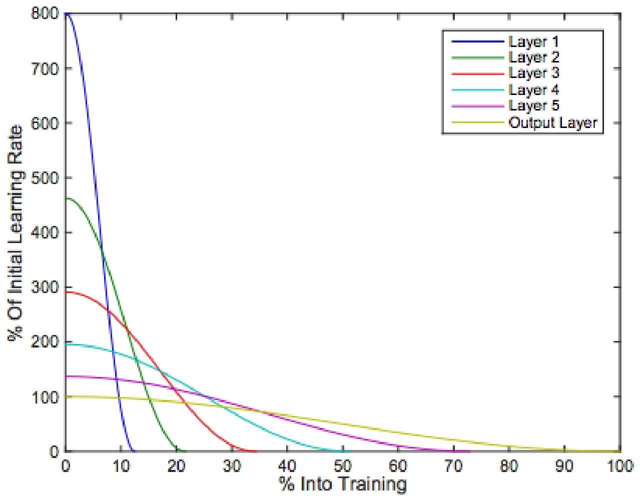


FIGURE 5.6: Scaled Cubic Learning Rate

Algorithm 2 FreezeOut

Input: Training data set $\mathcal{S}$(in the domain adaptation process it is the synthetic data set) split into B batches of size I, the task T, a pre-trained multi-layer perceptron and a maximum number of iterations N .

Procedure:

```
for t = 1,...,N do
    A. Alter Learning Rate for each layer (i) by an annealing rate:
       $\alpha_i(t) = 0.5 * \alpha_i(0)(1 + cos(\pi t/t_i))$      (5.7)
    for b = 1,...,B do
        for d = 1,...,S do
            1. Propagate s through model.
            2. Capture error e at output.
            3. Back-propagate e through layers to retrain.
        end for
    end for
end for
```

Output: Retrained Model for target domain and task.

Automating the Selection of Initial Weights In order to find the best initial learning rates, an automated search of the possible hyper-parameter space, similar as to that used in the automated selection of hyper-parameters in the *Auto-Tune* pipeline is taken. Using *HyperOpt* with a Parzen Tree Estimator to narrow in on the best starting learning rate for each layer.

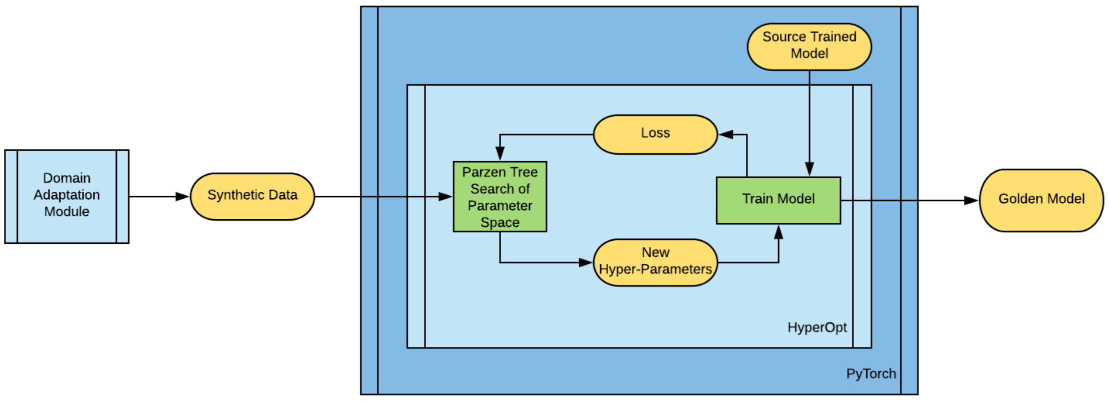


FIGURE 5.7: Software pipeline of retrainer module using *FreezeOut*.

The computation cost of *FreezeOut* is linked to the number of training iterations per layer.

Equation 5.8 Computation Cost of FreezeOut

$$\mathcal{C}_f = \sum((1 + t_i) \times c_i \times n_{itr}) \tag{5.8}$$

Where: n_{itr} is the total number of iterations and c_i is the number of layers.

This approach will significantly increase the computation time of the *FreezeOut* module and the overall computation cost needs to be limited to a reasonable level. This level will be discovered experimentally, but an initial starting point of 50 times the typical will be set as a maximum.

6

Experimental Setup

6.1 Testing Data

There are two types of data that will be evaluated in these experiments, continuous (images) and discrete (text). Nearly all types of data such as acoustic/wave, financial (i.e. stock reports), or natural (i.e. genomes) sit within these two categories.

6.1.1 Image Recognition

The following peer-reviewed data sets are commonly used in the machine learning field to rate the advancements made by new model variations, hardware accelerators and augmentation methods. The following data sets will be used in object detection problems.

MNIST

MNIST was formed from the combination of two data sets NIST1 and NIST3. The MNIST data set is a basic set of 70,000 images of handwritten numbers (60,000 training, 10,000 testing). These are simple greyscale images with distinct foreground and background.
Avaliable at: *http://yann.lecun.com/exdb/mnist/*

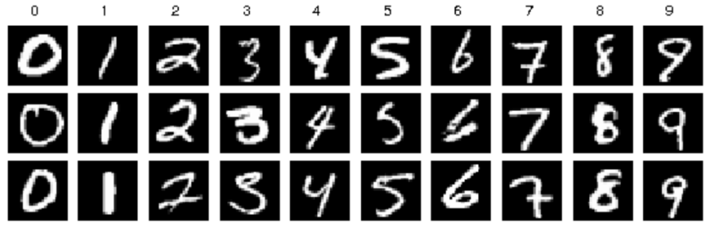


FIGURE 6.1: The MNIST data set

MNISTM

MNIST-M contains 9000 coloured hand-written digits on coloured textured backgrounds. Data sets of these nature are typically larger than 9000 (i.e. MNIST is 70,000 and has lower channel complexity). The high variance and low amount of the data will be useful for testing the bias of the models and approaches used in these experiments.
Available at: *http://users.itk.ppke.hu/ horan/mnistm/*

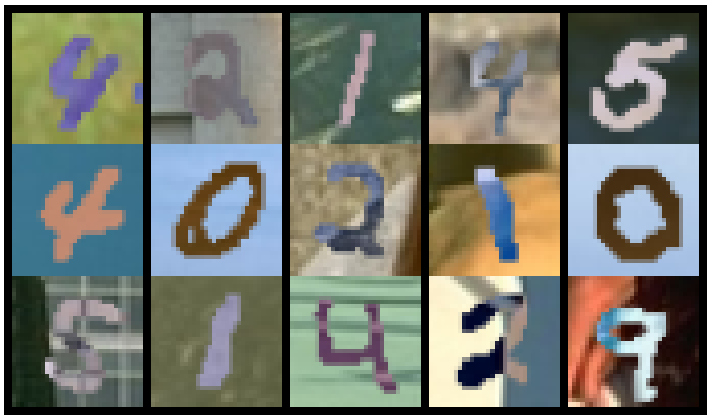


FIGURE 6.2: The MNISTM data set

CIFAR10/100

A collection of 60,000 32x32 images split into 10 classes, representing 10 common objects: airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. The low resolution nature of the images makes it useful in the development phase of machine learning.
Avaliable at: *https://www.cs.toronto.edu/ kriz/cifar.html*

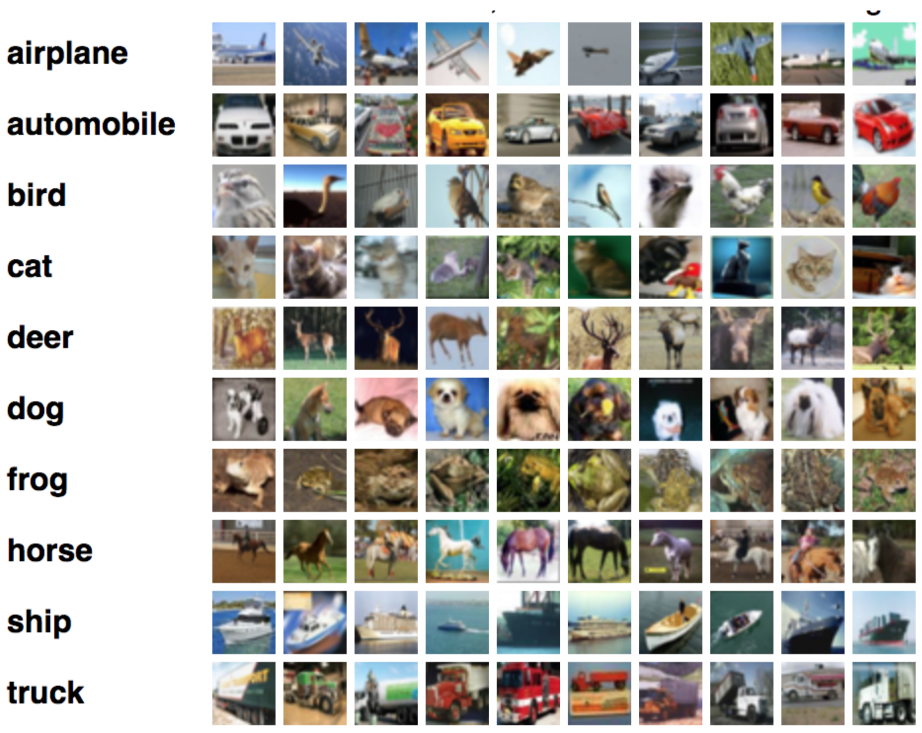


FIGURE 6.3: The CIFAR-10 data set

The CIFAR-100 data set extends the CIFAR-10 data set into 100 classes. However, the previous class size of 6000 images is reduced to 600. This problem space will be similar to the size of the MNISTM data set, except for a more complicated problem space.

6.1.2 Text Dataset

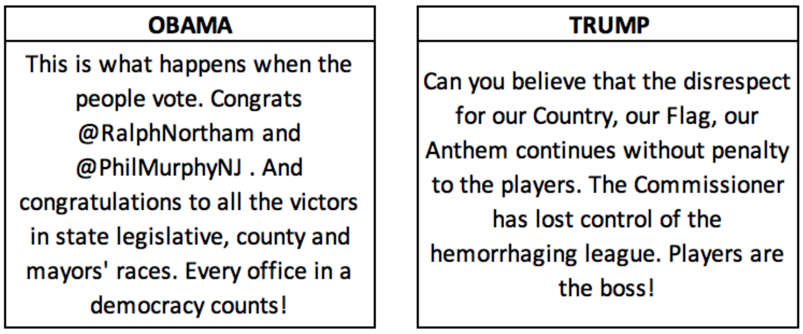
For lexiconic data type tests, the most important step happens before the model can be tested. For the most part, sentiment analysis is a more complex task at the data level than image recognition.

IMDB Movie Review Database

This data set contains 50,000 reviews (25,000 training samples, 25,000 testing samples) . The data set is split evenly into 25,000 positive reviews and 25,000 negative reviews. An additional 50,000 unlabelled documents are also included, which can be used to test unsupervised learning problems. Included in this data set is a pre-created Bag of Words vectors that can be used to compare the preprocessing modules approach against a non-semantic BoW approach [121].
Available at: *http://ai.stanford.edu/ amaas/data/sentiment/*

Tweets - Obama and Trump

Interestingly while Ex-Pres. Obama and current Pres. Trump have very different political and personal attitudes, the content of both Presidents twitter feeds have startlingly common word choice. "Action" and "Change" sentiment features equally often (69%) in their speeches. So while their vocabulary and the tone of speech may differ, their intent is very similar. Because of this, domain adaptation between these two sets provides a unique opportunity for accessing the ability of the domain adaptation module's ability to transfer vocabulary between domains while retaining semantic intent. [122]
Available at: *https://benellerby.github.io/trump-obama-tweet-classifier/*



OBAMA	TRUMP
This is what happens when the people vote. Congrats @RalphNortham and @PhilMurphyNJ . And congratulations to all the victors in state legislative, county and mayors' races. Every office in a democracy counts!	Can you believe that the disrespect for our Country, our Flag, our Anthem continues without penalty to the players. The Commissioner has lost control of the hemorrhaging league. Players are the boss!

FIGURE 6.4: Obama Vs. Trump Twitter Feed

Orcale NLL

Oracle NLL is a pre-vectorised NLP corpus that uses all possible words in the English language in contextually accurate sentences. It is essentially the largest generic vocabulary data set that has been made. On the inverse though it is only very rarely used and doesn't feature in many papers beyond the domain adaptation and translation research space [110]. This is likely due to its unnatural syntax that only makes it useful when it's being used to convert into other more natural texts.
Available on Request.

Accenture Ticket Datasets

This is an industry thesis and to prove the relevance of this work in the industry, a use case for the NLP problem space has been selected that uses an Accenture ticket data set. The data set of 23832 tickets is composed of 2 major categories (Incident and Access Request) and 32 sub categories. Only the major categories will be considered for the major use case.
Unavailable due to sensitive nature.

6.1.3 Common Evaluation Metrics

Bilingual Evaluation Understudy - BLEU Score

Traditionally, BLEU scores have been used as a quantitative measurement for translated texts between languages. In the NLP context it is a reliable metric to compare generated sentences to a reference sentence. In domain adaptation this reference sentence is the target domain the GAN seeks to emulate.

BLEU scores are evaluated by comparing n-grams (windows) in the synthetic data set against the reference target data set. Since this platform is using a vector representation for sentences, the cosine closeness of word vectors is used as a metric to evaluate the closeness of the sentences. The BLEU score is a rating from 0 to 1 for average of the closeness of the n-grams. The standard size for n-grams for larger bodies of work such as speeches is 4-6, for shorter pieces such as ticket headers it's 2-4.

This platform will use the Python Natural Learning Toolkit (NLTK) to calculate the BLEU scores. The corpus compoenent of NLTK will be used, which allows for a single sentence (the synthetically generated sentence) to be compared against a reference list of sentences (the target data set).

MNIST to MNISTM

The MNIST to MNISTM domain adaptation test has been become an unofficial, yet often used evaluation for domain adaptation approaches. Its popularity has sprung from a significant need for a 'clinical'-like test that includes both the complexities associated with images in domain adaptation, while still having the flexibility to allow for each DA problem to be addressed. The current holder of this benchmark belongs to the authors of *pixelDA* Bousmalis et al. [114] at 98.2% accuracy.

6.2 Experiments

A series of experiments were conducted that ensured the validity, accuracy and reliability of the approach undertaken for each module. In the case of the *Retrainer* module where a second data-level approach was taken, those results can be found in the appendix.

Only experiments from the pipeline modules are detailed here. The non-novel, but still important results that are gathered by evaluating the *Source Model Creator* and *Pre-Processing* modules are available in the appendix.

6.2.1 Domain Adaptation with Continuous Data: Pixel-Wise GAN

MNIST to MNISTM Benchmark

For this experiment, a MNIST to MNISTM domain adaptation problem will be conducted. Here the 70,000 MNIST data set will be passed to the GAN as the source data, the 9,000 MNIST-M will be passed as the target data. For this experiment an addition will be made to the typical domain adaptation pipeline. A discriminator which has been pre-trained on the whole target data set will be used to evaluate the synthetic data and give it a "confidence score". This discriminator will be referred as the "task model".

The MNIST-M data set will be split into a set of 9000,2000,500 and 200 image samples, with an even split in each label. The preprocessing will produce stratified batches of 20 images (in the 200 image sample set this will correspond to 2 of each label in a batch of 10 batches). The below test will be run for each given set.

1. Pre-train the discriminator to create the task model.
2. Adversarially train the GAN using the sampled source and target data set.
3. At each epoch, pass the synthetic data to the task model to produce a confidence score.
4. Also capture the loss from the generator and discriminator at the end of each epoch.
5. Halt once the GAN reaches convergence, as indicated by low momentum of change from the results of the task model.

The above test is repeated 10 times to gauge the maximum, minimum, average accuracy and average batches til convergence of the GAN to produce at each target data set amount.

Validity of Synthetic Results

A second test ensures that the generator is learning to emulate the domain, and not just copy the data. To do this, each synthetic image produced by the generator after it reaches convergence is then mapped against every image of the target domain.

1. Map the synthetic data against the target data.

2. Perform an $L2$ nearest neighbours calculation
3. Qualitatively review a random set of synthetic images against their nearest neighbour in the target data set from the same label.
4. Quantitatively review by checking the standard deviate of the two data sets

6.3 Domain Adaptation with Discrete Data: *SeqGANs*

Oracle NLL dataset

This test using the Orcale NLL data set was in the original implementation of *SeqGAN*. Repeating this test and making comparisons to the original, is a strong evaluation that both implementation of the core *SeqGAN* model and the improvements made are valid and optimal.

1. Pretrain Generator and Discriminator on Oracle NLL data set (Vectors).
2. Adversarialy train generator using discriminator.
3. Capture and ensure the loss of the generator decreases throughout and does not undergo mode collapse during adversarial training.

BLEU scores for Obama Sentence Generation

This test requires the modified *SeqGAN* to create Obama's sentence structure and vocabulary using Trump's own as a domain reference. The synthetic sentences's difference from the target domain is measured with a BLEU Score. 2,3 4 n-gram BLEUs will be captured.

1. Vectorise the sentences using *Doc2Vec* from preprocessing module.
2. Using Obama's sentences as the reference data set, train the *SeqGAN* to transform Trump's tweets (sans url and tweet specific language).
3. At the start and after each epoch perform a validation run. By getting the BLEU score of the synthetic data.

Next the validity of the output data must be accessed, ensuring that the GAN is learning, but not copying the target data. This is achieved by taking a Monte-Carlo statistical approach and repeating the experiment a thousand times. The frequency of the Obama's most popular 100 words in the target data set is measured against the synthetic data set.

6.4 Retrainer: *FreezeOut*

This module is to be implemented in the transductive transfer learning pipeline. However, early in development it was noted that the retrainer module could also be applied to the inductive transfer learning scenario ($\mathcal{D}_S = \mathcal{D}_T$, $\mathcal{T}_S \neq \mathcal{T}_T$). Testing the inductive transfer learning capability of the retrainer will not only highlight a possible extension of the platform's uses, but also access the ability of the platform to retain task specific data. As specified in the previous chapter, this is important as even in the transductive transfer learning pipeline, retaining task specific data will lead to higher accuracy models.

The results of concern in these tests are the 3 hallmarks of positive transfer learning: higher start, higher convergence and quicker training in comparison to an equivalent from-scratch model. Speed is also of concern, as *FreezeOut* should be expected to reduced at maximum the training time by 22% as per the original implementation [120].

6.4.1 Retaining Task Relevant Features: MNIST on Different Tasks

This test will evaluate the inductive transfer learning ability of the *FreezeOut* algorithm. This experiment will access the ability for *FreezeOut* to capture task specific data, and how much that is affected by the scaling of the initial learning rates. Accessing this is important to understanding if the *HyperOpt* wrapper is necessary.

1. Train a model on the MNIST data set for the multi-class task of classifying each number using the *Auto-Tune* module.
2. Replace the output layer of the model with two nodes, and using *FreezeOut* retrain the model for the task of identifying 0's.
3. Perform the same test using different initial learning rates.
4. Train another from scratch model on the MNIST data set for identifying 0's.
5. Repeat the above tests for a multi-class task using numbers grouped in sets ($\{0, 1, 2\}$, $\{3, 4, 5, 6\}$, $\{7, 8, 9\}$).
6. Capture precision of the *FreezeOut* model against the baseline models and the speed boost from using *FreezeOut* vs. from scratch.

6.4.2 Retaining Domain Relevant Features: Synthetic MNIST-M to True MNIST-M

This test will evaluate the ability of the *FreezeOut* algorithm to retain the features of the source trained model that are relevant to the domain of both data sets. As with the previous experiment the affect scaling has on the training process for this context will be investigated as well.

In order to ensure the results of this experiment are independent of the quality from the synthetic data set, three data sets are used.

1. **100% Target Data:** A representation of the best domain adaptation outcome using the MNIST-M data set (which will only be of the size 9000.
2. **0% Target Data:** A representation of the worst domain adaptation outcome using the MNIST data set.
3. **'High Confidence' Synthetic Data:** using data that is the best produced by the domain adaptation module, that has $> 95\%$ confidence score.

By testing with all three data sets it is possible to confirm the best, worst and expected outcomes from the module.

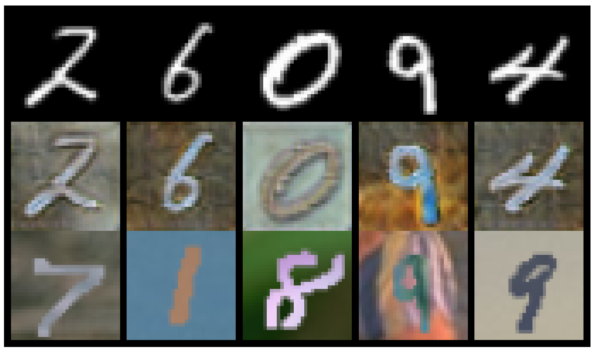


FIGURE 6.5: Data sets (top: 'Nil-Confidence Synthetic Data (MNIST Data)', middle: 'High Confidence Synthetic Data' bottom: 'Perfect Synthetic Data (MNIST-M Data)')

The following experiment will be undertaken:

1. Train a model on the MNIST data set for the multi-class task of classifying each number using the *Source Model Creator* module.
2. Using *FreezeOut*, retrain the model on the three synthetic data sets (MNIST, MNISTM and true synthetic). Repeat this for different initial learning rates.
3. Train from scratch comparative models using the full MNISTM data set, a smaller limited MNISTM data set and the true synthetic data set.
4. Compare the precision, training time and initial accuracy of all the models.

7

Results and Evaluation of the Modules

7.1 *Pixel-Wise Generative Adversarial Network*

7.1.1 Results

Evaluating GAN for different target data amounts

With the batch size set initially to 20 images the below results were captured:

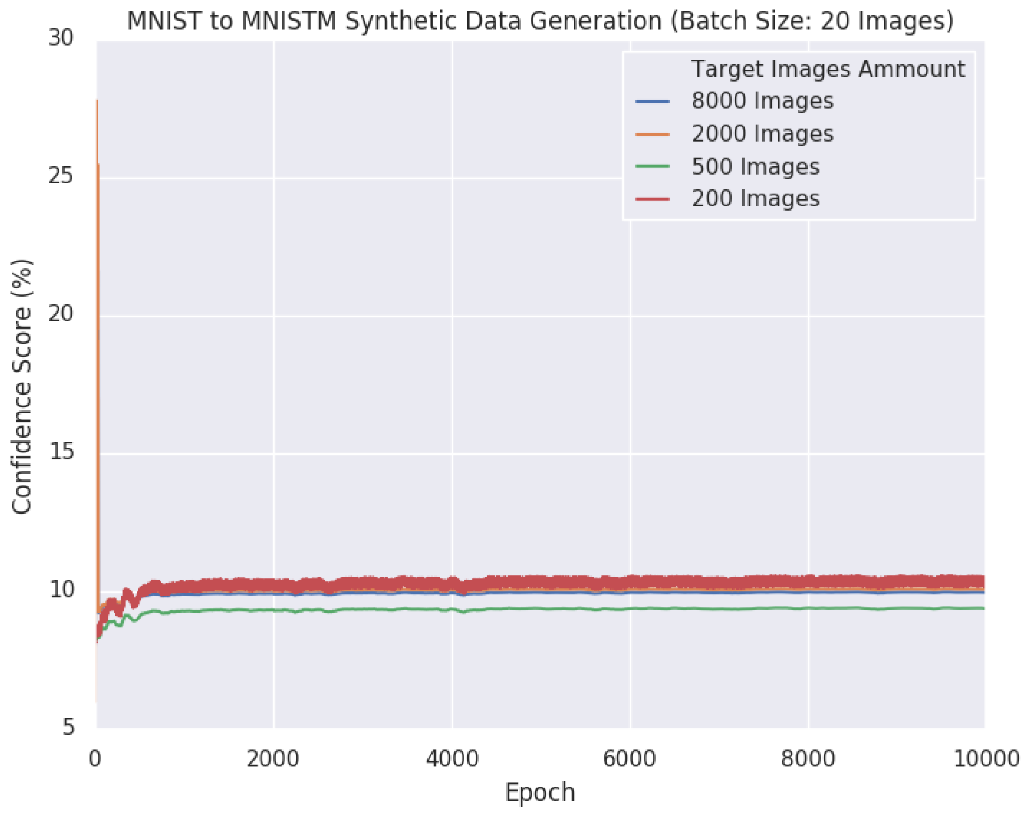


FIGURE 7.1: Target model's confidence score (%) of synthetic data using a batch size of 20 images.

Based on these results, the batch size was increased to 100 images, which significantly improved the accuracy task classifier on the synthetic images.

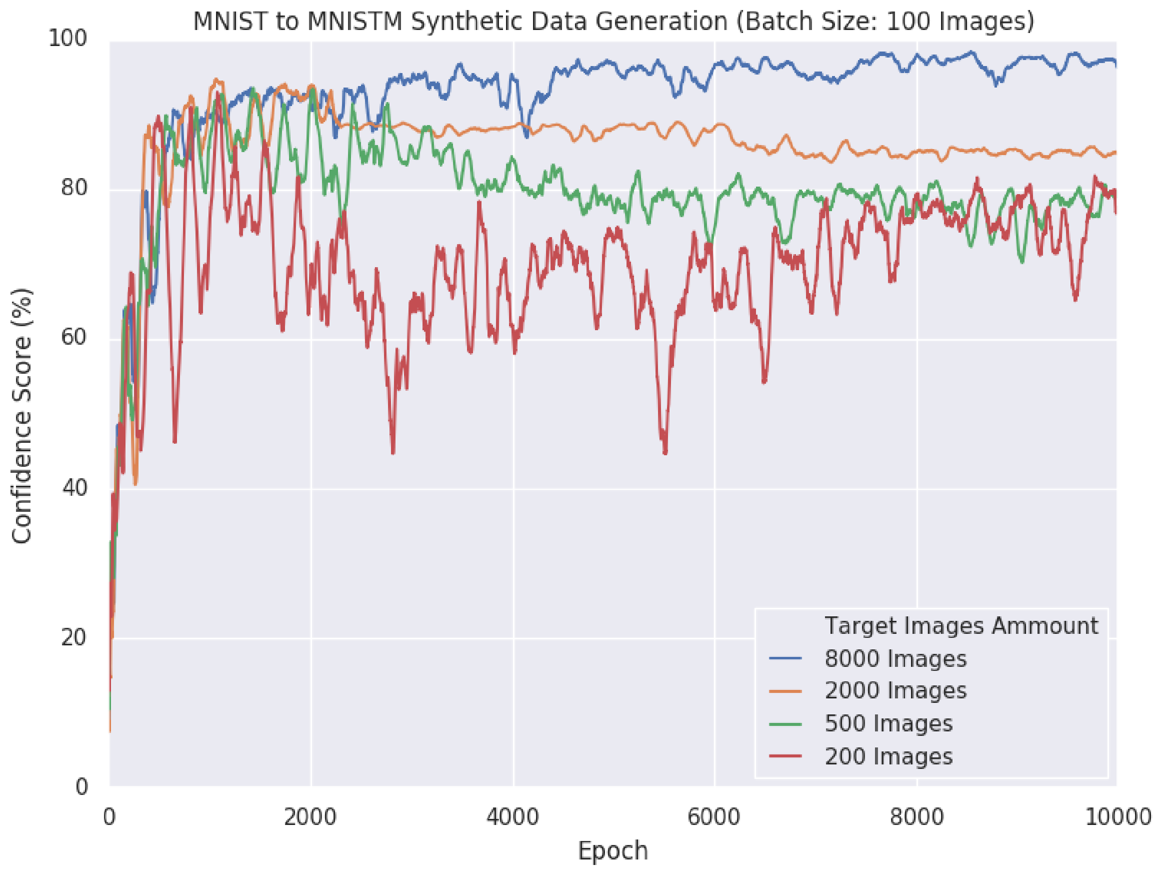


FIGURE 7.2: Target model's confidence score (%) of synthetic data using a batch size of 100 images.

The GAN's qualitative performance can be observed as it trains by taking a sample from the generator.

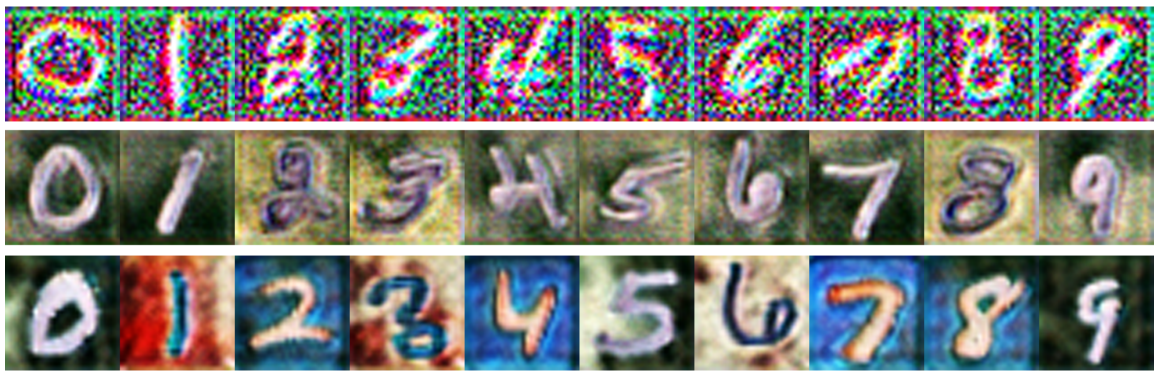


FIGURE 7.3: Synthetic Data produced during training.

Conducting the same experiment 10 separate times produced the below results:

TABLE 7.1: GANs effectiveness at different Target Data Sizes

Target Data Size	Max. Accuracy (%)	Min. Accuracy (%)	Avg. Accuracy (%)	Avg. Batches til Convergence
8000	96.89	95.20	96.72	2,100
2000	94.22	81.13	88.58	2,100
500	91.89	80.01	82.77	2,100
200	90 (Peak)	55 (Trough)	N/A	DNC

When testing with 500 images, 2 tests did not converge and were not added to the above metrics, but the failure to do is noted. In the case of 200 images all tests did not converge.

An important outcome was the ability for the GAN to transform source 1's to the target domain when it was written in the 'serif' form (with a horizontal line at the base). There are only 2 serif ones in the target data set when 2000 images were available, but 203 (out of 8000) available in the MNIST data set.

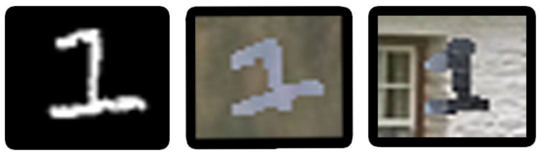


FIGURE 7.4: GANs ability to learn 'serif' ones from 3 target image examples (Left: Source MNIST, Middle: Synthetic, Right: Target MNISTM)

Ensuring GAN is Learning and not Copying

The second half of these results focuses on ensuring that the generator produces data in the target domain, but does not simply replicate said target data. The test measured the mean distance and std. deviation between synthetic images and the nearest target image of same label.

TABLE 7.2: Distance between Synthetic and Target Data

Mean Distance	Standard Deviation
0.49	0.86

A qualitative review was conducted by using the euclidean distance to select the closest synthetic image to a target image from each label.

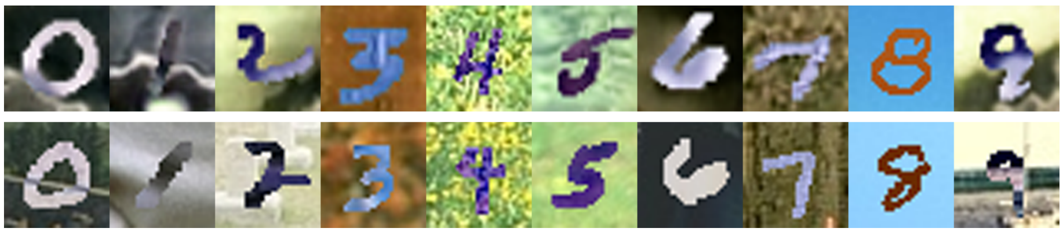


FIGURE 7.5: Comparison of synthetic media against most similar target image

7.1.2 Evaluation

The above results were in part, unaccounted for in the initial hypothesises. The GAN, as expected, could produce a high confidence synthetic data set when provided the maximum amount of target data. Unexpectedly however, when given less data the mean accuracy of the synthetic data did not significantly diminish. Instead the ability to converge was strained. Of significant note, is that until only 20 images of each label was provided, did convergence appear to be significantly troubled. This will mean that even when given severely limited resources, the GAN will not undergo mode collapse and will still produce a synthetic data set that can emulate to some significant degree the target domain.

Evaluating the Improvements

The MNIST to MNISTM domain ratification test is, as elaborated previously, a standard benchmark to evaluate transductive transfer learning approaches. The extensions on top of the strong initial pixel-wise approach have shown themselves to be worthwhile.

The pixel-wise approach in Bousmalis et al. [114] initial paper did not cover the effect of a reduced representation of the target domain. As such, it is unknown how much the improvements made to this pixel-wise approach, affected the ability to converge. However, the use of a curved gradient distribution and the alteration made to the softmax and ReLU layers to reduce the vanishing gradient problem are known to positively contribute to the speed, stability and final height of convergence.

From 3 Serif Ones to 203

The results from the MNIST to MNISTM transfer using 2000 images contained a suprising outcome. Within the 2000 images, was 3 one's that were written using a horizontal line at the bottom. Due to the small amount of these one's in the target data set, its probable that a classifier in the traditional pipeline would fail to learn their association with ones. However, the MNIST data set contains 203 of these. With the ability to transfer these 203 differently shaped ones, using only 3 target images as a reference, the GAN again highlights its amazing ability to create enough synthetic data to handle a user whose unable to provide much data in the target domain.

Ensuring the GAN is Learning

The second tests showed that the generator was not copying the target data, but instead learning the domain wholly. Indicating that if this platform was extended to incorporate a continuous learning module, then new source data would be successfully transferred into the target domain producing unique synthetic data.

These tests still leave questions on how the GAN will perform when the foreground and background begin to blend, which is the case in most real world object recognition problems. This will be covered in the major use case during system implementation. The components of this approach that will assist with crowded images, have been so far underutilised. Techniques for random cropping and channel switching should be explored to further improve on these results.

7.2 *SeqGAN*

7.2.1 Results

Oracle NLL Tests

Testing using the Oracle NLL data set showed a successful implementation of the *SeqGAN* and it's alterations. The modifications made to the initial *SeqGAN* approach were evidently effective in improving the generator during pretraining, however the loss from the generator did not overtly decrease as expected once the model was being adversarialy trained and capped out at a loss of ≈ 11.

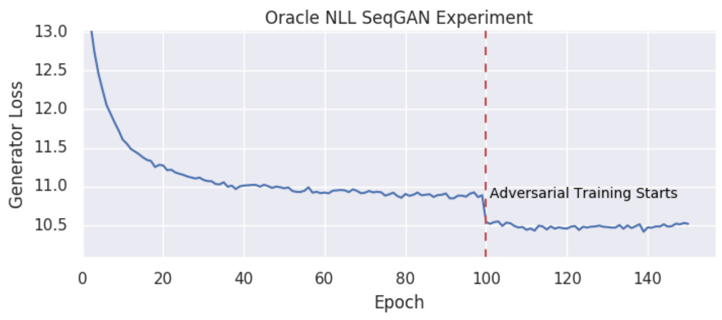


FIGURE 7.6: Oracle NLL error during pretraining and adversarial training.

Obama Sentence Creation

The results for the second tests in the production of tweets on the other hand were troublesome. When provided both a large and small target data set, the learning curve captured by sampling of the discriminator loss was near flat.

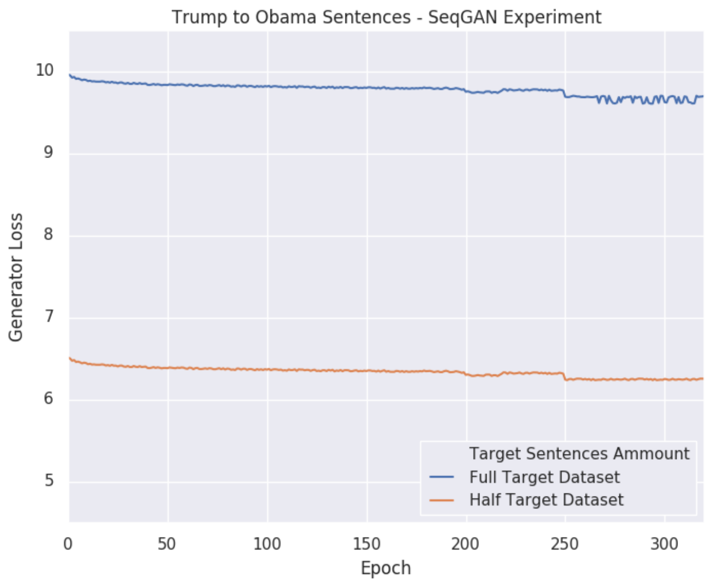


FIGURE 7.7: Loss from discriminator while training.

These losses were contradictory to a qualitative review which saw that the data produced by the *SeqGAN* was well inline with what would be expected of Obama's sentences.

TABLE 7.3: Conversion between Trump to Obama Speeches

Source Data - Trump	Synthetic Data - Obama
you are a liar and you have lied to the press	despite big stories our era serious journalists find themselves often without
you don't get to decide, I get to decide, I make the rules	stuck with choice between cuts our bottom line and what believe
keep me away from china	rather keeping away his attitude

The BLEU score of the synthetic data against target data across widening n-grams was very high:

TABLE 7.4: BLEU score comparison

Size	n-gram	Score
8000	2	75.2
8000	3	72.6
8000	4	61.3
500	2	53.6
500	3	43.1
500	4	36.9

The following results evaluate the frequency of the Obama's top 100 words and common phrases in the target and synthetic data highlighting how close the two data sets were. There was a strong indication of a Gaussian distribution centred at close to zero words difference.

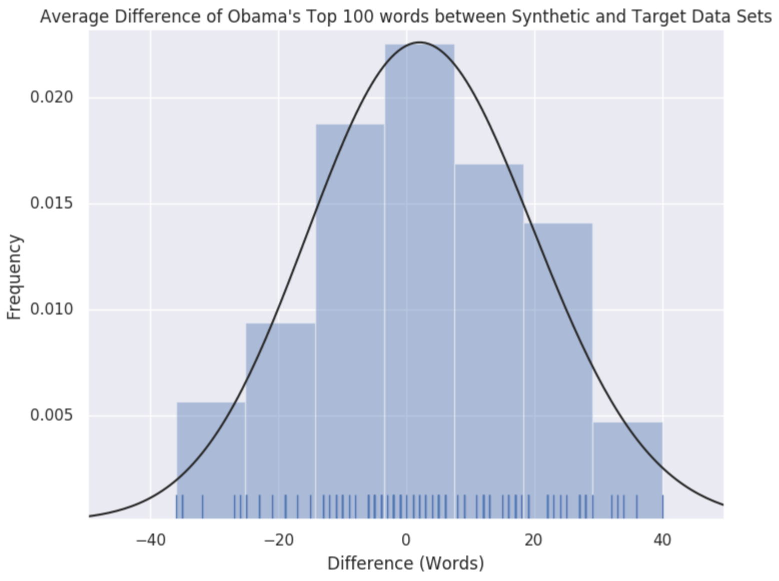


FIGURE 7.8: Monte Carlo approximation of the distribution of 100 unique words in synthetic data against target data set.

7.2.2 Evaluation

From the success of the initial tests using the Oracle NLL data set, used to prove the effectiveness of *SeqGAN* in its original implementation, this paper confirms that *SeqGAN* has been implemented correctly and the additions made to it did provide a modicum of improvement. These improvements were seen in the pretraining, but did not appear to carry over to the adversarial training. The initial hypothesis believed that any improvement would of been seen in both the pretraining and adversarial training.

This failure in the generator could be attributed to over-training of the discriminator. However, this would be observed as a severe drop into a flat line convergence, not a short drop into a slight slope. As the initial paper associated to *SeqGAN* reaches a similar convergence, it is likely that this is the limit for *SeqGAN* on this problem.

These results have however highlighted a significant issue with *SeqGAN*s and any other domain adaptation approach in the NLP space. The *SeqGAN* implemented here, retested and revaluated, did not learn the target domain. Instead it suffered from mode collapse almost immediately and learnt to copy the target data instead of learning the domain. This is observed by the almost identical distribution of the frequency of unique words in the target and synthetic data and the flat line in the generator loss.

Further exploration of this problem yielded a significant problem. GANs are explicitly designed to learn the representation of the target domain they are given. Despite the large amount of data in the full representation of Obama's sentences, it is not as diverse as expected. There are currently 115,000 'true' words in the English dictionary (not obsolete or derivatives) and Obama in a set of 38,000 sentences only used 2150 unique 'true' words. This is also doesn't include his common phrases, speech patterns and preferences.

Due to this very small representative space, it appears that *SeqGAN*s may struggle to learn the translation between domains in the NLP space. There has been little work in this area, and GANs have only been applied in works with little citing. However the main use case will still need to be tested, due to fundamental differences between the nature of support tickets and speeches from one person. Vocabulary and phrasing should be more diverse, in a data set where each sentence comes from a different person.

7.2.3 Conclusion

Vocabulary and repetitive use of the same phrase appeared to significantly reduce the ability of the *SeqGAN* to learn the target domain. However, unlike other GAN approaches this appeared to take the from of it simply copying target phrases, not just a single phrase. This is likely due to the "encoder-decoder" structure of the generator. This module test will mean that during the major use case, particular care will need to be taken on observing how variance in the data set affects the *SeqGAN*s performance.

7.3 *FreezeOut*

7.3.1 Results

Inductive Transfer Learning using MNIST:

The first inductive transfer learning test compares *FreezeOut*'s retraining of a MNIST model against a model trained from scratch for a simple binary task, evaluating 0's in the MNIST data set. In this experiment *FreezeOut* outperformed scratch trained models with a significantly higher initial rate of learning while still retaining the same final precision. The retrained models also showed less fluctation in learning then the scratch trained model.

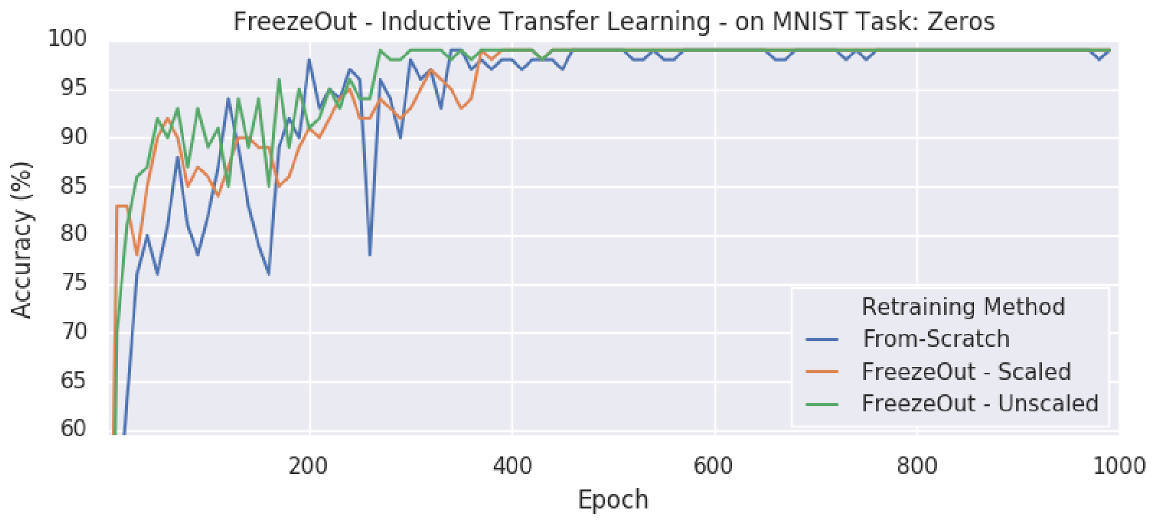


FIGURE 7.9: Using *FreezeOut* for retraining a MNIST domain model for a binary task of identifying 0's in MNIST data set

The second inductive test evaluates a more complex application. Where sets of numbers are grouped together. In this experiment *FreezeOut*'s performance was minimally better than the from scratch trained model. Reaching convergence just before the from scratch model. The results show a higher accuracy from the *FreezeOut* models.

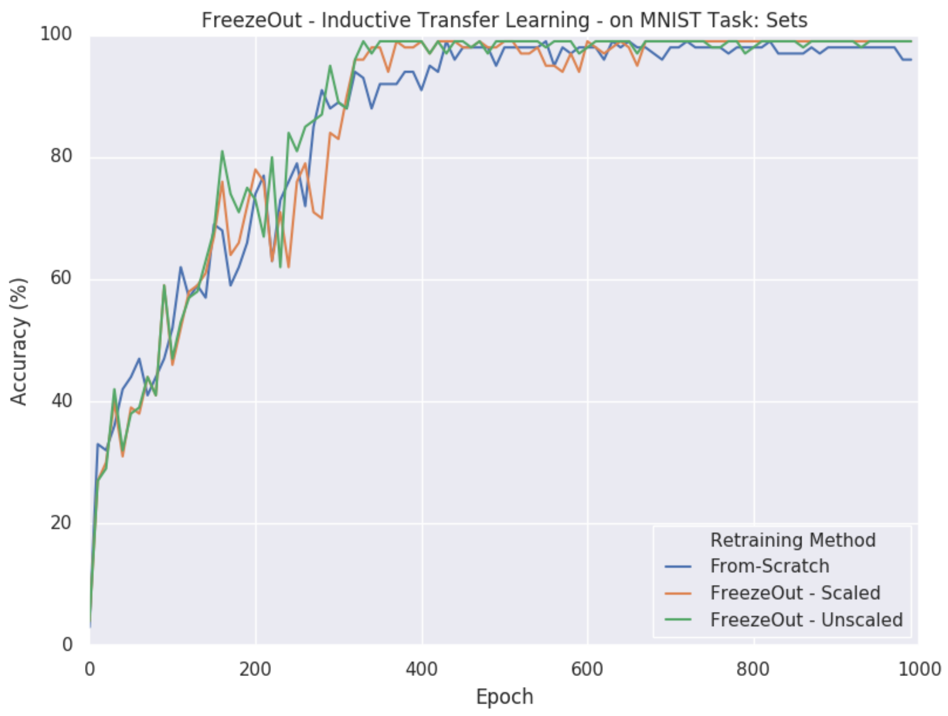
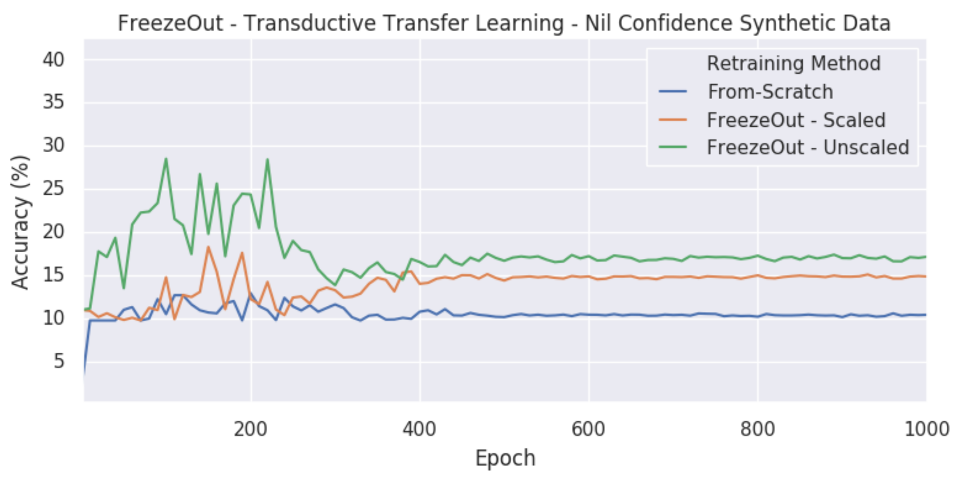


FIGURE 7.10: Experiment evaluating *FreezeOut* inductive transfer learning ability in retraining a MNIST domain model for a multi-class task of evaluating sets of numbers in the MNIST data set.

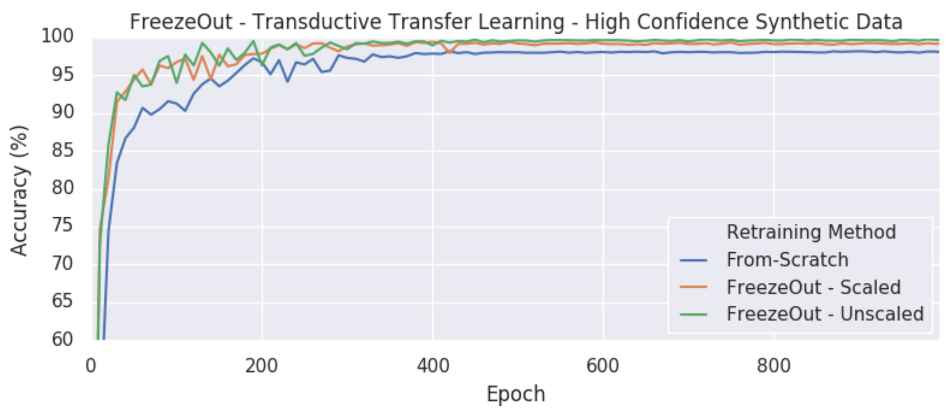
Transductive Transfer Learning in the MNIST-M Domain:

With the MNIST source model secured from the *Source Model Creator* module, the first set of tests produced precision results for the retraining of the MNIST source model using MNIST-M synthetic data. This test was used across 3 'synthetic' data sets: A nil-confidence set that was made from MNIST images, a high confidence set made from synthetic data produced by the domain adaptation module with a confidence of 98% and a 'perfect' set that is composed of purely MNIST-M images. The results captured the same test run using a model trained from scratch as well as MNIST models retrained using *FreezeOut*. Scaling was evaluated in all 3 experiments.

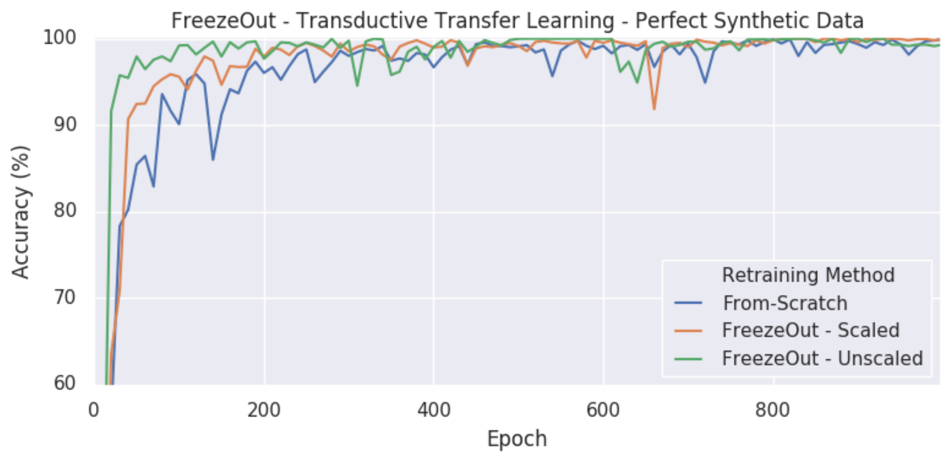
Due to the overall high performance captured in the above tests for both scaled and unscalled *FreezeOut* implementations. The necessity of using *HyperOpt* was reevaluated. The experiment using the high confidence synthetic data was repeated 8 times testing different starting scales.



(A) Nil Confidence Synthetic Data



(B) High Confidence Synthetic Data



(C) Perfect Confidence Synthetic Data

FIGURE 7.11: *FreezeOut* training and validation using different grades of synthetic data

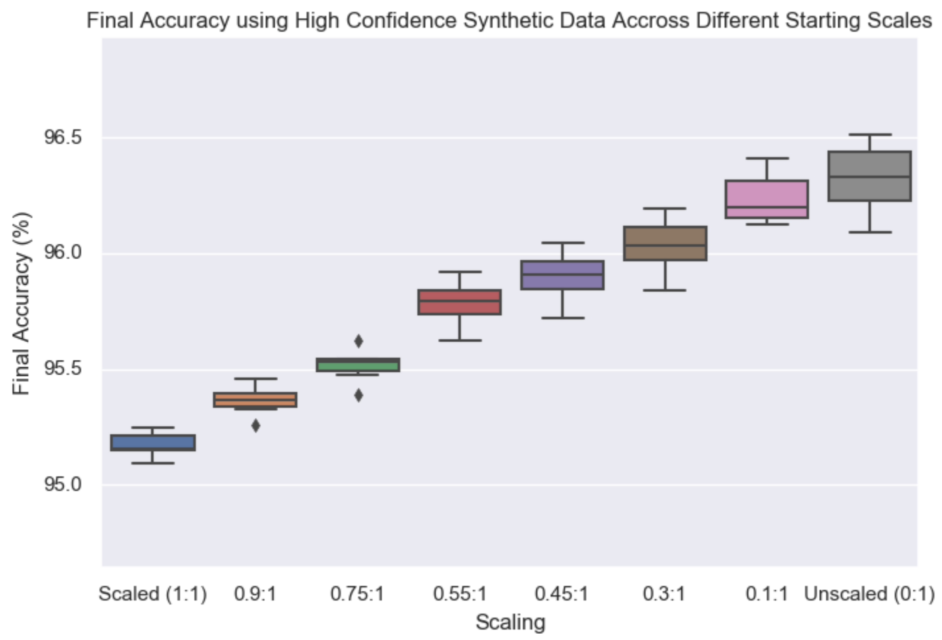


FIGURE 7.12: *FreezeOut* retraining with high confidence synthetic data using different starting scales

To compare accuracy with speed, the above tests also compared the time required to reach 90% accuracy, similar to how response time is evaluated in electrical systems.

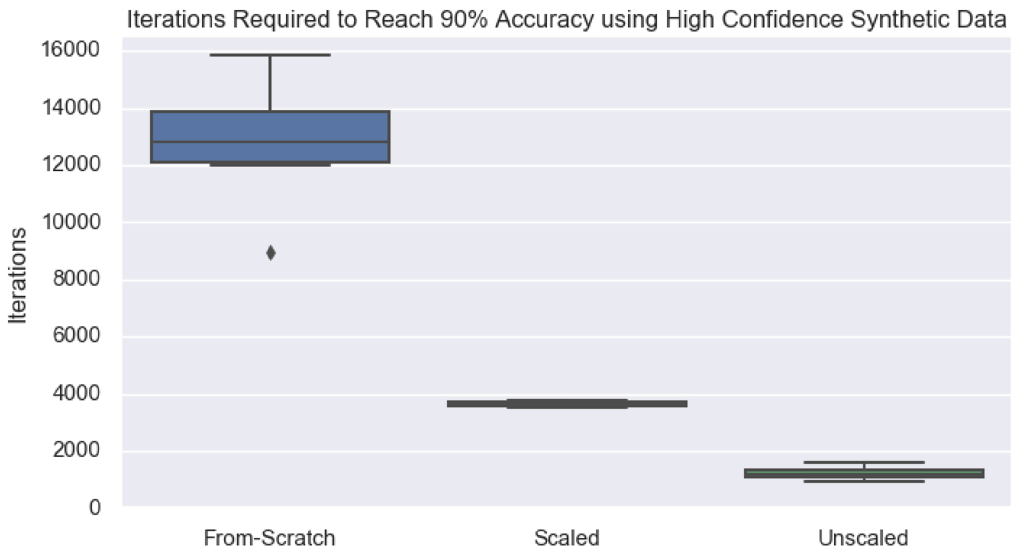


FIGURE 7.13: Iterations required to reach 90% accuracy using high confidence synthetic data

7.3.2 Evaluation

In both the inductive and transductive transfer learning applications *FreezeOut* showed promising results. In both applications, it beat from-scratch training methods in both reaching convergence and it's final accuracy. In the MNIST to MNISTM domain adaptation tests, from-scratch methods achieved a 98.1%, which was the overshadowed by *FreezeOut*'s ability to produce model with 99.4%. Providing validity for the ability of the *FreezeOut* approach in retaining both domain and task relevant features.

Another positive outcome was the non-effect scaling had on the inductive transfer results and a lack of scaling had positive effects on the transductive results. The longer travel time in the training space for all layers was clearly important if new domains needed to be completed, but not so much when task context data was evaluated. Likely weights in the lower parameter space play an important part when domains are drastically altered, this was observed when poor synthetic data was used.

A generic *FreezeOut* algorithm that uses unscaled learning rates appeared to be preferable in the transfer learning application. This removes the need for the implementation of the *HyperOpt* wrapper in the retraining module and vastly reduces the computational overhead.

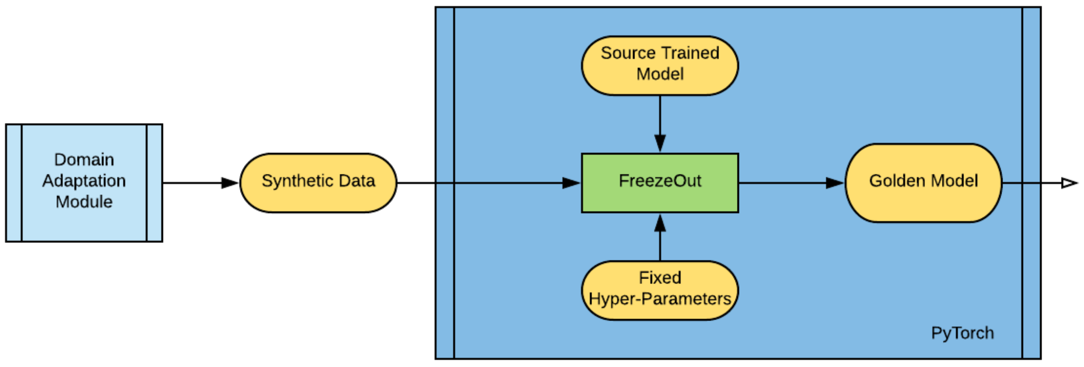


FIGURE 7.14: Retrainer module using FreezeOut

These results confirms the initial hypothesis proposed by this platform, that by transferring knowledge between adjacent domains or from task to task, the platform is benefiting from the transition. Not just in increased overall performance, but in the observable acceleration of the initial learning rate. *FreezeOut* is able to demonstrate a capacity for this transition that does not include a possible negative transfer learning outcome.

7.3.3 Conclusion

In this experiment *FreezeOut* achieved in creating models with accuracies higher than its from-scratch counterparts. This is significant when considering the current record for the MNIST to MNISTM benchmark sits at 98.1%. When using the high-confidence synthetic data to train a model from-scratch, an accuracy of 98.0% was achieved. However, when using the full pipeline and making use of the retraining model a repeatable accuracy of 99.4% was realised. This, along with the high results, proves the ability of the *FreezeOut* approach in retaining both domain and task relevant features.

8

Integration Architecture

As specified at the beginning of this body of work, it was stated that the "ML-as-a-Service" platform built to support and prove the validity of the 'two-step transductive transfer learning' pipeline would be minimalist and not cover all the requirements of a fully deployable product. However an outcome of this thesis would be the framework to which that product could be built atop of. Because of this, at all times the requirements to satisfy that final platform must be satisfied by everything that this body of work has produced. This chapter will cover the integration of all the modules so far and the requirements that will be proved by the next two use cases that of been satisfied in the modules so far.

8.1 "ML-as-a-Service" Platform Architecture

The proposed "ML-as-a-Service" platform has 4 key modules, inclusive of the pipeline's own 2 modules. Within the context of the platform these modules are:

1. **Source Model Creator:** Responsible for creating the *DenseNet* models for a given task and data set using the source data held within some form of data base. This module incorporates an *Auto-Tune* pipeline that removes the need for hyper-parameter tuning by the user.
2. **Pre-Processor:** Split into 2 components, that deal with the pre-processing of image and lexiconic data type files. This module will process both the user's inputted target data and the stored source data. It will conform and these normalise the data sets so that the same model can use both data sets without transformation.
3. **Domain Adaptation:** Responsible for transforming data from the source domain into the target domain, producing a synthetic data set. This synthetic data set will sit within the target domain, but be of equal size to the source data set. Two methods are used, split due to the differences of handling continuous and discrete data sources.

4. **Retrainer:** Takes models previously trained using source data and retrains them using the synthetic data set. This process allows the model to retain useful knowledge of the previous domain/task giving it additional accuracy/robustness when deployed in the target domain.

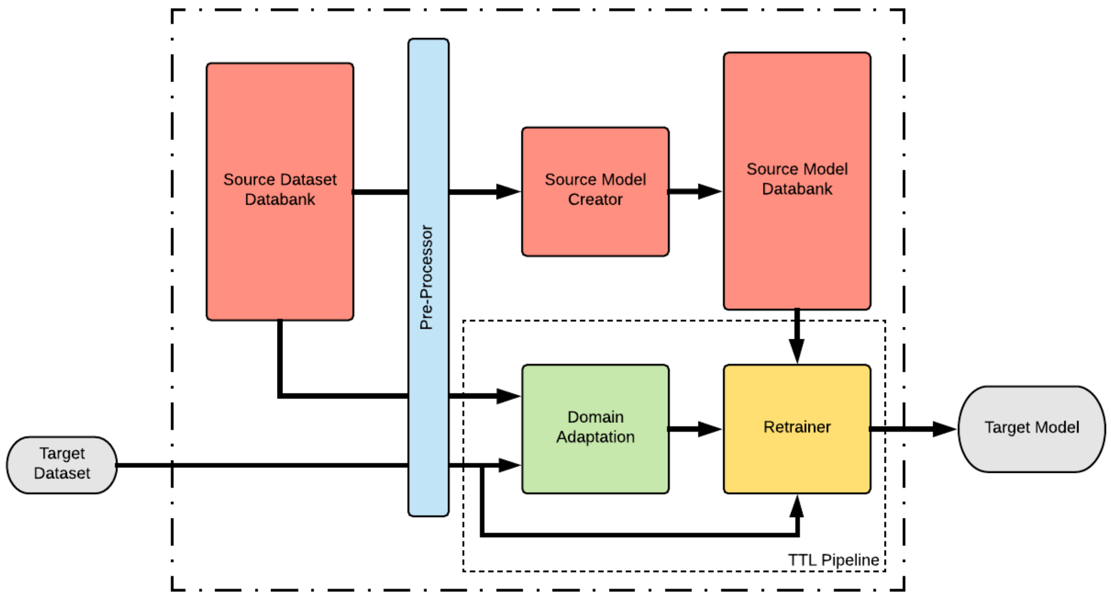


FIGURE 8.1: Platform Layout

8.2 Platform Requirements

The requirements that have been placed on each module so far are in order to satisfy the following requirements. These requirements are imposed on this platform due to the expectations and needs of a non-expert user of an "as-a-Service" platform.

8.2.1 Optimal

The classifier produced by the platform must be optimal in the target domain after having fully converged. This requirement is fulfilled by each module in the platform completing its role to optimal quality. The choice of *DenseNets*, *Doc2Vec*, the current best practises for GANs and the novel implementation of the *FreezeOut* approach, all modules have been built as the most up-to-date solution for their specific problem area.

8.2.2 Data Agnostic

The platform must be able to handle all data types. While only images and text are the focus of this thesis, the choice of model (*DenseNet*) and the framework of the pipeline must allow for other multi-modal data types such as voice, music, spreadsheets/csvs and biological data. By focusing on having the pipelines able to handle continuous and discrete data sources this platform remains data agnostic.

8.2.3 Low Data Quality/Size

The user must be allowed to provide as poor of a representation the target domain as possible. The size of the data set the user need provide, must be as little as required, so that the need for a large data sourcing projects are unnecessary. This requirement is the focus of the domain adaptation layer of the pipeline which seek to significantly reduces this bar to entry for inexperts by allowing them to make use of similar source data sets.

8.2.4 A 'Complete' Black Box

To satisfy the requirements for an "as-a-Service" platform used by a non-expert user, the whole platform must be deployable in a completely closed state. There must only be two inputs, Target Data and task. There must only be two outputs, the optimal model and an associated report for the model. The final model should require no changes to be deployed once made available to the user. The inclusion of the two non-pipeline modules and the use of the *auto-tune* pipeline ensured this requirement was met by removing all elements that required tuning from the core of the pipeline and allowing for them to be made automated when the platform was made into a full produce.

8.2.5 Scalable

This platform will seek to store every large data set and model it encounters or produces, in order to satisfy the requirement of an optimum model. This platform should make efficient use of hardware and software to remove any space and time overhead possible. The selection of the correct machine learning framework and other software packages plays a large role in meeting this requirement. This is met so far by the use of modular design principles that will allow for their replacement and update without affecting the performance of other modules.

9

Use Case 1: Image Recognition - Experimental Setup and Results

9.1 Problem Statement

When equipment is changed or alterations are made to the environment between the time period that the training data was captured and when the model was applied, it can be expected that accuracy of those ML models will significantly decrease. This problem and many similar to it can be summarised as a situation where the user is unable to provide a sufficient representation of the target domain, but has access to a large amount of data in a similar, but ultimately different domain. This use case will evaluate an extreme occurrence of this problem.

The pipeline will be given a relatively small *ImageNet* data set which has been blurred and distorted using a sepia filter. The source data set and model will come from the larger label-matched *CIFAR10* data set. The differences that exist naturally and have been introduced artificially will mimic the aforementioned problem with equipment and environmental differences.
This use case will capture:

- The minimum required amount of target data that is needed to provide a ′good′ approximation of the target domain for this application.
- The magnitude of dissonance between the source and target domain that can exist for this application.
- The effectiveness of the ′two-step transductive transfer learning′ pipeline for a complicated image recognition use case.

9.2 Hypothesis

There are two hypothesis that will be tested in this use case.

1. That the two-step transductive transfer learning pipeline produces models that are more accurate, then a purely domain adaptation approach when provided the same target data set. ($\mathcal{A}_{opt} \approx \mathcal{A}_{\mathcal{M}_{Synth}} > \mathcal{A}_{\mathcal{M}_T}$)

2. As the dissonance between the source and target data set is increased either by reducing the size of the target set or by increasing the impact of the occlusions i.e. blurring, the accuracy of the representation of the target domain by the synthetic data set will decrease. However, there exists an asymptote which the accuracy will converge to, which correlates to the minimum amount of data needed from the target domain. Such that as $|\mathcal{D}_S - \mathcal{D}_T| \longrightarrow \infty$ then $\mathcal{A}_{model} \longrightarrow \mathcal{A}_{min} \neq 0$.

9.3 Method

9.3.1 Creating Sepia and Blurred Data Sets

First the data sets must be created using the *ImageNet* library and the *CIFAR10* data set. For each data set, note the total variance between images.

1. Create a subset of the *ImageNet* library using only the labels that match the *CIFAR10* data set.
2. Using the *preprocessing* module ensure the conformity between the *ImageNet* subset and the *CIFAR10* data sets.
3. Apply a Sepia overlay to *ImageNet* data set.
4. Create 3 data sets of 100%,50%,10% images per class.
5. For each data set create 3 blurred data sets of 0%, 5% and 10% blurring.
6. For all data sets record their variance as the mean squared error (valid due to normalisation that occurs in pre-processing):

$$MSE = \frac{1}{n}\sum_{i=1}^{N}\sum_{j=1}^{n}\frac{(R_j - \mu_{R_i}) + (G_j - \mu_{G_i}) + (B_j - \mu_{B_i})}{3} \tag{9.1}$$

After this, there will be available 9 synthetic data sets of varying size, 1 conformed source data set of 6000 images per label and 1 target data set of between 1000-1500 of each label.

9.3.2 Evaluating Traditional Machine Learning Pipeline

A reference point for the results of the two-step transductive TL pipeline must be established. This is done by capturing the accuracy of the traditional machine learning pipeline.

1. Train a *DenseNet* using the full source data set.
2. Classify the target data set.
3. Train a *DenseNet* using the full target data set.
4. Classify the target data set.

9.3.3 Evaluating the new Two-Step TTL Pipeline

Then in order to capture the effectiveness of the two-step transductive transfer learning pipeline for this image recognition use case, the follow tests are carried out.

1. Push the 9 modified *ImageNet* data sets through the domain adaptation module to create 9 synthetic data sets.
2. Capture the confidence score of these synthetic representations using the model trained on the full target data set to classify them.
3. Synthetic data will be deemed to of undergone mode collapse if the target model does not reach a consistent accuracy or reaches a consistent, but extremely low confidence score. In these cases the output should be evaluated by eye.
4. Retrain the *CIFAR10* source model in the retrainer module on each synthetic data set.
5. For all tests compare against:
    - A model trained from scratch on the synthetic data set.
    - A model trained from scratch on the target data set.
    - A model retrained on the target data set.
    - The accuracy of the source trained model.

9.4 Results

9.4.1 Creating the Synthetic Data Sets

After mode collapse was observed to occur frequently in these tests, for each data set, the test was repeated 8 fold. This was done to ensure that if mode collapse occurred, it was guaranteed. All data sets either did or didn't experience mode collapse, repetition did not produce different outcomes.

TABLE 9.1: Pixel-Wise GAN's performance in Use Case 1

Size (images)	Blur (%)	MSE	Synthetic Confidence (Acc. %)
15096	0	0.272	78.1
7548	0	0.271	46.8
1510	0	0.215	~13 (MC)
15096	5	0.241	61.2
7548	5	0.178	~14 (MC)
1510	5	0.138	~12 (MC)
15096	10	0.115	~12 (MC)
7548	10	0.101	~8 (MC)
1510	10	0.092	~11 (MC)

9.4.2 Finding the Point of Mode Collapse

A second round of tests were conducted then to find the point where the GAN experienced mode collapse. It was found when the variance of the target data set has been reduced to approx. 0.213 (with a variance of 0.0048 across 8 tests). This represented 22% of the total images and 3% blurring or 28% images and 4% blurring. Blurring significantly decreased variance in the data set.

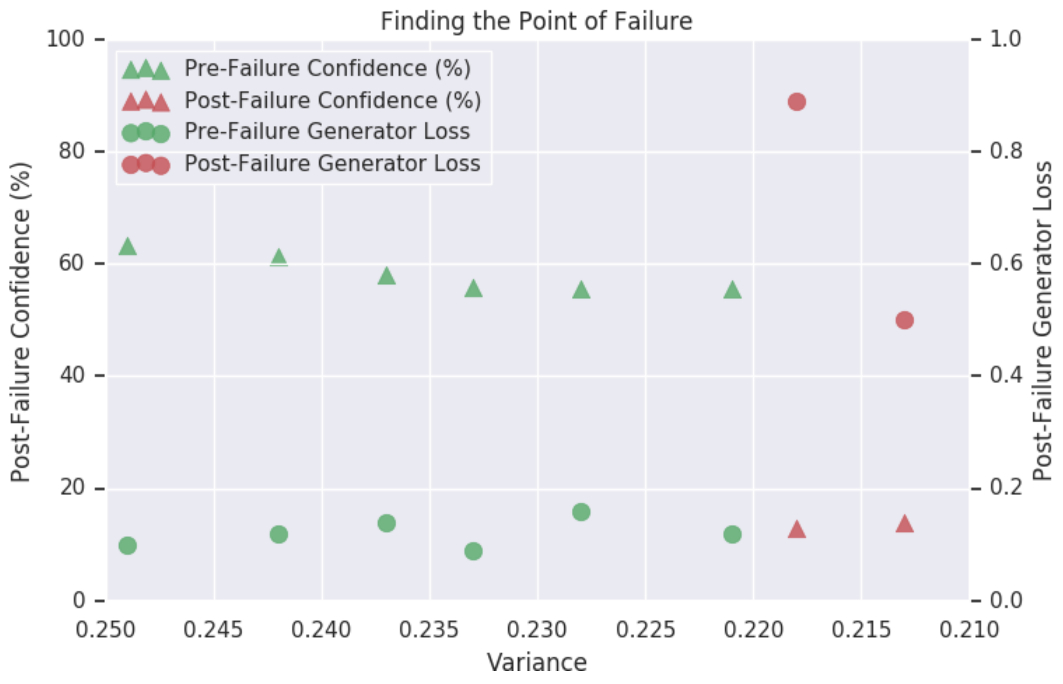


FIGURE 9.1: Confidence Tests around Point of Failure

A comparison of the synthetic and target data sets before and after the Point of Failure was done. However, unlike

here it was hard to notice a difference in the quantitative comparison due to the similarities that the sepia overlay introduced into the data set.

The quantitative review selected the closest synthetic image from each label to a target image using the MSE and euclidean distance between them. This review provided a much better insight into the GANs performance. Despite the similarities that the sepia overlay imposed, no direct copy of the target data was observed.

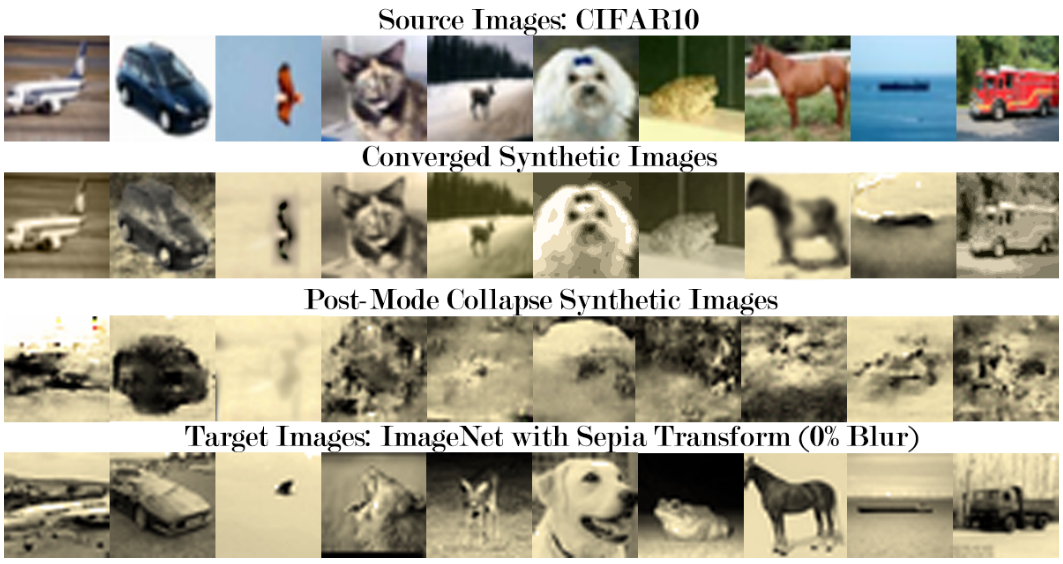

Source Images: CIFAR10
Converged Synthetic Images
Post-Mode Collapse Synthetic Images
Target Images: ImageNet with Sepia Transform (0% Blur)

FIGURE 9.2: Qualitative comparison of synthetic data before and after mode collapse

9.4.3 Evaluating Entire Pipeline

Based off the previous results, the 10% blurred data sets was not be included in these tests, in its place the data found just before the point of mode collapse that was found in the previous test was be supplemented.

The Point of Failure results are of particular interest, as they demonstrate the best application of this use case, where the user provides as little and as poorer a representation of the target domain as possible.

TABLE 9.2: Performance of Two-Step Pipeline in Use Case 1

	F1 Score (%)	**Recall (%)**	**Precision (%)**
100% **Images,** 0% **Blur**			
Synthetic + Retrained	94.43	95.83	93.03
Synthetic + From-Scratch	94.01	95.51	92.61
Target Data + From-Scratch	93.78	94.28	93.18
50% **Images,** 0% **Blur**			
Synthetic + Retrained	93.4	94.5	92.4
Synthetic + From-Scratch	91.95	92.65	91.65
Target Data + From-Scratch	78.33	85.93	71.93
10% **Images,** 0% **Blur**			
Synthetic + Retrained	21.4	13.0	60.0
Synthetic + From-Scratch	14.3	10.0	25.0
Target Data + From-Scratch	61.55	62.55	60.65
22% **Images,** 3% **Blur (PoF)**			
Synthetic + Retrained	**92.43**	96.83	84.83
Synthetic + From-Scratch	89.01	90.96	87.16
Target Data + From-Scratch	64.3	58.4	73.3
100% **Images,** 5% **Blur**			
Synthetic + Retrained	88.2	94.4	82.8
Synthetic + From-Scratch	87.5	91.8	83.6
Target Data + From-Scratch	86.8	90.4	83.6
50% **Images,** 5% **Blur**			
Synthetic + Retrained	15.4	10.2	28.20
Synthetic + From-Scratch	11.1	7.9	18.2
Target Data + From-Scratch	40.5	34.8	48.5
10% **Images,** 5% **Blur**			
Synthetic + Retrained	19.0	11.1	26.7
Synthetic + From-Scratch	14.3	9.09	23.30
Target Data + From-Scratch	18.2	11.1	34.1

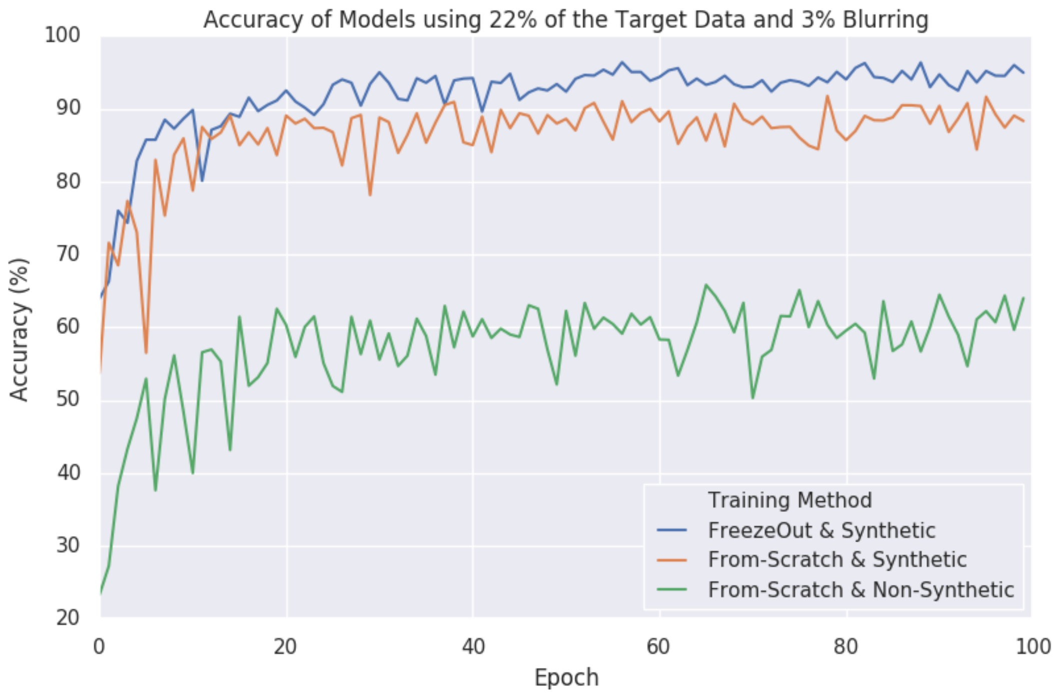


FIGURE 9.3: Comparison of Full and Half TTL pipeline and Traditional Pipeline using Point of Failure

10

Use Case 1: Image Recognition - Discussion

The results captured from this use case and those contained in the module testing, provide the validation for this two-step transductive transfer learning pipeline. Proving its effectiveness in navigating the most common problems experienced by non-expert users when lacking enough data for the target domain.

10.1 The Failure of Blurred Sepia Images

Giving the target images a sepia overlay did not appear to initially affect the accuracy of the GAN or the final classifier, beyond a small reduction in accuracy and synthetic confidence. However due to a significant reduction of information that both the sepia and blurring imparted, at a certain point (22% images and 3% blurring) there ceases to be enough of a representation of the target domain to build a model around and the GAN experiences mode collapse. Failing to converge, the resulting synthetic data is extremely inaccurate. Why this occurred in these experiments and not the MNIST to MNISTM benchmark, is likely a combination of the aggressive variance reduction that sepia tones and blurring apply to the target data and also the complexity of the images in both data sets.

10.2 The 'Yield Point of Failure due to the Lack of Representation'

The predicted asymptote that was hypothesised to exist based on the results from the module testing, did not exist here in a form of what was initially suspected. Instead of an asymptote, a curve similar to a stress-strain curve was seen instead.

Similar to a stress-strain curve, a 'yield point' existed instead.

What was seen, was that a linear decrease in the domain representation provided by the target images, produces an exponentially decaying accuracy which converges to a false asymptote. It is described as false, because at a certain point where the target domain is no longer sufficiently represented by the target data set, the generator experiences mode collapse and a synthetic data set can no longer be effectively produced.

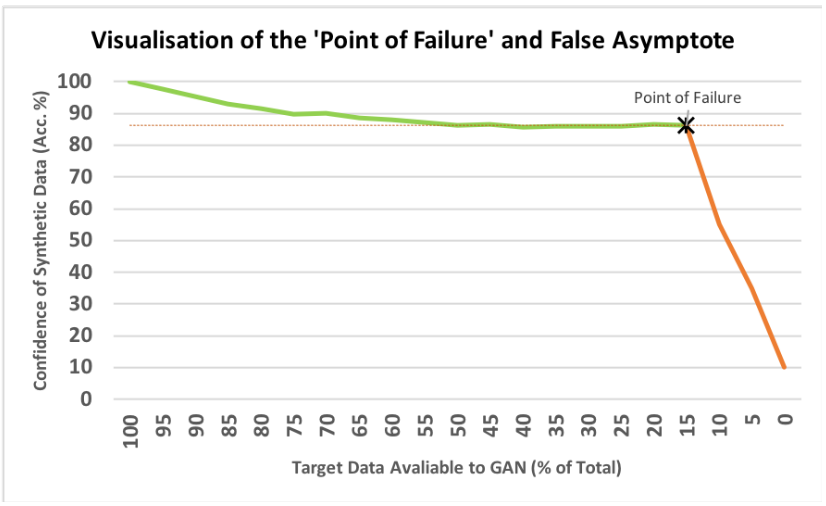


FIGURE 10.1: Visualisation of the Point of Failure and False Asymptote

10.2.1 Formalising the 'Point of Failure'

In the extensive cover of domain adaptation literature in the context of using GANs, beside a desire to avoid mode collapse, this 'Yield Point' has not been referred to before. As such this body of work will label it as the 'Yield Point of Failure due to the Lack of Representation' or simply the 'Point of Failure'. The 'Point of Failure' ($\mathcal{PF}$) will be defined as the point where mode collapse in the generator of a GAN is guaranteed to occur.

The experiments conducted so far have shown that a correlation exists between the variance of the target data set and the final confidence of the synthetic data set. As variance decreased, so did accuracy. However, at a point of limited variance, the 'Point of Failure' $\mathcal{PF}$ is found and the generator is guaranteed to experience collapse. This can be expressed as:

$$\begin{aligned} \mathcal{D}_{Synth} &\longrightarrow \mathcal{D}_T \text{ When } \mathrm{Cov}_T \longrightarrow \infty \\ \mathcal{D}_{Synth} &\neq \mathcal{D}_T \text{ When } \mathrm{Cov}_T \leq \mathcal{PF} \end{aligned} \tag{10.1}$$

However, this is only a correlation at this point. Variance is inherently linked to other experimental factors including the quality and quantity of the images. Locating the metric associated to $\mathcal{PF}$ will require extensive testing across data sets, to either isolate variance or find the root cause of the depreciating quality of the GAN.

10.3 The Use of a Pretrained Discriminator to Evaluate the Synthetic Data

In both the module and use case tests, a 'confidence score' was assigned to the synthetic data sets by a discriminator trained using the full target data set. To prove the validity of this approach an assumption was relied upon. That if all synthetic data sets were measured by the same approach, then whether or not the approach is valid, the final ranking is valid even if the scores are not by themselves. The use of the pretrained discriminator was only then retrospectively proved valid, once the synthetic data was used to train models, which had accuracies of comparable ranking.

10.4 Expanding the Source Task Beyond the Target's Label Space

What was not captured in this experiment was the effect that training the source model on the whole *ImageNet* data set would have (such that $\mathcal{L}_T \in \mathcal{L}_S$ but $\mathcal{L}_T \neq \mathcal{L}_S$). Those results were outside the scope of a transductive transfer learning pipeline. However, by inferring from the module tests that were conducted in Chapters 6 and 7, the retrainer showed its ability to capture both domain and task relevant data. From those results and the results of this use case, it is possible to state that by expanding the initial knowledge of the source model into other domains and tasks not relevant to the target domain and task, then a more robust representation of the domain surrounding the target domain can be generated. Thus producing models with higher accuracy deployed within a more dynamic environment.

10.5 The Significance of these Results

Benchmarks are continually pushed higher with each body of work produced by the research community, indicating that the use of GANs for domain adaptation is still a maturing field of study. In this body of work, the improvements made to the *pixel-wise* GAN and the implementation of the novel retraining layer enables the creation of models that perform better than both the traditional pipeline and other comparable transductive transfer learning approaches.

The use of the *FreezeOut* algorithm as a retraining method along with a successful domain adaptation layer, created a pipeline with superior results than any one of those approaches by itself. The pipeline's ability to improve on Bousmalis's current world standing in the MNIST to MNISTM domain adaptation test benchmarks its ability. While this use case's complex scenario and the discovery of the 'Point of Failure' highlight its limitations.

10.6 Summary

This use case delivered three major outcomes in regards to the pipeline's performance in the image recognition space. The first was proof that as long as the GAN does not experience mode collapse the synthetic media it produces, to the size and quantity that it does, can be used to train models that far surpass the ability of the traditional pipeline. The second outcome, the retrainer layer showed significant promise, continually surpassing the from-scratch training. The final outcome concerns the new found 'Point of Failure', which whether caused by low variance in the target data set or other wise, is an important metric that needs to be investigated. These outcomes while limited to the image recognition space are still indicative of the expectations that a successful application this two-step transductive transfer learning pipeline can deliver.

11

Use Case 2: Sentiment Analysis - Experimental Setup and Results

11.1 Problem Statement

This use case will use a target data set, composed of IT service desk ticket headers ($\mathcal{S}_T$) from within a corporate environment ($\mathcal{D}_T$). Each ticket will have a label ($y_i \in \mathcal{Y}_T$) which corresponds to the help desk the ticket needs to be forwarded onto. These ticket data sets are historically very noisy and when these systems are setup, there is little training data available. To remedy this, a cleaned and obfuscated ticket data sets ($\mathcal{S}_S$) from another company is typically used to pre-train the model. The models are then remade once more target data becomes available. Since the data sets used are not within the target domain and have been even further removed due to the obfuscation process, they train poor starting models that take a while to be retrained using traditional means. This creates unstable product launches that lead to significantly reduced client confidence in the platform.

Through a domain adaptation approach, these obfuscated data sets can first be transformed into the target domain. Then by following up with a retraining of models previously effective in other companies, models trained within the target domain can then be deployed.

11.2 Hypothesis

Based on the results from module testing of the *SeqGAN* and the outcome of use case 1, another 'Point of Failure' is expected to be found. However, within the NLP space, it is expected that the representation is seen in not only the size of the vocabulary, but also the variance of words and phrases.

$$\begin{aligned} \mathcal{D}_{Synth} &\longrightarrow \mathcal{D}_T \text{ When } \mathrm{Cov}_T \longrightarrow \infty \\ \mathcal{D}_{Synth} &\neq \mathcal{D}_T \text{ When } \mathrm{Cov}_T \leq \mathcal{PF} \end{aligned} \tag{11.1}$$

$\mathcal{PF}$ is defined as the point where at the target's data variance is lacking enough to cause mode collapse within the *SeqGAN*'s generator.

To meet the requirements of the "ML-as-a-Service" platform for a final optimal model, the synthetic domain must include the target domain at the minimum when the target domain has only been represented using a minimal data set. Because of this, the $\mathcal{PF}$ cannot be found to exist except under the most extreme conditions. Before that point is found the final model trained with successful synthetic representations must be optimal and be a better solution then any model trained using purely the target or source data set ($\mathcal{A}_{opt} \sim \mathcal{A}_{\mathcal{M}_{Synth}} > \mathcal{A}_{\mathcal{M}_T} > \mathcal{A}_{\mathcal{M}_S}$).

11.3 Method

11.3.1 Building data sets

The data sets used are from two Accenture clients. One larger (47000 total), cleaned and obfuscated data set that is referred to as the *BS* data set and a second smaller (2800 total), data set that will be used as the target data set. Both data sets only contain 2 labels (Incident and Service Request).

As each data set is created throughout these experiments the total variance is taken. When creating a data set the following method is observed.

1. Vectorise data set using *Pre-Processing* module with *Doc2Vec* approach.
2. Apply the following variance formula.

$$\sigma(x,y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \tag{11.2}$$

This will attach a variance to each data set that can then be used when altering the target data set to ensure that the when the 'Point of Failure' $\mathcal{PF}$ is located it is directly related to the variance of the data set and not just its size.

11.3.2 Locating Point of Failure

Take a binary search approach to locate the 'Point of Failure' $\mathcal{PF}$. Looping while the $\mathcal{PF}$ has not been found, starting from a full target data set.

1. Using the new target data set, transform the larger source data set using the *SeqGAN* within the *Domain Adaptation* module.
2. If the GAN is able to converge, halve the variance of the target data set. On each loop reach the desired variance by starting from the full target data set and remove random sentences.
3. If the GAN is unable to converge, add back half the removed variance from the data set (i.e. If 25% of the total variance was available for this iteration then add back 17.5%).
4. If the GAN is unable to converge after using all available data, supplement with additional data from other Accenture ticket data from the same company.

11.3.3 Accuracy of the Pipeline for the NLP Use Case

Once the $\mathcal{PF}$ is found, produce a synthetic representations of the target domain using enough data just before the $\mathcal{PF}$, a third in between the total amount of available data and $\mathcal{PF}$, two thirds and all the available data. Such that 4 data sets are produced. These 4 data sets are used to retrain a source trained model produced by the *Source Model Creator* module. Evaluate the accuracy of these models against the models created from-scratch trained models using the complete target data set and source data set.

11.3.4 Comparing against Jittering

Since the module testing highlighted an issue whereby the *SeqGAN* would simply copy the target domain and not learn it, the following test compares the previous results against a target data set inflated by jittering.

1. Take two copies of each ticket in the target data set and insert two random words that appears in the top 25% most frequent words for this data set into one those copies and only one random word into the other. Creating two new data sets, one with double the target data size using only the one word jittering and the other triple the size using the one and two word jittering.
2. Pass the *Retrainer* module these data sets.

11.4 Results

11.4.1 The 'Point of Failure'

The 'Point of Failure' occurred when 100% of the original target sentences was available. Figure 11.1 shows that the pretraining and adversarial training using the initial 50% variance data set and a subsequent 100% test had effective pretraining phases, but reached mode collapse quickly into adversarial training. By using similar data sets from the same company a larger data set composed of approximately 1.8 times the amount of data equal to a variance increase of 150% adversarial training was able to occur.

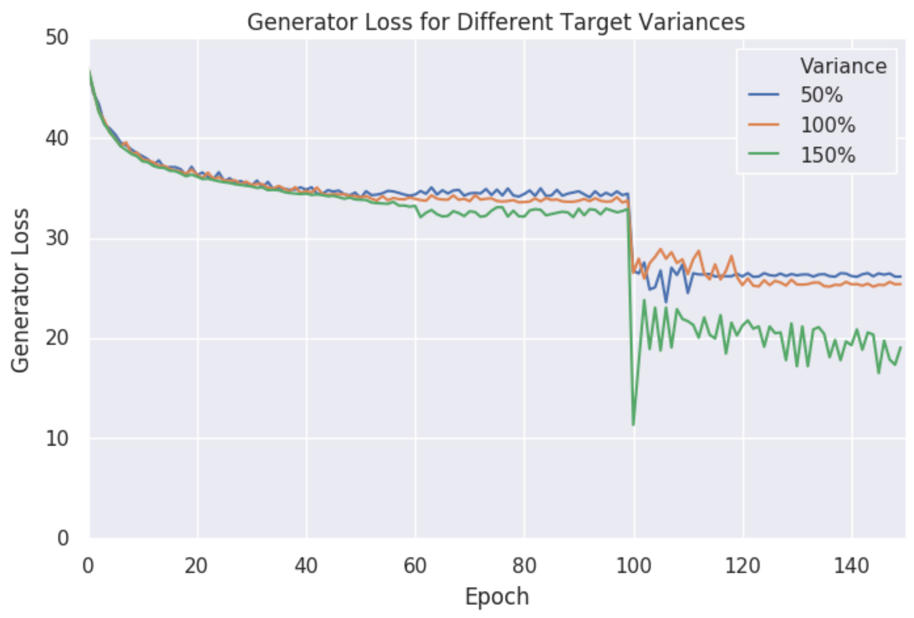


FIGURE 11.1: Snippet of Generator Loss during Pre-training (0-100 epochs) and Adversarial training (101+ epochs).

5 tests put the 'Point of Failure' to occur between $140\% - 144\%$ of the original target data set's variance. However, the sentences that were produced by the generator during failure were all unique between each other. Indicating that the generator was experiencing mode collapse while receiving different source and target sentences.

As was established in the module test, from inspection the synthetic data sets, such as the data set that was captured just after the 'Point of Failure' was reached, does not appear to be incorrect.

TABLE 11.1: Converged *SeqGAN* transformation

Type	Source	Synthetic
Access	*role change in finance team*	*badge access removed new user*
Access	*SAP Access Request*	*check user accessibility*
Access	*new starter*	*port opening*
Incident	*code spelling mistake oracle*	*execute pending items PROD*
Incident	*service unavailable*	*menu Amazon*
Incident	*Lost my access to outlook*	*email failed sent outbox notification*

However by taking a quantitative review, its seen that again upon mode collapse, the generator has learnt to just copy the target data regardless of input. The identical nature of the post-mode collapse synthetic data sets are to the target data is captured in a BLEU style, n-gram comparison of the data sets.

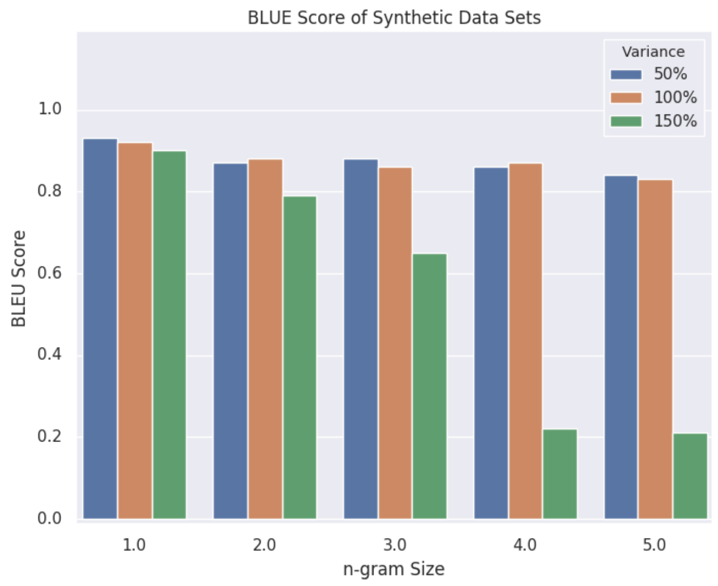


FIGURE 11.2: BLEU scores of the synthetic data sets against the target data set

11.4.2 Full Pipeline Test

Based on the previous test, 3 synthetic data sets are passed on to the next stage of the pipeline, each data set coming from target data with 50%, 100% and 150% variance. Noting that only the 150% variance data set of the 3, converged in the previous test. All 3 data sets are trained from-scratch and the *Retrainer* module. Alongside the synthetic data, 2 other data sets made from jittering with 2 and 1 extra noise words also tested.

TABLE 11.2: NLP Full Two-Step Pipeline Test

	F1 Score (%)	Recall (%)	Precision (%)	Support
Synthetic (50% variance)				
Access	72.0	60.0	90.0	6899
Incident	85.1	93.8	77.8	16933
Synthetic (100% variance)				
Access	73.5	67.8	80.3	6899
Incident	86.8	90.6	83.2	16933
Synthetic (150% variance)				
Access	77.4	63.9	98.30	6899
Incident	88.2	98.4	79.90	16933
Target Data Baseline				
Access	66.2	53.0	88.0	500
Incident	69.2	60.0	81.8	882
Jittering (1 Word)				
Access	68.9	81.9	59.5	1000
Incident	79.8	88.2	72.8	1764
Jittering (1 + 2 Word)				
Access	72.5	59.5	92.6	1500
Incident	84.5	78.3	91.8	2646

The same tests were conducted again without the use of *FreezeOut*. Note that the below test results were averaged from 3 runs.

TABLE 11.3: NLP Using Only Domain Adapation Layer

	F1 Score (%)	**Recall (%)**	**Precision (%)**	**Support**
Synthetic (50% variance)				
Access	70.8	66.0	76.4	6899
Incident	79.3	74.8	84.40	16933
Synthetic (100% variance)				
Access	71.3	56.4	96.9	6899
Incident	82.3	84.1	78.9	16933
Synthetic (150% variance)				
Access	75.8	96.1	61.4	6899
Incident	85.6	83.5	87.80	16933
Target Data Baseline				
Access	57.1	92.6	41.30	500
Incident	62.2	96.5	45.80	882
Jittering (1 Word)				
Access	60.4	50.3	75.5	1000
Incident	78.1	72.3	84.8	1764
Jittering (1 + 2 Word)				
Access	66.7	56.5	81.3	1500
Incident	79.9	86.1	74.50	2646

12

Use Case 2: Sentiment Analysis - Discussion

The outcomes from this experiment were not a positive indication for this pipelines approach to domain adaptation within the NLP space. The amount and quality of target data required by the *SeqGAN* in order to avoid mode collapse was significantly higher than the amount required by the *DenseNet* to classify a smaller target data set. Thus failing a core requirement for this pipeline.

Furthermore tests done with jittering highlighted that the data produced by the *SeqGAN* upon mode collapse, was not significantly better than a jittered data set. Using less data jittering appeared to achieve equal results if the *Retraining* module was used. *SeqGAN*s likely introduced bias into the data set from sampling the more frequently used words and phrases within the target set.

12.1 Variance's Absence from Literature

The results of this experiment once again showed a correlation between variance of the target data set and the success of the GAN in producing synthetic data. This is especially important, given that this correlation has been shown to transcend the GAN approach used.

However so far in the literature, variance in the target data set has not been put forward as a contributor to the GANs ability to create synthetic data. In the papers written for *OptionGAN*s [123] and *InfoGAN*s [124], both wrote and provided evidence for the abilities for their GAN to operate in "complex" domains. Since this body of work has shown that variance is one of the major metrics to measure a GANs ability to avoid mode collapse, then likely "complex" domains should be considered as domains that require greater variance in the target data sets or a model such as *OptionGAN*s or *InfoGAN*s that require less variance in the given data sets to succeed.

12.2 Problems with *SeqGAN*s

The original *SeqGAN* paper [110] makes no mention of the requirement for data sets with high variance. A requirement that appears to be very strict. The module tests using the same data and task as those in the *SeqGAN* paper were equally successful on this platform. However, once applied in a domain of less definition, it appears *SeqGAN*s are not appropriate. Given that conventional media including forum posts (i.e. Stack Overflow), cited papers and the recommendation from the creator of modern GANs Ian Goodfellow, all point to *SeqGAN*s as the current best synthetic media production approach within the discrete data space, something from the conversation is missing. Upon review there are two major problems with Yu's original paper, that have surfaced in this thesis.

12.2.1 The Use of the BLEU Score Metric:

By assessing the output of the GAN using just the BLEU score between it and the target data set, Yu's team are not ensuring that the synthetic data is not simply a copy of the target data. As the BLEU metric which measures dissonance between two data sets would reward this outcome. Using the BLEU score here is not appropriate. Instead higher weighting should be given to closely matching n-grams from multiple target sentences than those that come from the same sentence.

12.2.2 Size of the Target Data Sets:

The Oracle NLL data set contains over 140,000 unique true words in the English language. To generate Chinese poems, Yu's team used 16,736 poems each with 4 lines of 20 characters, this corpus contains poems from a 4000 year time period, making it a high variance data set. To produce music, Yu used 695 folk songs (average folk song length is 4:04 minutes) with 88 pitch levels and a 0.4 second sampling speed. All of these data sets are very large and diverse. Yu's team did not in their paper, evaluate the effect a shrinking domain representation would have on their work. Which is out of place with how other papers have tested their approach. The teams of Bousmalis [114] and Tzeng [125] which were referenced in this body of work, ensured that their GANs did not copy the target domain. Though both teams did not test for a correlation between variance and accuracy.

It's important to note that this is not calling into question the validity of Yu's work. The results of his team were reproducible during this platform's modules test. However, it does question if the tests made by Yu and conclusions drawn by others when referencing *SeqGAN*s are not sufficient.

12.3 The Effect of Mode Collapse in the NLP Context

The way *SeqGAN*s behave once they undergo mode collapse appears to be very unique. The BLEU tests showed that the *SeqGAN* would upon mode collapse, copy the target data set instead of learning it. The synthetic data tha t was then used to create models obtained an accuracy similar to the jittered data set. Mode collapse in other GAN applications typically takes the form of the generator only being able to produce one datum repetitiously. However in this context, it appears that the generator produces a data set with variance.

How *SeqGAN*s are able to prevent the form of mode collapse that leads to no useful data being generated needs to be investigated. As if the worst outcome for the use of a GANs in the domain adaptation context is equal to that of jittering, which is the current best solution, then this is an optimal outcome.

12.4 Summary

The use of *SeqGAN*s was not an appropriate approach to NLP domain adaptation given the requirements of this platform. The mode collapse that occurs due too little variance in the target data runs counter to the user's original need for the platform. The ability of the retrainer module to still effectively utilise the faulty outputs from the *SeqGAN* provides significant credibility to the ability of the two-step approach as a whole.

However, despite whatever peripheral information or success these experiments have provided, they have left this platform without a way to handle discrete data sources.

13

Conclusion

13.1 Contributions of this Thesis

The following is a summary of the outcomes from the work conducted in this thesis.

- Provided strong evidence that the novel two-step transductive transfer learning pipeline composed of a domain adaptation and retrainer layer produces classifiers that are more accurate then the same model if produced by any one of those layers individually. Results such as the breaking of the MNIST to MNISTM benchmark demonstrate how the use of this pipeline improves upon current transductive transfer learning approaches.

- Conducted a comparison between data-boosting and model augmentation approaches for retraining, highlighted that that both approaches improved the model's final performance over a from-scratch training approach. However, model augmentation approaches appear to be more stable.

- Provided evidence that *SeqGAN*s are not an appropriate domain adaptation approach, due to a high requirement for variance in the target data set.

- Extended the *pixel-wise* GAN approach for the domain adaptation application, by incorporating more modern GAN techniques. These additions improved the accuracy of the synthetic representations created by the generator.

- Provided a metric that can be used in any domain adaptation approach, that measures the ability for the GAN to produce synthetic data based on the variance of the target data.

- Formalised a framework for a "ML-as-a-Service" platform, based around a two-step transductive transfer learning pipeline and interchangeable, but required supportive modules. With its low, target data requirement this platform will significantly decrease the barrier to entry the non-experts experience when implementing machine learning solutions.

13.2 Further Work

13.2.1 Completing the "ML-as-a-Service" Platform

With the failure to produce a viable NLP domain adaptation approach using GANs, the "ML-as-a-Service" platform remains incomplete.

The success of the imagery domain adaptation approach through the use of a *pixel-wise* GAN highlights the possibilities of an effective domain adaptation layer. However with the failure of *SeqGAN*s, an adaptation of current technologies or the creation of a novel domain adaptation approach for NLP use cases needs to be sought. All future approaches that are considered must allow the user to provide as little target data as the *pixel-wise* image approach, a requirement *SeqGAN*s could not satisfy.

Building an "as-a-Service" platform has requirements beyond the accuracy of the models it produces. ML application development targeted at non-expert users is currently very uncommon. Since non-expert users will have little prior experience with an "ML-as-a-Service" platform, a UI/UX design process must be implemented. Alongside this, the deployment of these platforms in a cloud environment will be necessary when the client base becomes multinational and data cannot be stored within other companies' environments due to strict data retention laws e.g. GDPR in the EU. An investigation into UI/UX design for "as-a-Service" platforms, cloud computing and the relevant laws to ensure compliance for this platform needs to be conducted.

13.2.2 Formalising Domain Adaptation Proofs

Unlike many other areas of ML research, domain adaptation does not appear to have standardised benchmarks to compare against. When building models classifying CIFAR, ImageNet and GLUE are all standards that have yet to appear in domain adaptation. Unofficially MNIST to MNIST-M is used for image domain adaptation, however it lacks strong coverage across many application requirements that still need to be tested for. The construction of a clear set of proved benchmarks must be undertaken. Reconducting these experiments using these benchmarks would further prove its success, or provide better understandings for the failure of the NLP approach.

13.2.3 Incorporating into Accenture's Infrastructure

Accenture has developed their own pipeline known as 'ML Core' and is incorporated within their major platforms. This pipeline utilises the traditional approach to ML and the issues that have already been identified as the need for this thesis. Incorporating the work captured here into this pipeline will be important for translating the value of this work into industry.

13.3 Conclusion

In order to produce an "ML-as-a-Service" platform, this thesis focused on the requirements that arise when the user is not an expert in machine learning. A non-expert user base necessitated the removal of the requirements currently imposed by the traditional machine learning pipeline that requires the user to produce large amounts of data (of any kind) in the target domain.

In an attempt to achieve this, the proposed novel two-step transductive transfer learning pipeline was implemented with mixed results. The first layer of the pipeline made use of a domain adaptation approached which supplemented the lack of target data by transforming large amounts of adjacent source domain data into the target domain.

For continuous data sets such as images, a *pixel-wise* GAN approach was taken. This approach was largely successful, notably demonstrating a 1.4% accuracy improvement in the MNIST to MNISTM classification benchmark over the original *pixel-wise* approach by Bousmalis and team [114]. The use of *FreezeOut* as the second layer in the pipeline, through module testing, proved to be the source of this increased accuracy. By retraining source domain models in the adjacent target domain, the models demonstrated a deeper knowledge base. This made them more robust to shifts in the target domain, when introduced unseen validation data sets.

However, the approach taken to transfer NLP data sets which are in the discrete data space was not as successful. This thesis sought to use a state-based, policy gradient style GAN. *SeqGAN*s are in nearly all up-to-date literature, accredited as the most effective approach for the synthetic reproduction of text data. However, due to a significant requirement for range and variance in the target data set, *SeqGAN*s were not appropriate in fulfilling the core goal of the platform. This failure however did create the basis for a metric based on range and variance that can be used to evaluate the quality of target data sets provided by the user.

Overall the results of this thesis and its two-step transductive transfer learning pipeline were highly positive. The success of the *pixel-wise* GAN approach coupled with the *FreezeOut* retraining process was not over-shadowed by the failure of the *SeqGAN* discrete data adaptation approach. Instead it highlighted how with the appropriate approach, this pipeline can significantly improve in an area that if success is not found in soon, the machine learning industry as a whole could stagnate.

Bibliography

1. Garavaglia, S. *Winning management support: cost justifying AI projects on Wall Street* in *Proceedings First International Conference on Artificial Intelligence Applications on Wall Street* (1991), 60–61. doi:`10.1109/AIAWS.1991.236572`.

2. Kawakami, H. & Hiraoka, T. *Contemplating AI Technologies from the Viewpoint of Benefit of Inconvenience* in *2013 Conference on Technologies and Applications of Artificial Intelligence* (2013), 335–336. doi:`10.1109/TAAI.2013.72`.

3. Akbar, A., Khan, A., Carrez, F. & Moessner, K. Predictive Analytics for Complex IoT Data Streams. *IEEE Internet of Things Journal* **4,** 1571–1582. ISSN: 2327-4662 (2017).

4. Coiera, E. *The Guide to Health Informatics* 3rd ed. Chap. 19. ISBN: 9781444170498 (CRC Press, 2015).

5. Fox, M. S. AI and expert system myths, legends, and facts. *IEEE Expert* **5,** 8–20. ISSN: 0885-9000 (1990).

6. Yukdowsky, E. Artificial Intelligence as a Positive and Negative Factor in Global Risk. *Global Catastrophic Risks* (2008).

7. Brooks, R. *et al.* Artificial Intelligence and Life in 2030. *One Hundred Year Study on Artificial Intelligence: Report of the,* 2015–2016 (2016).

8. Grosz, B. J. & Stone, P. *A Century Long Commitment to Assessing Artificial Intelligence and Its Impact on Society* (December Communications of the ACM (CACM), 2018).

9. Of America Merrill Lynch., B. *Robot Revolution - Global Robot & AI Primer.* 30–32. `http://www.bofaml.com/content/dam/boamlimages/documents/PDFs/robotics_and_ai_condensed_primer.pdf.` (2015).

10. Schmelzer, R. *will there be another ai winter?* 2018. `https://www.cognilytica.com/2018/02/22/will-another-ai-winter/.`

11. Hendler, J. Avoiding Another AI Winter. *IEEE Intelligent Systems* **23,** 2–4. ISSN: 1541-1672 (2008).

12. Wolf, M. J. *Before the crash: Early video game history* (Wayne State University Press, 2012).

13. Jackson, P. *Introduction To Expert Systems* 5th ed., 2–4. ISBN: 978-0-201-87686-4 (Addison Wesley, 1998).

14. Gershgorn, D. *the data that transformed ai research and possibly the world 2018* 2017. `https://qz.com/1034972/the-data-that-changed-the-direction-of-ai-research-and-possibly-the-world/`.

15. Zhang, Y., Yao, J. & Guan, H. Intelligent Cloud Resource Management with Deep Reinforcement Learning. *IEEE Cloud Computing,* 60–90 (2017).

16. Avdagic, I. & Hajdarevic, K. *Survey on machine learning algorithms as cloud service for CIDPS* in *2017 25th Telecommunication Forum (TELFOR)* (2017), 1–4. doi:`10.1109/TELFOR.2017.8249467`.

17. Hu, Y. *Outsourcing secured machine learning (ML)-as-a-service for causal impact analytics in multi-tenant public cloud* in *2017 2nd International Conference on Telecommunication and Networks (TEL-NET)* (2017), 1. doi:`10.1109/TEL-NET.2017.8343282`.

18. Siwicki, B. Elekta taps IBM Watson Health to bring AI capabilities to oncology tech. *Health Care IT.* `https://www.healthcareitnews.com/news/elekta-taps-ibm-watson-health-bring-ai-capabilities-oncology-tech` (2018).

19. Kleinberg, S. 5 ways voice assistance is shaping consumer behavior. `https://www.thinkwithgoogle.com/consumer-insights/voice-assistance-consumer-experience/` (2018).

20. Witten, I. H., Nevill-Manning, C. G. & Maulsby, D. *Interacting with learning agents: implications for ml from hci* in *Workshop on Machine Learning meets Human-Computer Interaction, ML* **96** (1996), 51–58.

21. Angra, S. & Ahuja, S. *Machine learning and its applications: a review* in *Big Data Analytics and Computational Intelligence (ICBDAC), 2017 International Conference on* (2017), 57–60.

22. Faggella, D. *How to Apply Machine Learning to Business Problems* 2018. `https://www.techemergence.com/how-to-apply-machine-learning-to-business-problems/`.

23. Brownlee, J. How Much Training Data is Required for Machine Learning? *Machine Learning Mastery.* `https://machinelearningmastery.com/much-training-data-required-machine-learning/` (2017).

24. Pan, S. J. & Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* **22,** 1345–1359. ISSN: 10414347 (2010).

25. Christopher D. Manning, P. R. & Schütze, H. *Introduction to Information Retrieval* 32. ISBN: 0521865719 (Cambridge University Press, 2008).

26. Bengio, Y., Courville, A. & Vincent, P. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* **35,** 1798–1828 (2013).

27. Bergstra, J. S., Bardenet, R., Bengio, Y. & Kégl, B. *Algorithms for hyper-parameter optimization* in *Advances in neural information processing systems* (2011), 2546–2554.

28. Provost, F. *Machine learning from imbalanced data sets 101* in *Proceedings of the AAAI'2000 workshop on imbalanced data sets* (2000), 1.

29. Richardson, L. & Ruby, S. *Restful Web Services* First, 10. ISBN: 9780596529260 (O'Reilly, 2007).

30. Yager, R. R. & Zadeh, L. A. *An introduction to fuzzy logic applications in intelligent systems* 1–8 (Springer Science & Business Media, 2012).

31. Teich, D. A. Self Service Business Intelligence Isn't Here, Artificial Intelligence May Be The Missing Piece. `https://www.forbes.com/sites/davidteich/2018/07/26/self-service-business-intelligence-isnt-here-artificial-intelligence-may-be-the-missing-piece/#3caabd0e5cbd` (2018).

32. Lopez, A. Natural Language Understanding. `https://pdfs.semanticscholar.org/presentation/22dc/c06e9643fd539e579e6553e8af90d6f61c36.pdf` (2018).

33. Chang, M. & Chang, M. iWordNet: A New Approach to Cognitive Science and Artificial Intelligence. *Advances in Artificial Intelligence* **2017,** 1–3 (2017).

34. Feng, S. Machine Learning Algorithms (2015).

35. Gori, M. *Machine Learning: A Constraint-based Approach* 50–60 (Morgan Kaufmann, 2017).

36. Anderson, D., Lunt, T. F., Javitz, H., Tamaru, A., Valdes, A., *et al. Detecting Unusual Program Behavior Using the Statistical Component of the Next-generation Intrusion Detection Expert Systems (NIDES)* 7–8 (SRI International. Computer Science Laboratory, 1995).

37. Eskin, E. *Anomaly Detection over Noisy Data using Learned Probability Distributions* in *In Proceedings of the International Conference on Machine Learning* (Morgan Kaufmann, 2000), 255–262.

38. Suthaharan, S. Machine learning models and algorithms for big data classification. *Integr. Ser. Inf. Syst* **36,** 1–12 (2016).

39. Suthaharan, S. Machine learning models and algorithms for big data classification. *Integr. Ser. Inf. Syst* **36,** 1–12 (2016).

40. Pang, B., Lee, L. & Vaithyanathan, S. *Thumbs up?: sentiment classification using machine learning techniques* in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10* (2002), 79–86.

41. Ericson, G., Rohm, W. & Jenks, A. *How to choose algorithms for Microsoft Azure Machine Learning* 1–3. `https://docs.microsoft.com/en-us/azure/machine-learning/studio/algorithm-choice` (Microsoft, 2018).

42. Pilotte, P. *Analytics-driven embedded systems - Developing analytics and prescriptive controls* 2018. `http://www.embedded-computing.com/embedded-computing-design/analytics-driven-embedded-systems-part-2-developing-analytics-and-prescriptive-controls`.

43. Wolpert, D. H. & Macready, W. G. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation* **1,** 67–82 (1997).

44. Geman, S., Bienenstock, E. & Doursat, R. Neural networks and the bias/variance dilemma. *Neural computation* **4,** 4–12 (1992).

45. Dietterich, T. G. & Kong, E. B. *Machine learning bias, statistical bias, and statistical variance of decision tree algorithms* tech. rep. (Technical report, Department of Computer Science, Oregon State University, 1995).

46. Valentini, G. & Dietterich, T. G. Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods. *Journal of Machine Learning Research* **5,** 725–775 (2004).

47. Kearns, M. J. *The computational complexity of machine learning* 1–4 (MIT press, 1990).

48. Holte, R. C. Very simple classification rules perform well on most commonly used datasets. *Machine learning* **11,** 63–90 (1993).

49. Tyka, A. M. C. O. M. Inceptionism: Going Deeper into Neural Networks. *Google Research Blog* (2015).

50. Bellman, R. *The theory of dynamic programming* tech. rep. (RAND Corp Santa Monica CA, 1954).

51. Theodoridis, S., Koutroumbas, K., *et al.* Pattern recognition. *IEEE Transactions on Neural Networks* **19,** 376 (2008).

52. Hughes, G. On the mean accuracy of statistical pattern recognizers. *IEEE transactions on information theory* **14,** 55–63 (1968).

53. Schölkopf, B. *The kernel trick for distances* in *Advances in neural information processing systems* (2001), 301–307.

54. Deng, L., Yu, D., *et al.* Deep learning: methods and applications. *Foundations and Trends® in Signal Processing* **7,** 199–200 (2014).

55. Chienti, M. *Are deep neural nets "Software 2.0"?* 2017. `https://www.michaelchimenti` `com/2017/11/deep-neural-nets-software-2-0/`.

56. Pal, S. K. & Mitra, S. Multilayer Perceptron, Fuzzy Sets, Classifiaction (1992).

57. Sharma, S. *What the Hell is Perceptron?* 2018. `https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53`.

58. Robert, C. *Machine learning, a probabilistic perspective* 2014.

59. Walkarn, U. *An Intuitive Explanation of Convolutional Neural Networks* 2017. `https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/`.

60. Matsugu, M., Mori, K., Mitari, Y. & Kaneda, Y. Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Networks* **16,** 555–559 (2003).

61. Grel, T. *Region of interest pooling explained* 2018. `https://deepsense.ai/region-of-interest-pooling-explained/`.

62. Long, J., Shelhamer, E. & Darrell, T. *Fully convolutional networks for semantic segmentation* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), 3431–3440.

63. Arpathy, K. *Convulution Neural Networks for Visual Recognition* 2018. `https://cs231n.github.io/convolutional-networks/`.

64. Ciregan, D., Meier, U. & Schmidhuber, J. *Multi-column deep neural networks for image classification* in *2012 IEEE Conference on Computer Vision and Pattern Recognition* (June 2012), 3642–3649. doi:`10.1109/CVPR.2012.6248110`.

65. Farfade, S. S., Saberian, M. J. & Li, L.-J. *Multi-view face detection using deep convolutional neural networks* in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval* (2015), 643–650.

66. Hecht-Nielsen, R. in *Neural networks for perception* 65–93 (Elsevier, 1992).

67. Kiwiel, K. C. Convergence and efficiency of subgradient methods for quasiconvex minimization. *Mathematical programming* **90,** 1–25 (2001).

68. Huang, G. *et al.* Snapshot ensembles: Train 1, get M for free. *arXiv preprint arXiv:1704.00109,* 1 (2017).

69. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* **15,** 1929–1958 (2014).

70. Xiao, T., Li, H., Ouyang, W. & Wang, X. *Learning deep feature representations with domain guided dropout for person re-identification* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), 1249–1258.

71. Simonyan, K. & Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

72. Koistinen, P. & Holmström, L. *Kernel regression and backpropagation training with noise* in *Advances in Neural Information Processing Systems* (1992), 1033–1039.

73. Weizenbaum, J. ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM* **9,** 36–45 (1966).

74. Young, T., Hazarika, D., Poria, S. & Cambria, E. Recent trends in deep learning based natural language processing. *arXiv preprint arXiv:1708.02709* (2017).

75. Sutton, C., McCallum, A., *et al.* An introduction to conditional random fields. *Foundations and Trends® in Machine Learning* **4,** 278–286 (2012).

76. Ng, A. *CS229 Lecture notes - Generative Learning algorithms* 2013.

77. Ó Séaghdha, D. & Korhonen, A. Probabilistic distributional semantics with latent variable models. *Computational Linguistics* **40,** 587–631 (2014).

78. Collobert, R. *et al.* Natural Language Processing (almost) from Scratch. *CoRR* **abs/1103.0398.** `http://arxiv.org/abs/1103.0398` (2011).

79. Zafrany, S. *NLP Gen (Word2Vec)* 2018.

80. Itti, L., Koch, C. & Niebur, E. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence* **20,** 1254–1259 (1998).

81. Zou, X., Zhao, X., Yang, Y. & Li, N. Learning-based visual saliency model for detecting diabetic macular edema in retinal image. *Computational intelligence and neuroscience* **2016,** 1 (2016).

82. Krizhevsky, A., Sutskever, I. & Hinton, G. E. *Imagenet classification with deep convolutional neural networks* in *Advances in neural information processing systems* (2012), 1097–1105.

83. Liu, Y., Cai, Q., Zhu, X., Cao, J. & Li, H. *Saliency detection using two-stage scoring* in *Image Processing (ICIP), 2015 IEEE International Conference on* (2015), 4062–4066.

84. Qin, Y., Lu, H., Xu, Y. & Wang, H. *Saliency detection via cellular automata* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), 110–119.

85. Fu, K., Gong, C., Gu, I. Y., Yang, J. & Shi, P. *Salient object detection using normalized cut and geodesics* in *Image Processing (ICIP), 2015 IEEE International Conference on* (2015), 1100–1104.

86. Gao, D., Han, S. & Vasconcelos, N. Discriminant saliency, the detection of suspicious coincidences, and applications to visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **31,** 989–1005 (2009).

87. Li, C., Yuan, Y., Cai, W., Xia, Y. & Dagan Feng, D. *Robust saliency detection via regularized random walks ranking* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), 2710–2717.

88. Muhammed, M. A. E., Ahmed, A. A. & Khalid, T. A. *Benchmark analysis of popular ImageNet classification deep CNN architectures* in *Smart Technologies For Smart Nation (SmartTechCon), 2017 International Conference On* (2017), 902–907.

89. Huang, G., Liu, Z., Van Der Maaten, L. & Weinberger, K. Q. *Densely Connected Convolutional Networks.* in *CVPR* **1** (2017), 3.

90. Pan, F. & Zhang, L. New image super-resolution scheme based on residual error restoration by neural networks. *Optical Engineering* **42,** 3038–3047 (2003).

91. Peyrard, C., Mamalet, F. & Garcia, C. *A Comparison between Multi-Layer Perceptrons and Convolutional Neural Networks for Text Image Super-Resolution.* in *VISAPP (1)* (2015), 84–91.

92. Medina, E., Petraglia, M. R., Gomes, J. G. R. C. & Petraglia, A. *Comparison of CNN and MLP classifiers for algae detection in underwater pipelines* in *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)* (Nov. 2017), 1–6. doi:10.1109/IPTA.2017.8310098.

93. Köpüklü, O., Köse, N. & Rigoll, G. Motion Fused Frames: Data Level Fusion Strategy for Hand Gesture Recognition. *arXiv preprint arXiv:1804.07187* (2018).

94. Zhang, C. *et al.* A hybrid MLP-CNN classifier for very fine resolution remotely sensed image classification. *ISPRS Journal of Photogrammetry and Remote Sensing* **140.** Geospatial Computer Vision, 133–144. ISSN: 0924-2716 (2018).

95. Wiedemann, G. & Heyer, G. Page Stream Segmentation with Convolutional Neural Nets Combining Textual and Visual Features. *arXiv preprint arXiv:1710.03006* (2017).

96. Wang, T., Wu, D. J., Coates, A. & Ng, A. Y. *End-to-end text recognition with convolutional neural networks* in *Pattern Recognition (ICPR), 2012 21st International Conference on* (2012), 3304–3308.

97. Valiant, L. G. A theory of the learnable. *Communications of the ACM* **27,** 1134–1142 (1984).

98. Tsochantaridis, I., Hofmann, T., Joachims, T. & Altun, Y. *Support vector machine learning for interdependent and structured output spaces* in *Proceedings of the twenty-first international conference on Machine learning* (2004), 104.

99. Parisi, G. I., Kemker, R., Part, J. L., Kanan, C. & Wermter, S. Continual Lifelong Learning with Neural Networks: A Review. *arXiv preprint arXiv:1802.07569* (2018).

100. Jiang, J. & Zhai, C. *Instance weighting for domain adaptation in NLP* in *Proceedings of the 45th annual meeting of the association of computational linguistics* (2007), 264–271.

101. Cortes, C., Mohri, M., Riley, M. & Rostamizadeh, A. *Sample selection bias correction theory* in *International Conference on Algorithmic Learning Theory* (2008), 38–53.

102. Motiian, S., Piccirilli, M., Adjeroh, D. A. & Doretto, G. *Unified deep supervised domain adaptation and generalization* in *The IEEE International Conference on Computer Vision (ICCV)* **2** (2017), 3.

103. Ben-David, S., Blitzer, J., Crammer, K. & Pereira, F. *Analysis of representations for domain adaptation* in *Advances in neural information processing systems* (2007), 137–144.

104. Ajakan, H., Germain, P., Larochelle, H., Laviolette, F. & Marchand, M. Domain-adversarial neural networks. *arXiv preprint arXiv:1412.4446* (2014).

105. Ganin, Y. & Lempitsky, V. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495* (2014).

106. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. & Courville, A. C. *Improved training of wasserstein gans* in *Advances in Neural Information Processing Systems* (2017), 5767–5777.

107. Nettleton, D. F., Orriols-Puig, A. & Fornells, A. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial intelligence review* **33,** 275–306 (2010).

108. Goodfellow, I. *et al. Generative adversarial nets* in *Advances in neural information processing systems* (2014), 2672–2680.

109. Zhu, J.-Y., Park, T., Isola, P. & Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint* (2017).

110. Yu, L., Zhang, W., Wang, J. & Yu, Y. *SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient.* in *AAAI* (2017), 2852–2858.

111. Che, T. *et al.* Maximum-likelihood augmented discrete generative adversarial networks. *arXiv preprint arXiv:1702.07983* (2017).

112. Max, S. *Optimal Control for the Automated Design of Machine Learning Models* PhD thesis (University Of Sydney, 2017).

113. Le, Q. & Mikolov, T. *Distributed representations of sentences and documents* in *International Conference on Machine Learning* (2014), 1188–1196.

114. Bousmalis, K., Silberman, N., Dohan, D., Erhan, D. & Krishnan, D. *Unsupervised pixel-level domain adaptation with generative adversarial networks* in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* **1** (2017), 7.

115. Conneau, A., Schwenk, H., Barrault, L. & Lecun, Y. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781* (2016).

116. Radford, A., Metz, L. & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

117. Shi, W. *et al. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), 1874–1883.

118. Salimans, T. *et al. Improved techniques for training gans* in *Advances in Neural Information Processing Systems* (2016), 2234–2242.

119. Yao, Y. & Doretto, G. *Boosting for transfer learning with multiple sources* in *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on* (2010), 1855–1862.

120. Brock, A., Lim, T., Ritchie, J. M. & Weston, N. FreezeOut: Accelerate Training by Progressively Freezing Layers. *arXiv preprint arXiv:1706.04983* (2017).

121. Maas, A. L. *et al. Learning Word Vectors for Sentiment Analysis* in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies* (Association for Computational Linguistics, Portland, Oregon, USA, June 2011), 142–150. `http://www.aclweb.org/anthology/P11-1015`.

122. Mettomäki, A. Comparison of representations in Barack Obama and Donald Trump's inaugural addresses. *University of Jyväskylä, Department of Language and Communication Studies* (2017).

123. Henderson, P. *et al.* Optiongan: Learning joint reward-policy options using generative adversarial inverse reinforcement learning. *arXiv preprint arXiv:1709.06683* (2017).

124. Chen, X. *et al. Infogan: Interpretable representation learning by information maximizing generative adversarial nets* in *Advances in neural information processing systems* (2016), 2172–2180.

125. Tzeng, E., Hoffman, J., Saenko, K. & Darrell, T. *Adversarial discriminative domain adaptation* in *Computer Vision and Pattern Recognition (CVPR)* **1** (2017), 4.

126. Huang, G., Sun, Y., Liu, Z., Sedra, D. & Weinberger, K. Q. *Deep networks with stochastic depth* in *European Conference on Computer Vision* (2016), 646–661.

127. Springenberg, J. T., Dosovitskiy, A., Brox, T. & Riedmiller, M. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806* (2014).

Appendix A

Source Model Creation Module Experiments - *DenseNet*

A.1 Method

A.1.1 1. Testing Image Recognition

These tests will evaluate *DenseNet*s for simple and complex IR tasks. When *DenseNet* was first detailed it was tested against the CIFAR10/100 and the SVHN data sets. This implementation of *DenseNet* will evaluate against the same data sets.

The follow test for each data set will be undertaken.

1. Download and split data set into a folder, one for each label.
2. Read data into *PyTorch* dataloader, split into batches of size 64.
3. Set Auto-Tuner to explore Dense Layer size (max 250).
4. Set *DenseNet* output layer to 10 nodes.
5. Run process for 50 tuning epochs of 100 training epochs.

For all the above tests, comparisons will be made against other leader:

- ResNet (G. Huang's entry in the ImageNet competition 2016) [[126]]
- Multi-Layered Perceptron made using *SkLearn* python package.
- The All Convolutional Net [[127]]

A.1.2 2. Testing Natural Language Processing

A single test in the NLP will evaluate if *DenseNet*s are able to learn intent from written movie reviews. The *Stanford IMDB movie review corpus* is vectorised using the *Doc2Vec* in the pre-processing module. The *DenseNet* model will receive these vectors and map them to a binary sentiment, positive or negative.

1. Download *Stanford IMDB movie review corpus* and passthrough *Doc2Vec*.
2. Store *Doc2Vec* model as layer between data set and *DenseNet*.
3. Train and validate *DenseNet* as normal, passing reviews through *Doc2Vec* beforehand.

A.1.3 Results

The first experiment trained a *DenseNet* with a growth rate of 12, on the SVHN dataset. The *DenseNet* model was embedded within the autotuner which selected 76 dense layers as the optimal configuration. However, results showed that other equal optimal results were found continuously ranging from 14 to 192 dense layers. A lot of time was spent in the auto-tuner finding results that were of the same accuracy. The auto-tuner should have left once the gradient of the search flattened out, it did not.

TABLE A.1: *SVHN Tests*

Model Type	Accuracy (%)	Sensitivity (%)	Specificity (%)
DenseNet	99.94	99.91	98.35
ResNet	99.89	98.20	98.71
MLP	96.90	94.10	92.39

The 2nd test used the CIFAR10 data set. The growth rate again was 12 and the the auto-tuner selected 140 dense layers as optimal. In this experiment the auto-tuner was more precise and was able to select 82 as an optimal result without extensive search of equal opportunities.

TABLE A.2: *CIFAR10 Tests*

Model Type	Accuracy (%)	Sensitivity (%)	Specificity (%)
DenseNet	94.88	93.61	88.31
ResNet	88.13	87.31	88.99
MLP	49.18	48.86	49.24
All CNN	90.91	No Data	No Data

The final experiment using CIFAR100, in terms of the auto-tuner's ability to find an optimal hyper parameter space, performed the same as CIFAR10. Due to the complexity of the experiment, *DenseNet*'s growth rate was extended to 24 which significantly increased the time and space complexity of *DenseNet*. This experiment took up 48gb of RAM representing a jump from 0.8Mil to 15.3Mil captured parameters. The final depth of the net was 220 dense layers.

TABLE A.3: *CIFAR100 Tests*

Model Type	**Accuracy (%)**	**Sensitivity (%)**	**Specificity (%)**
DenseNet	75.84	88.90	68.34
ResNet	64.42	61.05	68.43
MLP	51.18	45.34	49.03
All CNN	61.22	No Data	No Data

A.1.4 Evaluation

The metrics for measuring *DenseNet*'s capacity to act as this platform's generic model include accuracy, time and space complexity and the input data's invariance. While accuracy across all tests was higher than those recorded by other well known models, the CIFAR100 test raised concerns about space complexity. In order to handle the higher domain requirements of a large, many object input such as CIFAR100 the growth rate was increased and a larger amount of dense layers were selected. The 1912% increase of parameters produce a high space requirement that is not commonly seen in most commercial applications. However, the 11% accuracy improvement and significantly more stable sensitivity and specificity associated with those results highlights the necessity for that space requirement.

These results represent a confirmation that across the IR and NLP space that not only is a CNN approach applicable, but that of that family *DenseNets* are the currently best performing model. While the research conducted in the literary review already showed this to be true, it was important to make several confirmations. These results ensure that this module's *DenseNet* setup can be applied generically. Limited automation of depth selection in order to find the best model setup for a given task stretches this generic application significantly.

Appendix B

Pre-Processing

B.1 Method

The following two sets of experiments evaluated the functioning of the Pre-Processing and it's effectiveness in providing standardised, high entropy data to the rest of the platform.

B.1.1 Image Pre-Processing

Using the CIFAR10 dataset.

1. Ensure all images are made encoded using the HSL spectrum.
2. The effect the *BatchNorm* layer will have.
3. The reduction of variance between results before and after shuffling and stratification of the training data is applied.

B.1.2 Natural Language Processing Pre-Processing

1. Train and Validate *DenseNet* using the 'Bag Of Words' vectors provided by in the movie review data set.
2. Train and Validate *DenseNet* using the *Doc2Vec* vectors created by the *Pre-Processing* module.

B.2 Results

B.2.1 Image PreProcessing

The effect of the *BatchNorm* layer was observable in the distribution of the images before and after normalisation.

The results for CIFAR10 high

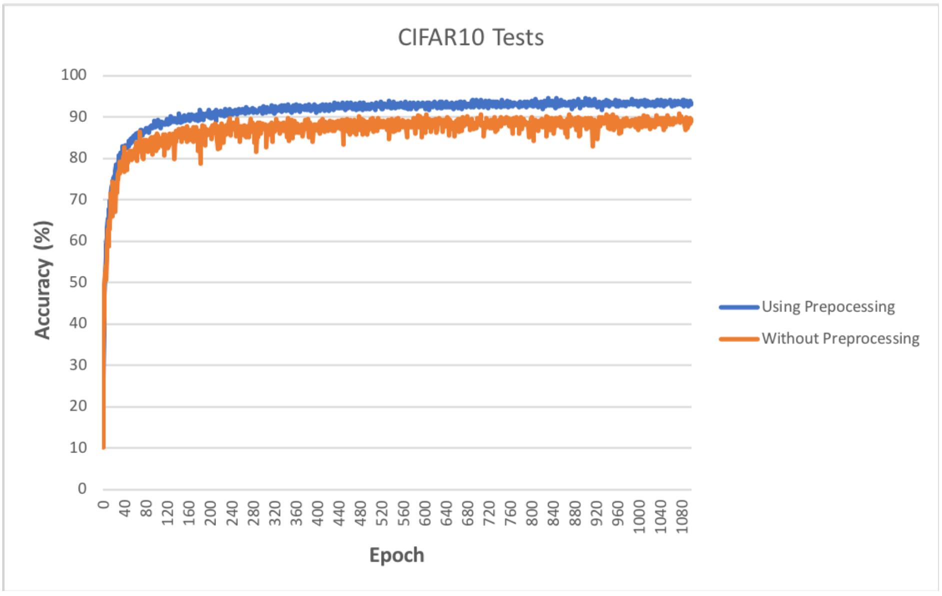


FIGURE B.1: CIFAR 10 results before and after preprocessing

TABLE B.1: *CIFAR10 Tests*

	F1 Score (%)	Variance	Recall (%)
Before Prepocessing	89.56	83.99	94.22
After Prepocessing	94.88	93.61	88.31

B.2.2 Text PreProcessing

TABLE B.2: *IMBD Movie Reviews*

	F1 Score (%)	Variance	Recall (%)
Using Bag of Words	90.3	91.4	89.3
Using *Word2Vec*	90.8	91.5	90.1
Using *Doc2Vec*	91.7	95.0	88.6

B.3 Evaluation

These results provide validation for the *Pre-Processing* module that will enable it to be deployed as the data standardisation layer in the "ML-as-a-Service" platform tests. The success of the individual components including *DenseNet*s, *Doc2Vec* and *BatchNorm* layers are all expected given they come from well cited research. However, these tests ensured that their implementation matched expected results. In order to ensure that assumptions made in other modules are fulfilled.

Appendix C

Domain Adaptation - Data Augmentation Approach Using *TrAdaBoost*

C.1 Approach

Traditionally *TrAdaBoost* is used as an instance transfer training method for a transductive transfer learning case. It's primary purpose is to make use of knowledge of the same domain that has been split across several sets. It has not been applied to a problem where the primary purpose is not to ratify domains, but instead to retrain a knowledgeable model by boosting data of importance. Traditionally *TrAdaBoost* increases the weight of *Same Distribution Data* and decreases the weight of *Diff-Distribution Data* between the source and target domain, before retraining the model.

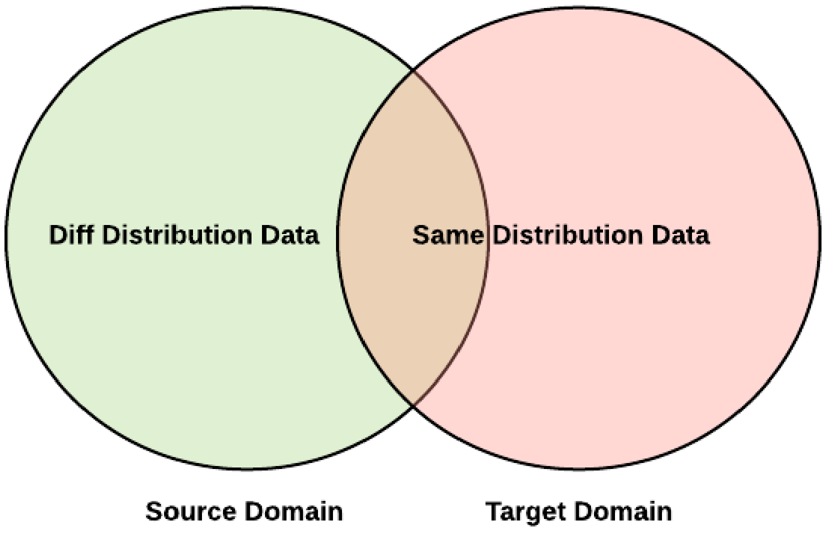


FIGURE C.1: Data distribution between source and target data set

In both scenarios, the data for retraining will be placed into two categories. The first contains *Task Relevant Data* that is specific to the task that the model will be applied to. The first hypothesis of the modified *TrAdaBoost* algorithm is that boosting task relevant data leads to higher precision in the model. The second category contains *Context Relevant Data*, which helps shape the models understanding of the domain. The second hypothesis of this modified algorithm, is that boosting context relevant data will reduce the variance of the model.

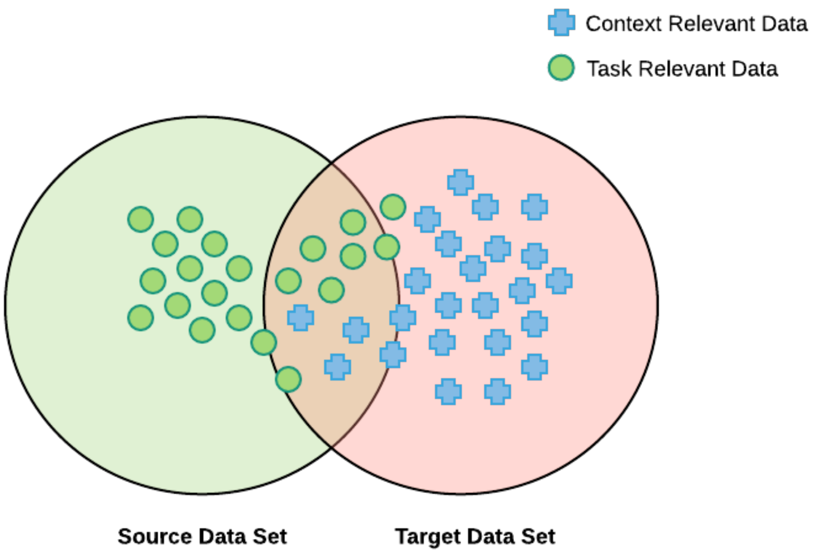


FIGURE C.2: Distribution of task and context relevant data in source and target data sets

Exclusively in the transductive transfer learning scenario the dissonance between the source and target domains has already been ratified in the domain adaptation module. However, *TrAdaBoost* may be able to boost the weight of synthetic data that sits more completely in the target data set. This approach which would improve the precision of the model, would be in line with *TrAdaBoost*'s traditional design.

TrAdaBoost traditional algorithm is as follows. Note that what has been referred previously in this paper as the source and target domain is referred to in the below algorithm as the *Diff Distribution Domain* ($\mathcal{D}_D$) and *Same Distribution Domain* ($\mathcal{D}_S$). This format will only be used in the below algorithm and not again.

Algorithm 3 TrAdaBoost

```
Input: The two labelled data sets from $\mathcal{S}_D$ and $\mathcal{S}_S$, the an unknown data set S, a
newly initialised model M, an associated trainer O and the maximum number of
iterations N.
Procedure:
  for t = 1,...,N do
    1. Set $\mathbf{p}^t = \mathbf{w}^t(\sum_{i=1}^{n+m} w_i^t)$.
    2. Call Learner providing it the combined training set $\mathcal{S}_T = \mathcal{S}_D + \mathcal{S}_S$ with the
       distribution $\mathbf{p}_t$ over $\mathcal{S}_T$ and the data set S. Then, get back a hypothesis $\mathbf{h}_t$ : X $\to$
       Y (or [0, 1] by confidence).
    3. Calculate the error of $h_t$ on $\mathcal{S}_S$:
       $$\epsilon_t = \sum_{i=n+1}^{n+m} \frac{w_i^t \cdot |h_t(x_i) - c(x_i)|}{\sum_{i=n+1}^{n+m} w_i^t} \qquad (C.1)$$
    4. Set $\beta_t = \epsilon_t/(1-\epsilon_t)$ and $\beta = 1/(1+\sqrt{2ln(\frac{n}{N})})$
       Note that, t is required to be less than 1/2
    5. Update the new weight vectors
       $$w_i^{t+1} = \begin{cases} w_i^t \beta^{|h_t(x_i)-c(x_i)|}, \ 1 \leq i \leq n \\ w_i^t \beta^{-|h_t(x_i)-c(x_i)|}, \ n+1 \leq i \leq n+m \end{cases} \qquad (C.2)$$
  end for
Output: The Hypothesis
  $$h_f(x) = \begin{cases} 1, \ \prod_t^N = [N/2]\beta_t^{h_t}(x) \geq \prod_t^N = [N/2]\beta_t^{-}\frac{1}{2} \\ 0, \ otherwise \end{cases} \qquad (C.3)$$
```

Currently the *TrAdaBoost* algorithm is building its hypothesis in order to re-weight the importance of data that either exists in the same domain of the target, or exists in the source domain but positively contributes to the model's performance. This instance learning approach could be used to avoid the domain adaptation layer. However, as explored in the literary review, instance transfer can lead to negative transfer learning often when the initial source data set contains bias.

The modifications made to *TrAdaBoost* include alterations to the inputs of the original algorithm and changes to the hypothesis when in the inductive transfer scenario.

In the transductive transfer scenario, the process will only provide the learner with the target and synthetic data set. The synthetic domain will mimic the supposed source domain from the initial algorithm. By using this spread of data, the hypothesis will extend to include the variance between the two domains in the error.
In the inductive transfer method, the source and target data set are the same, with no unlabelled data set to provide. Here boosting that leads to increase precision will be from task relevant data. A positive outcome from the inductive transfer experiments will confirm this.

C.1.1 Retrainer: *TrAdaBoost*

1. MNIST on Different Tasks

In both the inductive and transductive transfer learning scenarios, finding and boosting task relevant data will be a core focus. In the inductive context the focus is purely on task related data. Therefore testing inductive transfer learning approaches, is the same as testing the ability for the retrainers ability to transfer task relevant features.

This setup will evaluate the effectiveness of the modified *TrAdaBoost* for the inductive transfer learning scenario. A model that initially, using the Auto-Tune pipeline, trained to identify MNIST numbers 0-9 is retrained for two different tasks. The first task is to identify the 0's of the number set, the second task is to identify sets of numbers (0,1,2,3,4,5,6,7,8,9).

1. Using MNIST data set with a binary label space of 0's and the rest.
   (a) Place two split and shuffled MNIST data sets using above label space as $\mathcal{S}_T$ and $\mathcal{S}$.
   (b) Sub in the pre-trained MNIST model as M, using PyTorch's optimiser for O.
   (c) Retrain model using *TrAdaBoost*.
2. Using MNIST data set with a multiclass label space of three sets (0,1,2,3,4,5,6,7,8,9).
   (a) Place two split and shuffled MNIST data sets using above label spaces as $\mathcal{S}_T$ and $\mathcal{S}$.
   (b) Sub in the pre-trained MNIST model as M, using PyTorch's optimiser for O.
   (c) Retrain model using *TrAdaBoost*.

2. Using Synthetic MNIST-M Data Against True MNIST-M Data

This setup will evaluate *TrAdaBoost*'s ability to retrain models for a new domain. The algorithms ability to positively retrain a model regardless of the dissonance needs to be evaluated.

1. Using the MNIST-M data with labels 0-9.
   (a) Place shuffled target MNIST-M data set into $\mathcal{S}_T$ and the synthetic data into $\mathcal{S}$.
   (b) Sub in the pre-trained MNIST model as M, using PyTorch's optimiser for O.
   (c) Retrain model using *TrAdaBoost*.
2. Using Synthetic data with labels 0-9 that has a 98% confidence in the MNIST-M domain.
   (a) Place shuffled target MNIST-M data set into $\mathcal{S}_T$ and the synthetic data into $\mathcal{S}$.
   (b) Sub in the pre-trained MNIST model as M, using PyTorch's optimiser for O.
   (c) Retrain model using *TrAdaBoost*.
3. Using MNIST data with labels 0-9.
   (a) Place shuffled target MNIST-M data set into $\mathcal{S}_T$ and the synthetic data into $\mathcal{S}$.
   (b) Sub in the pre-trained MNIST model as M, using PyTorch's optimiser for O.
   (c) Retrain model using *TrAdaBoost*.

C.2 Results

C.2.1 Experimental Setup 1: Inductive Transfer Learning using MNIST

The first inductive transfer learning test compares *TrAdaBoost*'s retraining of a MNIST model against a model trained from scratch for a simple binary task, evaluating 0's in the MNIST data set. In this experiment *TrAdaBoost* produced a final model that initially trained faster then the from scratch model, but was slower to reach convergence. The overall accuracy of both models was the same within an expected variance.

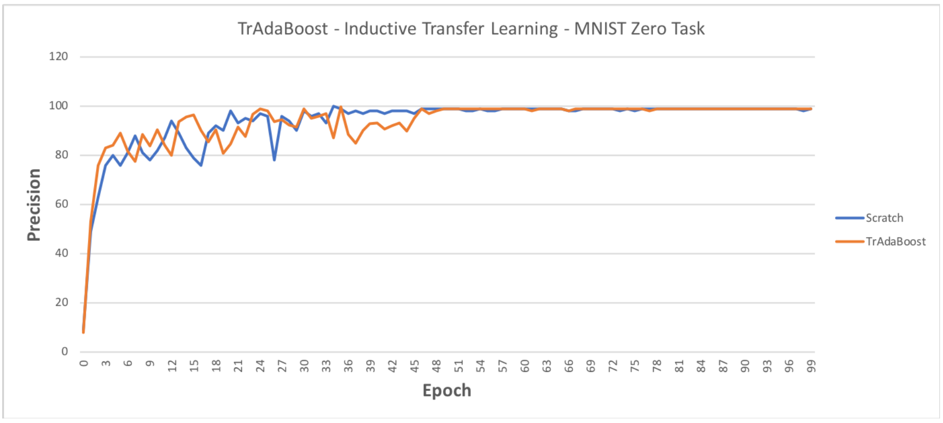


FIGURE C.3: Experiment evaluating *TrAdaBoost* inductive transfer learning ability in retraining a MNIST domain model for the task of evaluating 0's in the MNIST data set. Compared against a model trained from scratch.

The second experiment accesses the ability of *TrAdaBoost* on more complex tasks. In this case sets of numbers are grouped together. *TrAdaBoost* performed similar in this test, failing to reach convergence before the from scratch model, while still achieving a similar overall accuracy.

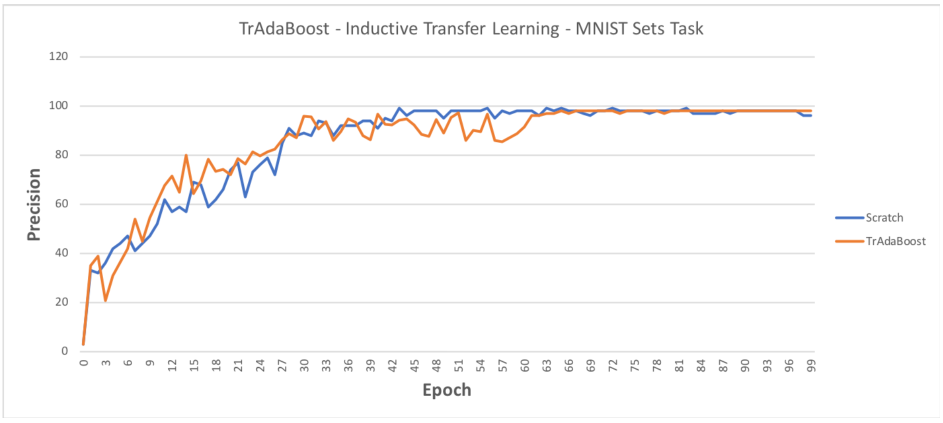


FIGURE C.4: Experiment evaluating *TrAdaBoost* inductive transfer learning ability in retraining a MNIST domain model for the task of evaluating sets of numbers in the MNIST data set. Compared against a model trained from scratch.

C.2.2 Experimental Setup 2: Transductive Transfer Learning using Synthetic Data in the MNISTM Domain

The first test using *TrAdaBoost* for transductive transfer learning used the perfect synthetic representation of the MNIST-M domain. The results showed a model that behaved very similar to the from-scratch model, but fluctuated heavily around the convergence point. The convergence point also sat below that of the from scratch model by a marginal amount. The *TrAdaBoost* model appeared to reach a convergence point faster than the from scratch model.

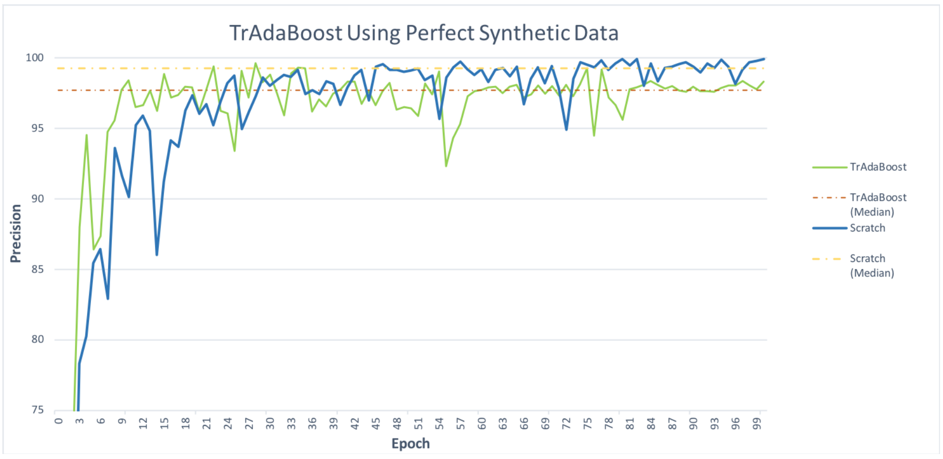


FIGURE C.5: A comparison of using *TrAdaBoost* for transductive transfer learning on perfect synthetic data set vs. the same data set used to train a model from scratch.

The second test used the synthetic data set produced by the domain adaptation module. This data set which was able to be classified by a MNIST-M model at an accuracy of 98%. The results from *TrAdaBoost* while as accurate as the model trained from scratch was slow to reach convergence.

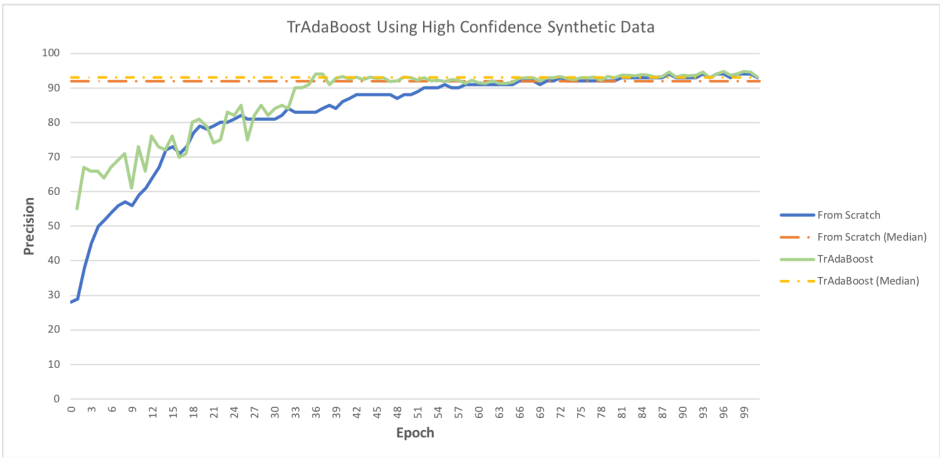


FIGURE C.6: A comparison of using *TrAdaBoost* for transductive transfer learning on an high confidence synthetic data set vs. the same data set used to train a model from scratch.

The third test evaluated the input of a poor synthetic representation of the MNIST-M domain, this was mimicked by using MNIST data set for a poor synthetic representation.

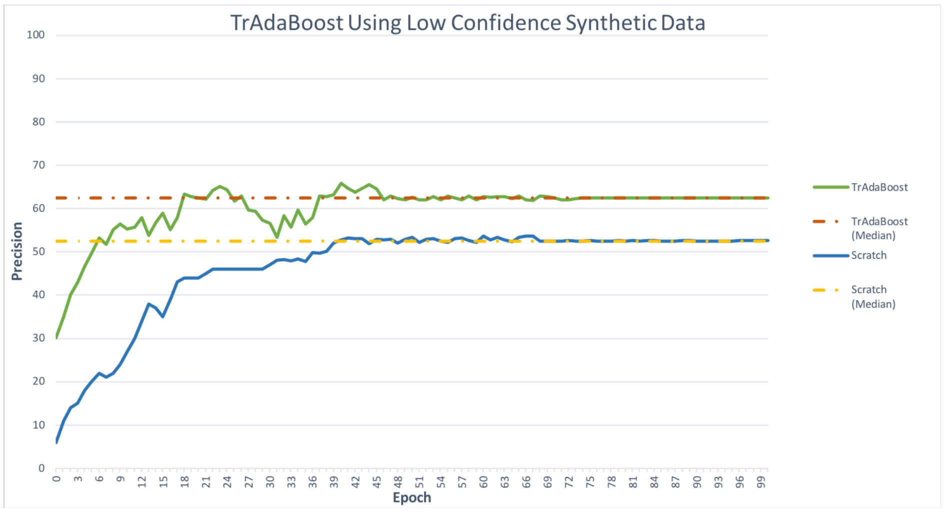


FIGURE C.7: A repeat test using poorer quality synthetic data to represent the MNIST-M domain.

There was a clear improvement over the from scratch trained model. Due to the large disparity between the results using poor and high quality synthetic data, the initial tests results were rerun.

These re-tests included manually altering batch sizes, waiting longer for convergence and a careful critic of CPU worker delegation. The alterations led to no significant improvement.

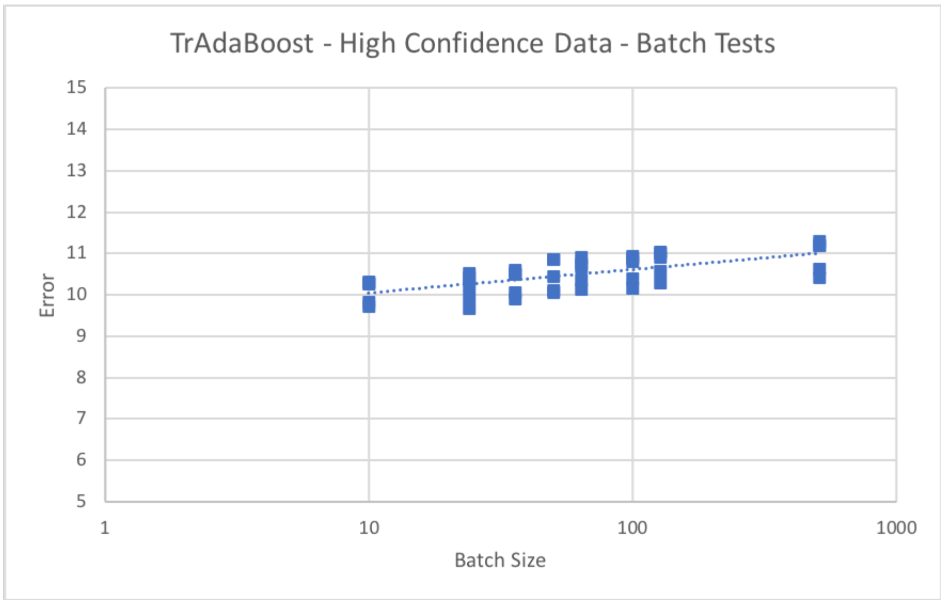


FIGURE C.8: Median accuracy of *TrAdaBoost* using different batch sizes

C.3 Evaluation

A major source of interference can be inferred from the unsatisfactory performance of *TrAdaBoost* in these experiments. The source of the first interference can be seen in the irregular patterns of the learning in all the experiments. The cause of this can be traced back to the covariance of the synthetic data in relation to the target domain. Despite the best efforts of the domain adaptation module, the synthetic data will likely not sit one-to-one with the target domain.

Due to the noise in the data set, *TrAdaBoost*'s continually fluctuating hypothesis is unable to converge. Critically this appears to escalate when the variance between data sets is exaggerated, as seen in the larger fluctuations in the transductive scenario's second experiment. The outcome of this observation goes against what was initially hypothesised, *TrAdaBoost* can not cover any shortcomings from the domain adaptation module. As the lack of convergence caused by the fluctuating hypothesis drastically reduces the effectiveness of the algorithm.

C.4 Conclusion

TrAdaBoost which was initially only designed as a transductive transfer learning method was reworked in order to be applied as a retraining method. The algorithm was still able to retain the benefits of its initial purpose, as shown in the good results from using poor synthetic data in a transductive transfer learning application. However, in all other application *TrAdaBoost* performed poorly. In failing to converge faster than the from-scratch method, *TrAdaBoost* showed itself unable to perform as a retrainer. Modifications could be made to the overall pipeline, and this could be explored if the final transductive and inductive pipelines are less accurate than these initial results.