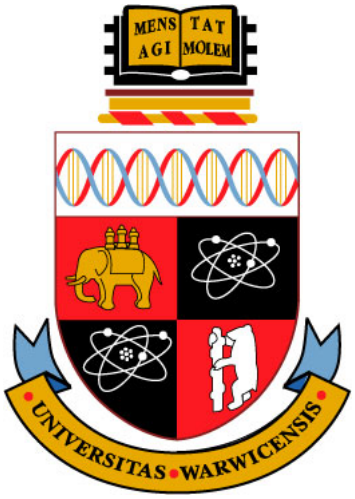


AI Bias in Facial Recognition

CS907 Dissertation Project

Jordan Chigbo

Supervisor: Dr. Claire Rocks

Department of Computer Science, University of Warwick

A dissertation submitted in partial fulfilment for the degree of

Master of Science in Computer Science.

September 2021

Abstract

Algorithms subtly structure our lives. They help ascertain eligibility for housing, the suitability of job candidates and even the length of prison sentences in criminal justice. Algorithms optimise systems for operational efficiency upon their application, but the validity of their decisions is largely ignored. As public and private institutions increasingly depend on algorithms, it is important for the biases in its decision-making to be considered. Researchers have found that leading facial recognition algorithms demonstrate race, gender and age biases. These algorithms have applications in surveillance and healthcare, highlighting its deep importance and relevance in society. At present, there is not a significant level of research on the algorithmic bias of facial identification for race, age and gender. This study seeks to develop an approach to minimise the bias by investigating numerous factors upon which algorithmic bias is dependent on. We add novelty to research on algorithmic bias by amalgamating an existing dataset with additional images to expand its racial representation. This dataset formed the basis of our investigations. We first trained VGG19, ResNet50 and AlexNet neural networks with the dataset, and then carried ResNet50 (the best performing network) forward to study the effect of dataset demographic distribution and representative image quality on model performance.

Through a systematic interpretation of the results, we ascertain that firstly, algorithmic bias is quite dependent on the facial recognition algorithm of choice. Furthermore, the bias is affected by subpopulation distributions within the dataset. We also found that gender prediction accuracy was not impacted as much as race prediction accuracy when the image quality was varied in blur, lighting and occlusion. We provide suitable explanations for all of the results and explore various avenues of developing the research results.

Keywords - machine learning, convolutional neural networks, facial recognition, artificial intelligence, algorithmic bias, race bias, public interest technology

Acknowledgements

Firstly, I give my thanks to The Most High for being the Rock throughout all instances in my life. I commit this work into Your hands.

As my time in formal education draws to a close, I acknowledge my family for their unwavering support throughout these foundational decades of my life. To my sisters Chikodi and Chiamaka, my mother Bolaji, and my father Uche, I give my utmost gratitude for empowering me to always try my best in my academic endeavours; from primary school until this very moment of writing. My personal growth during university, particularly over this MSc course, is simply a microcosm of the numerous journeys in life that I have gone through, am currently going through and am also yet to start. I am grateful to have had such a valuable support system throughout this journey. Your care and sincerity for my wellbeing knows zero bounds, and I am firmly confident that completing this chapter of my life is largely due to your combined efforts and investments in me. I strive to be a brother and son who can reciprocate just half of what you have each done for me in your individual capacities. Thank you.

Lastly, but by no means at least, I thank Dr. Claire Rocks for supervising me throughout the course of this project. The nature of this topic is very timely in today's society, and so I genuinely appreciate the opportunity that you provided me by submitting this as a project suggestion at the start of the academic year. Writing a dissertation project is certainly intense, but I have thoroughly enjoyed researching this topic over the past months as I was able to educate myself on, raise awareness for and contribute research to an important sociopolitical crisis via the technical medium that I understand best. Your support and guidance shared throughout this entire process was invaluable and so I extend my thanks.

Abbreviations

Artificial Intelligence	AI
Convolutional Neural Network	CNN
Cross-Race Effect	CRE
Deep Convolutional Neural Network	DCNN
Machine Learning	ML
Facial Recognition Technology	FRT
Black	B
White	W
South Asian	SA
East Asian	EA
Middle Eastern	ME
Hispanic/Latino	HL

Contents

List of Figures

List of Tables

1 Introduction

Education, finance, and surveillance, among other social interests, have long been viewed as necessarily involving manual, human-driven practices. However, with the advent and sustained application of artificial intelligence (AI), the utility of human involvement is subject for debate. Machine learning (ML) systems rely on analysing data in order to find insightful relationships through pattern recognition [17]. These pattern recognition algorithms form the basis of products and services which drive the course of modern society; ranging from Amazon's virtual personal assistant Alexa, to HireVue - an AI-driven video interview platform [26]. While these products strive to streamline business processes (and thus lower company overheads), they have a destructive tendency that cannot be measured through an readily tangible statistic such as financial loss. A socioeconomic lens confirms that for the most part, the wide-scale adoption of AI systems amplifies and perpetuates numerous existing forms of discrimination, including but not limited to: race, sex and age [17].

More than 50 years of research evidences that people are more accurate in recognising faces of their own race, than faces of other races. This phenomenon, more formally known as the Cross-Race Effect (CRE) [19], has since surfaced in multiple facial recognition algorithms. Researchers have found that leading facial recognition systems have different accuracy rates for different demographic groups. The first study to showcase this result was a 2003 report by the National Institute of Standards and Technology (NIST) [48]. NIST found that female subjects were more difficult for algorithms to recognise than male subjects, and younger subjects more difficult to recognise than older subjects. A 2018 collaborative research venture between MIT researcher Joy Buolamwini and Microsoft's Timnit Gebru made headlines with a report showing that gender classification algorithms (which are related, though distinct from race classification algorithms) had error rates of just 1% for white men, but around 35% for dark-skinned women [9]. A subsequent 2019 report by NIST confirmed these findings; they found the majority of commercially available algorithms to exhibit greater error rates for Asians, African Americans and Native Americans, compared to white individuals [22]. Furthermore, relatively higher accuracies were associated with men, and also with middle aged adults.

In spite of these dire findings, the reports made encouraging suggestions for de-biasing prevalent facial recognition algorithms. Methods of mitigation include developing more inclusive code prac-

tices by having a diverse team involved with the algorithm development process. They will be able to detect unconscious blind spots of others when making considerations for how to structure the algorithm. Furthermore, ensuring a fair representation of demographic groups in the algorithm training data will help to reduce algorithmic bias. From a more sociopolitical standpoint, it was submitted that lobbying organisations that use these algorithms should be accountable for promoting their destructive values and incentives, and so long as they are using these algorithms ultimately for financial gain, they should also commit to funding research into public interest technology. Facial recognition technologies have been employed in several sectors, such as healthcare [30], retail [5, 15], education [6] and transport [40, 38]. Having understood the technology's reverberating effects of historical racism and sexism, it is imperative to investigate the matter as its widespread adoption in society increases.

This paper embarks to develop an approach to measure, interpret and minimise the bias in facial recognition procedures by performing detailed experiments on a set of bias factors. **Chapter 2** will identify and critique articles of literature relevant to the project, in order to determine a understanding of the current state-of-the-art. **Chapter 3** will detail the design and methodology of the project's technical contribution. **Chapter 4** will feature a critical deep-dive into of the results obtained in the study. **Chapter 5** summarises and concludes the findings of the report, and **Chapter 6** will evaluate the project from a more holistic and operational perspective.

1.1 Objectives

The main objectives of this project are to:

- Investigate and understand the advantages and disadvantages of various machine learning techniques in the facial recognition domain.
- Gain a fundamental understanding of the process of recognising and predicting relevant attributes from faces in images.
- Construct a series of datasets that suitably represent the distributions of select cities across the world which are varied in race.
- Test the datasets on various facial recognition algorithms and measure their respective performances.

- Apply image processing transformations to the data to explore the effect of representative image quality on model performance.

2 Literature Review

To understand the current state of research in the domain of algorithmic bias in facial recognition methods, in this section we review and critique relevant scholarly pieces in order to synthesise a set of hypotheses that we can produce investigations around.

2.1 Algorithmic Bias of Neural Networks for Facial Recognition Methods

It is beneficial to separate the timeline of facial recognition algorithm development into two: before the adoption of deep convolutional neural networks (DCNNs), and after its adoption. This is because DCNNs changed the approach to facial recognition algorithms in a remarkable manner.

2.1.1 Pre-DCNN

Research into the differential accuracy of facial recognition algorithms across race originate back to the 1990s. One of the earliest studies to demonstrate race bias in algorithms predicated on an auto-associative network reported that learning-based computational models are heavily influenced by race [44]. The model was trained with White faces as the majority race, and Asian faces as the minority race, or vice-versa. O'Toole *et al.* discovered that the model was "better able to reconstruct unlearned majority faces than minority faces". They also found that the inter-face similarity was higher for the reconstructed minority faces than for reconstructed majority faces, indicating that the model was coding the majority faces more distinctively than the minority faces. This study found that the identification accuracy was greater for the race with which the model had a greater exposure to learning from.

This finding was corroborated in a 2004 probe by Givens *et al.* [20], who determined that covariates such as race differentially affected algorithm accuracy. A 2006 algorithm competition based on a dataset of faces of varying race groups, alongside varying factors such as face size and environment (indoors or outdoors) echoed the conclusion of Givens *et al.*: race, as a covariate, impacts algorithms in manners that are difficult to predict.

A 2011 investigation into the effects of bias in algorithms ascertained that the geographic origin of the algorithm is a factor in race bias [49]. A 'super'-algorithm comprised of 8 Western algorithms and the East Asian equivalent of 5 algorithms were scrutinised, and

both instances demonstrated a cross-race effect in facial identification. However, the reproducibility of this study is for discussion: this is the only academic study that shows that algorithmic origin has a statistically significant effect on race bias. Furthermore, the base algorithms tested in the study were 'black-box', meaning that the researchers were unaware of their internal implementations. It is possible that the training datasets for the trainable algorithms had varying racial compositions, which may have contributed to the race bias.

A noteworthy study by Klare *et al.* in 2011 [33] compared three public, readily-available algorithms to three in-house algorithms and judged the effect of demographic composition on the algorithms' respective accuracies. All algorithms demonstrated a significantly low identification accuracy associated with young, Black and female faces, compared to other demographic groups. Furthermore, the researchers found that using a race-balanced training dataset reduced the effects of specific demographic biases, however it did not completely remove them.

2.1.2 Post-DCNN

The introduction of CNN-based algorithms in 2014 brought about a significant acceleration in facial recognition algorithm development. Key to the relevance of DCNNs in this field, is an understanding the function of a neural network (NN) and a convolutional neural network (CNN).

The premise of an NN is inspired by neurons within the human brain; mimicking the way that they transmit signals. A NN breaks down an input into multiple layers of abstraction: an input layer, at least one intermediate, 'hidden' layer, and an output layer [59]. NNs are trained by being shown many examples of images or sounds, for example, to recognise embedded patterns - the same way the human brain does. Figure 2.1 illustrates a typical NN architecture.

The behaviour of the network is defined by how its individual nodes are connected, and by the strengths (weights) of those connections. During network training, the weights automatically adjust according to a given learning rule until the NN correctly performs the desired task. As well as pattern recognition, NNs are capable of other supervised learning techniques, such as classification and regression, alongside the unsupervised learning task of clustering.

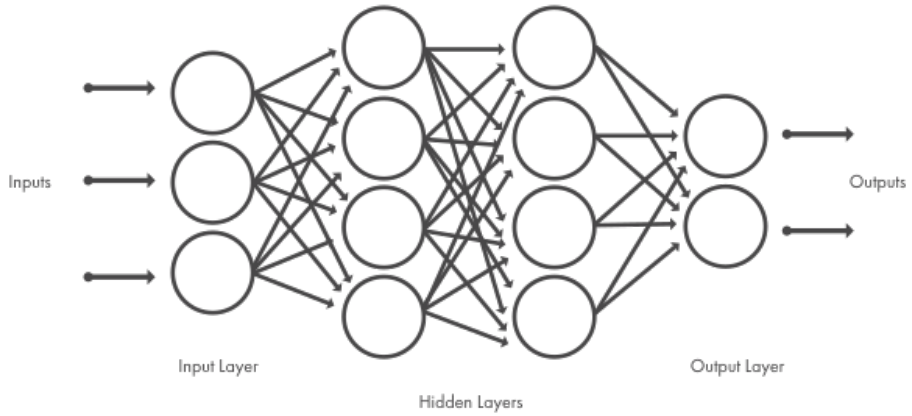


Figure 2.1: A typical neural network architecture [59].

2.1.3 DCNN Fundamentals

CNNs are a variant of the original neural network, dating back to 1982, with Fukushima and Miyake's creation of a self-organising model, for visual pattern recognition mechanism [18]. CNNs are an extension of NNs, which are classifiers. There is an additional feature learning stage, which carries forward image features detected by numerous convolution filters. The filter applies a sliding matrix of weights to an input matrix. At each convolutional layer, back-propagation is used to optimise the weights. After the convolution process, the resultant matrix is normalised and pooled in order to form a compressed representation of the image at the input layer of the network. After several convolution and pooling iterations, the traditional fully connected NN (i.e., a multi-layer perceptron) is brought about, where the classification process occurs. Figure 2.2 is an indicative example of a CNN architecture - distinctly noted are the feature learning and the classification stages. The difference between a CNN and DCNN is that DCNNs feature a larger amount of hidden layers than CNNs. Although, their makeups both follow the general convention of: convolution layer - pooling layer - fully-connected layer.

Now understanding the inner workings of CNNs, we can consider its usage in the realm of facial recognition algorithms. At present, race bias in CNNs trained for facial recognition has not been heavily investigated. A 2016 study by El Khiyari *et al.* [32] evaluated accuracy discrepancies across a commercial algorithm and a publicly available CNN, VGG-Face [47]. The demographic attributes under scrutiny were sex (male/female), age (young/middle-age/mature),

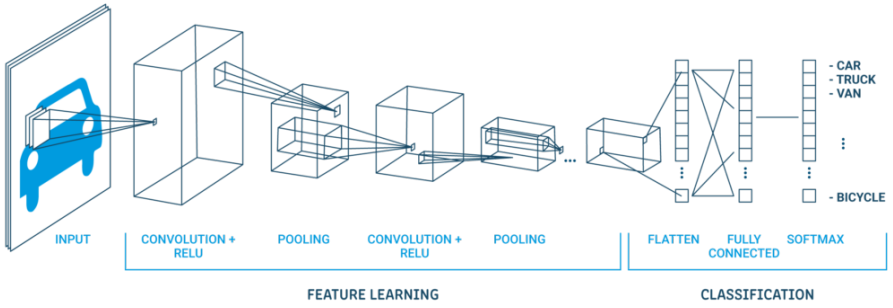


Figure 2.2: An indicative CNN architecture which detects the nature of a vehicle [53].

and race (white/Black). It was found that over all comparisons, the best accuracy was obtained for middle-age White males, and the poorest accuracy was associated with young, Black females.

In 2019, Cook *et al.* explored the effect of demographic variation on 11 undisclosed commercial facial recognition systems [13]. Each algorithm obtained images from 363 subjects across numerous demographic groups. The covariate results showed that skin reflectance had the most impact on performance. Lower (i.e. darker) skin reflectance was associated with longer classification wait times and lower similarity scores in same-identity distribution. In the same year, Howard *et al.* tried Black and white faces on a leading "commercial algorithm" [27], in which its false acceptance rates were far greater for white males than Black males. This is a finding contradictory to most existing research, however Howard *et al.* believe that the reason for these results arise from the wide age range in their training dataset.

Krishnapriya *et al.* compared accuracy for Black and White faces [54] across four algorithms: VGG-Face [47], a ResNet (residual network) CNN [25], and two commercial algorithms. It was found that the most recent commercial algorithm and the ResNet-based algorithm performed best on Black faces, scoring an accuracy of . Conversely, VGG-Face and the other commercial algorithm performed better on White faces. The paper also explored the effect of image quality on race bias. Krishnapriya *et al.* found that overall accuracy improved when the model was trained on International Civil Aviation Organisation (ICAO) compliant images. These results not only serve as evidence for race bias in newer CNNs, but provide insight into how image quality affects prediction accuracy. The latter finding will be explored later in this section.

Architecture	Year	Number of Layers	Test Accuracy [%] [28]
AlexNet	2012	8	84.6
VGG-16	2014	16	92.7
VGG-19	2014	19	92
GoogLeNet	2014	22	93.3
ResNet	2016	up to 152	96.4

Table 2.1: Summary of leading CNN-based neural networks designed for facial recognition.

Table 2.1 summarises the leading CNN-based architectures designed for a facial recognition application.

We see that model test accuracy improves over time, and so for the purpose of our report, an accuracy of 95% and upwards can be considered excellent. A model that scores accuracies below this threshold will not be regarded as such an excellent model.

From reviewing the timeline of race bias investigations facial recognition algorithms (pre and post-CNN), an overarching fact can be abstracted: differences in accuracy occur across race, gender and age groups. In some instances, algorithms have shown a human-like interaction between the origin of the algorithm and race of the face under test. This conclusion harks back to the Cross-Race Effect - we can verify that this phenomenon transcends human-to-human interaction, and is also evident in the products of humans, i.e. these algorithms.

2.2 Image Quality Considerations

The limited capacity to recognise faces in images with sub-par quality is a long-standing problem that still presents a challenge to humans, but more critically, machines purposed with this sole task of identification. Understandably, image quality deviations are commonplace in realistic environments, however its effects are not generally considered in model performance. Since most investigations are carried out with (pseudo)-ideal test variables, it is easy to overlook the feasibility of these studies in respect to the real world. Testing a facial recognition model on an image of a face with sufficient lighting, minimal blockage and high pixel density is very well for laboratory environments. However, images taken in a real-world setting will likely not embody such an ideal trinity of characteristics. Table 2.2 categorises occlusion challenges in different scenarios with their occlusion typical examples.

Literature exists on the exploration of recognition algorithms' performance in respect to synthetic occlusion. In 2018, Sárándi *et al.*

Occlusion Scenario	Examples
Facial accessories	eyeglasses, sunglasses, scarves, mask, hat, hair
External occlusions	occluded by hands and random objects
Limited field of view	partial faces
Self-occlusions	non-frontal pose
Extreme illumination	part of face highlighted
Artificial occlusions	random black rectangles random white rectangles random salt & pepper noise

Table 2.2: Categorisation of occlusion challenges [65].

investigated the robustness of 3D pose estimation to occlusion [58]. By applying geometric occlusions to numerous image examples, the authors succeeded in finding a method with "state-of-the-art benchmark performance" which were "sensitive even to low amounts of occlusion". This study was carried out in the context of realistic human-robot shared environments, where the autonomous robots' perception must be developed enough to ascertain smooth, safe and comfortable human-robot interaction. In the scope of this report, where AI is liaising with 2D images of human faces, it is also imperative that we assess the effects of image occlusion on algorithmic bias to ensure a reliable relationship between machine and maker.

A more recent study by Tsai *et al.* developed an approach to detect and recognise faces in an occlusion-resistant manner [61]. From a given input, the system calculated the face regions and facial landmarks (in a similar manner to dlib - see Section 3.1.2), aligned the face by the facial landmarks and then input the image into the facial recognition network for identification. The proposed system successfully improved the experimental recognition accuracy when the face was occluded. As well as occlusion, Tsai *et al.* determined that lighting and the distance between the camera and the subject's face were also critical factors in facial recognition accuracy. This study only considered facial recognition accuracy in the context of detecting emotion and pose. The classification of race, age or gender was not looked at. This lack of appreciation for demographic metrics throughout literature only supports the necessity of this investigation.

2.3 Evaluating Model Performance

It is worthwhile to review the generally accepted procedures for evaluating the performance of facial recognition methods, as it will aid in developing an effective testing and validation regime.

Accuracy is the simplest validation metric to both conceptualise and compute when estimating training algorithms. It is a statistical review of model performance that is directly dependent on the size and distribution of the dataset. Further to this, it is only useful when the training classes are balanced. Accuracy is a calculated by the number of true-positive (TP), true-negative (TN), false-positive (FP) and false-negative (FN) classified values.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{2.1}$$

Inspecting Equation 2.1, it is clear that accuracy is a scale-dependent metric; that is, in the case of extremely unbalanced classes (e.g. a binary set of 95% positively labelled and 5% negatively labelled data), the computed value of accuracy would be very misleading.

A better, more accepted alternative to a naïve computation of accuracy, particularly when dealing with imbalanced classes, is the F1-score. F1-score is characterised by

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{2.2}$$

This is a standard harmonic mean statistic, which is used as it penalises extreme values.

To visualise the classification performance, a leading option is the receiving operating characteristic (ROC) curve. The AUC (area under the curve) represents the usefulness of the task. An area of 1 represents a perfect test, and an area of 0.5 represents a worthless test. It is analogous to a fair coin flip. The ROC curve is a plot of the True Positive Rate (TPR) against the FPR (False Positive Rate). Equations 2.3 and 2.4 show these respective formulae.

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{2.3}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{2.4}$$

ROC curves are advantageous because they allow for an adequate comparison over a range of different class distributions. In the context of face prediction; for race, the TPR and FPR will be computed in a 'one-vs-all' fashion. A positive would represent a correctly predicted race, whereas a negative would represent an incorrect race prediction. This reduces the multi-class issue to a binary classification. As this investigation considers sex as either male or female, a 'one-vs-all' approach for sex is not necessary. A standard binary classifier shall suffice.

Race and gender are classification tasks, however the task of estimating age falls under regression. Commonly used performance metrics in the realm of regression are mean average error (MAE) and mean standard error (MSE). These errors are respectively calculated as:

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}, \tag{2.5}$$

$$\text{MSE} = \frac{\sum_{i=1}^{n} (y_i - x_i)^2}{n}, \tag{2.6}$$

where x is the predicted value, y is the true value and n is the number of observations. MAE is a quantity used to measure how close predictions are to the outcomes. It is an average of all the absolute errors of the observations. MSE however, is a function equivalent to the expected value of the squared error loss. This difference occurs due to the randomness of the age prediction - it can either be above or below the true value. MSE is good as a measurement of the quality of an estimator because it includes both the variance of the estimator and also its bias.

2.4 Algorithmic Bias Factors

Having reviewed the existing literature in this field, we gain an understanding of the technical issues regarding facial identification for differing races, ages and genders. From this, we can synthesise a set of factors which the bias is dependent on:

- **Algorithms**: the effect of the facial recognition method of choice on the model accuracy.
- **Sub-population distributions**: how well the model fares under training datasets with varying compositions of demographic distributions.
- **Representative images**: the effect of photo quality and lighting conditions on the accuracy of the model.

In the upcoming sections of this paper, these factors will be explored in further detail.

3 Design

The following section explores the conception of, and rationalisations behind the various technical elements developed for this project.

3.1 Dataset

3.1.1 Selection

A number of public datasets were assessed for potential use in the research. Notable datasets included FFHQ [31], Pilots Parliament Benchmark (PPB) [35], UTKFace [66] and PubFig [34]. Table 3.1 summarises the key characteristics of the aforementioned datasets.

Dataset	**Source**	**Size**	**In-the-Wild?**	**Annotations**
FFHQ	Flickr [16]	70,000	Yes	None
PPB	Government Official Profiles	1,300	No	Age and Gender. Skin colour is computed as a race proxy
PubFig	Public Figures	59,000	Yes	Celebrity Name
UTKFace	CACD [10]	28,000	Yes	Age, Race and Gender

Table 3.1: Summary of various publicly available datasets

Critically analysing the characteristics of each dataset yielded the following conclusions:

- FFHQ contains 70,000 images, all of which are unannotated. Given the time constraints in project delivery, manually labelling these many images would certainly be a costly procedure. Given a larger budget/amount of time, leveraging an outsourcing tool such as Amazon Mechanical Turk [1] may prove effective in labelling this grandiose dataset.

- While PPB provides age and gender annotations, the dataset had skin colour generated as a means to avoid race computation, as the authors posit that race is an unstable and volatile social construct. Because we define a standard set of race

groups in this paper, the skin colour classifications provided in the dataset are somewhat obsolete. As well as this, it is the only dataset with images that are not in-the-wild, meaning that the images do not satisfy the context of this research: a 'wild', real-world environment where facial recognition surveillance systems are typically used. The images in this dataset are akin to those on official identification documents such as a passport. Furthermore, the image subjects (government officials) are all likely to be in their adult age, resulting in a lack of representation of youth and elderly people in the dataset.

- PubFig is in-the wild, however it has no annotations useful to this research. This gives rise to the same issue that FFHQ faces. Additionally, the 59,000 images in PubFig are of 200 people, who are all celebrities. Again, for the context of this research, this is an insufficiently representative data pool.
- This leaves UTKFace - with 28,000 samples, an in-the-wild nature and age, race and gender annotations, the dataset proves promising for further consideration.

3.1.2 Preparation

UTKFace has a fairly varying set of race annotations (Black, White, Asian, Indian and Others), however there are opportunities for improvement. The PPB authors' argument of race as an unstable construct is relevant, and thus inspired the following modifications:

- Renamed 'Indian' to 'South Asian': South Asia encompasses several ethnic groups such as those in India, Bangladesh and Sri Lanka among others, and as the groups look less distinct, this renaming offers a more inclusive taxonomy.
- Renamed 'Asian' to 'East Asian': similarly, East Asian countries (as well as multiple countries in the Indochinese region) have fairly similar facial structures and appearances, and so the decision was made to amalgamate the two areas under the East Asian category.
- Divided 'Others' into 'Arab' and 'Latino/Hispanic': The decision was made to accommodate these race groups because of the necessity to fairly represent all groups, which is an important feat in the wider context of this project. Furthermore, when tailoring the demographic compositions of the dataset to reflect that of other relatively diverse municipalities, Latino

and Hispanic representation may be more in abundance. This holds true for the Arab race group also.

During the division of 'Others', it was found that the majority of the faces were classified as Latino/Hispanic, therefore a small amount of Arab faces were present in the dataset. To account for this weak representation, Flickr was scraped for photos of "arab man", "arab woman", "arab teenage boy", "arab teenage girl", "arab boy" and "arab girl". These discriminatory searches were deliberately made in order to gain a wider array of ages across the photos, in efforts to sustain a wide age representation. Since these images were sourced outside of UTKFace, each image was manually cropped to a bounding box of the person's face, and their ages were manually estimated. This age estimation inherently incorporated a degree of bias into the dataset since due to the subjectivity associated with age estimation.

To improve upon this strategy, multiple people could estimate the age of an image and reach a consensus. As well as this, because the facial bounding boxes were obtained manually, a slight bias is introduced. UTKFace makes use of dlib [14], a computer vision package, to align and crop the images. Thus, images cropped by hand would be slightly algorithmically obtained results. Figure 3.1 shows the variations in bounding boxes across dlib and OpenCV, another leading computer vision package. On visual inspection, the difference in bounding box area does not seem too problematic, however, as eventual training data, these extra pixels can affect the predictive power of the model because more pixels not pertaining to the face are included in the image. However, this is permissible due to the in-the-wild nature of the UTKFace dataset.

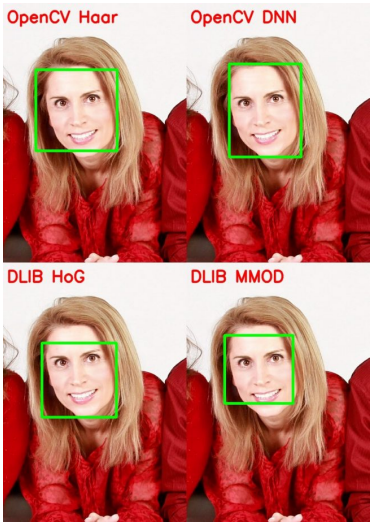


Figure 3.1: Bounding boxes generated by OpenCV and dlib [23].

To balance the dataset in race, all race group sizes were trimmed to the size of the smallest category (Arab), reducing the size to 13,266. This is still a suitably sized dataset - 2111 examples per race offers acceptable representation and confident results.

3.1.3 Composition Variants

Alongside the original UTKFace dataset and its balanced variant, 3 further variants were formed. Ethnically diverse metropolises across the world were scrutinised for their race group composition, and the following cities were thus selected:

- **London, UK**: the 2011 United Kingdom census data [50] was polled to provide a demographic breakdown by race. As the census' race groups slightly diverge from the framework defined in this paper, the decision was made to consider "any other ethnic group" as Hispanic/Latino, because it is the only race group not covered by this paper's defined groups.
- **New York City, USA**: The US 2019 census data [62] was amalgamated with specific Arab estimation data from the Arab America Society [4] to form a composition reflecting the populous capital of New York state.
- **Houston, USA**: The US 2019 census [62] also provided a breakdown of Houston's population by ethnicity.

Table 3.2 tabulates the dataset variants' compositions in race.

Dataset Variant	Race Frequency					
	Black	**White**	**S. Asian**	**E. Asian**	**Hispanic/Latino**	**Arab**
Original (UTKFace)	4526	10078	3975	3434	1692	
Balanced	2211	2211	2211	2211	2211	2211
London	423	1306	265	143	46	28
New York	285	1025	148	359	376	18
Houston	560	566	17	133	950	16

Table 3.2: Dataset variant race compositions. Note: For the 'Original (UTKFace)' row, S. Asian, E. Asian, (Hispanic/Latino + Arab) respectively correspond to Indian, Asian and Others, in the original UTKFace dataset.

3.1.4 Image Quality Variants

Because this project is being carried out in the context of public surveillance, editing the dataset images may provide interesting insights into facial recognition technology usage in public. The effect of real-world, noisy training data on the model's predictive power will be investigated. This will be done by adding image blur and occlusion in varying intensities to mimic outdoor conditions such as poor weather, sub-par lighting and/or congested streets. The effects of lighting via gamma correction will also be explored. Table 3.3 tabulates the image effects and the associated intensities under consideration in this report.

Image Effect	**Intensity Variants**		
Blur	10	20	30
Occlusion	25%	50%	75%
Lighting (Gamma Correction)	0.25	0.5	2

Table 3.3: Effects and its associated intensities to be applied to the images. Note: for the blur variants, let each value be regarded as n, which will be used as a $n \times n$ box convolution blur filter to be applied to the image.

It was decided to use 25% occlusion increments due to the nature of the dataset images. The images are of bare faces - none of the subjects are wearing covering accessories. As UTKFace uses dlib to align and crop the images, we can safely consider each image to be centred about the subject's nose. Hence, we take that as our origin to create outward geometric occlusions from. An example of this is given in 4.14. To sift through each image in the dataset and manually superimpose a common real-world accessory such as sunglasses or a scarf would be an extremely lengthy procedure. All of the faces are unique and have different face shapes, and so they'd need to be placed very carefully, as opposed to using an algorithm to do it. Furthermore, using a human to superimpose the occlusion objects incorporates a bias because the placement of the object is down to the occluder's discretion. As well as this, from an operational per-

spective, this method was not possible under the time constraints of the project.

3.2 Tools, Libraries and Frameworks

Python was selected as the development language of choice in this project. This is largely due to the maturity and reduced learning curve of its machine learning packages scikit-learn and keras. These packages, alongside others in the Python scientific computing stack such as NumPy [41] and pandas [46] are often used in a professional capability, which only promotes its stability and feasibility. MATLAB, a popular contender, was not used because its deep learning toolbox comes at an additional cost.

The code was written in Visual Studio Code, a leading IDE which offers support for IPython (an architecture for interactive Python development), as well as integrated SSH capabilities to remotely work on the DCS machines.

The nature of this research is computationally intensive, therefore one may consider using a language that has the capability to manipulate low-level memory such as C++. However, the learning curve associated with the language is high. A benefit of C++ is that it does have direct access to TensorFlow - the backend upon which keras is built. C++ has a 'zero-overhead principle' [7], whereby "what you do use, you couldn't hand code any better" and so developing a model in its native language would reduce the associated overheads of calling the functions, whereas as its Python-wrapped version, keras, would run slower due to its need to make calls to the TensorFlow backend. As future work, a C++ implementation of the project may be created in order to utilise TensorFlow at a native level. It would also be ideal to leverage MPI - a parallel programming interface [37]. This would result in significantly faster model runtimes.

Various additional elements were used in developing the code:

- **CUDA**: a parallel computing platform for general computing on GPUs [3].
- **Conda**: a Python environment handler, to simplify package management and deployment [42].

The program was run on the GPU batch within the Department of Computer Science at the University of Warwick - the specifications of which are in the appendix. Slurm, a Linux job scheduler commonly used for computing clusters [55], was used in order to

track and schedule different instances of the running code. When using the batch compute system, completing one epoch during the training stage takes approximately 10 minutes, whereas if using the standard DCS servers, the epoch takes almost triple the time.

3.3 System Implementation

3.3.1 Solution Requirements

Table 3.4 outlines the formal requirements of the system.

Requirement	**Priority**
Resize raw images into a new dimension (e.g. 224×224)	M
Train specified neural network on a given training set	M
Modify the training set to fit a given proportion	M
Test specified neural network on a given test set	M
Apply occlusion of a given amount to a given test instance	M
Apply blurring of a given amount to a given test instance	M
Apply gamma correction of a given intensity to a given test instance	M
Handle valid or corrupt test images	D
Easily integrate with an external system to be used in real-world practice	D
Evaluate algorithmic bias by accuracy and error	M
Evaluate algorithmic bias with an extended set of metrics and plots:	D
Design code with OOP principles	D

Table 3.4: FaceNet solution requirements. Note: M = Mandatory, D = Desirable, O = Optional

These requirements outline the metric of success of the technical contribution in this project. These requirements will be revisited during the project evaluation stage.

3.3.2 Codebase

Figure 3.2 is an illustration of a high-level system overview.

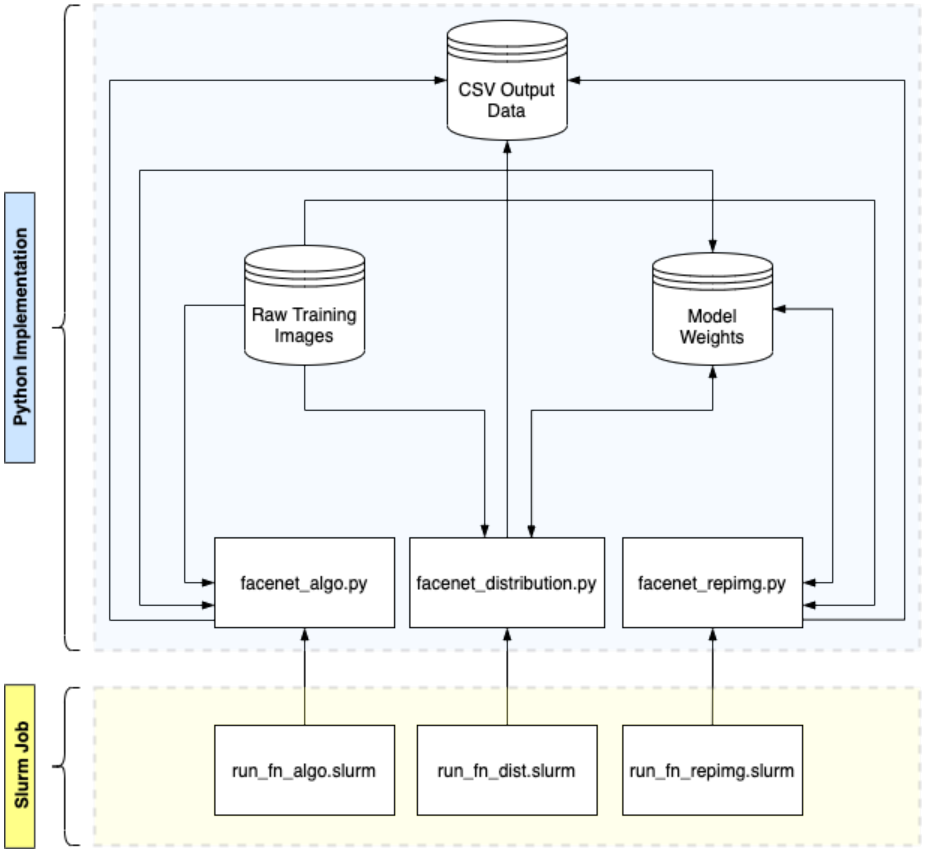


Figure 3.2: FaceNet high-level system design diagram.

The system was designed in three sections; roughly following the same structure as the Discussion section. A FaceNet Python script was made for each investigation. All three FaceNet Python files have similar code infrastructure, however there are slight tweaks depending on the investigation variable of choice. Figure 3.3 shows the program flowchart. The processes exclusive to each respective Python script are highlighted in yellow.

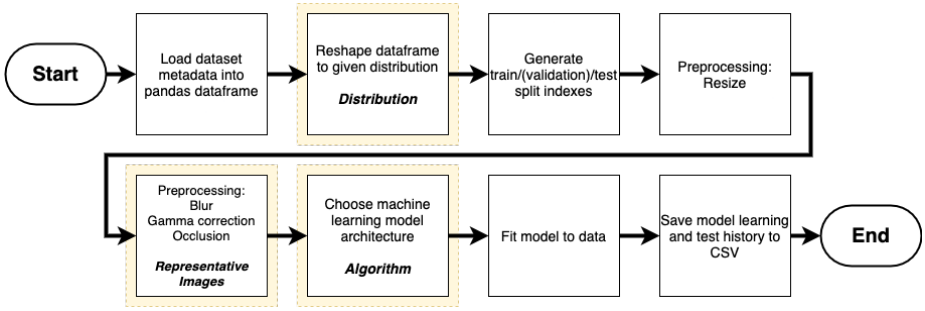


Figure 3.3: Flowchart for the Python scripts.

facenet_algo.py: There is a *network* variable which is set to either 'alexnet', 'vgg19' or 'resnet50'. Dependent on the value of *network*, the program returns the appropriate model to then be compiled and have data fitted to it. Before fitting the data, the program checks whether a Hierarchical Data Format (.h5) file containing the model architecture and trained weight values exists. If it does, this file is simply loaded and returned, saving the ≈90 minute time required to train and test the model. Otherwise, the model is trained and tested in real-time. At an average file size of 50 MB, storing model information in .h5 files is a suitably cheap alternative to TensorFlow's `StoredModel` object which typically produces larger files. This logic is also used in the other Python scripts.

facenet_dist.py: The Balanced, London, New York and Houston demographic proportions (see Table 3.2) were stored as arrays, and formed the values in a dictionary, with the cities as the associated keys. The *train_dist* and *test_dist* variable values were switched between 'balanced', 'london', 'newyork' and 'houston'. The system checks which train and test distributions are selected, and then resizes the respective race counts to the required amounts. Where the train and test distributions are the same (i.e. *train_dist* == *test_dist*), the program cleverly splits the training data into train, validation and test sets without any duplicate instances.

facenet_repimg.py: The preprocessing stage contained logic for transforming the image to a desired setting fit for consumption by the machine learning model. OpenCV, a library of computer vision tools [43], was used in order to perform the blurring, gamma correction and occlusion operations.

The mode of blurring used was a standard box blur. The box blur is a uniformly weighted convolution in which each pixel in the resulting image has a value equal to the average of its neighbouring pixels in the input image [2]. For indicative purposes, a box blur of kernel size $(3, 3)$ can be written as:

$$\frac{1}{3^2}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}. \tag{3.1}$$

Fortunately, the tedious process of manually computing this convolved value for each pixel is not necessary, as OpenCV provides a `cv2.blur(image, kernel_size)` function which encapsulates this logic.

Gamma correction is however, a more manual process. Gamma correction, more formally known as the Power Law Transform, maps

a narrow range of dark input values into a wider range of output values, with the opposite being true for higher input values [60]. This is characterised in Equation 3.2:

$$O = I^{\frac{1}{\gamma}}, \tag{3.2}$$

where O and I are the output and input pixel RGB values, and γ is the gamma constant. γ values below 1 darken the image, and conversely, γ values above 1 lighten the image. Before the image is passed into the transform, all pixel RGB values in the range [0, 255] are scaled to [0.0, 1.0]. The normalised values are stored in a NumPy array. This serves as a lookup table, which is applied to the input image in order to output the gamma corrected image. The time complexity of an entry lookup in a lookup (hash)-table is $\mathcal{O}(1)$, meaning that the gamma correction procedure is optimally fast.

The occlusion logic required simple geometric calculations. Figure 3.4 is a sketch of the measurements used to calculate the area of occlusion.

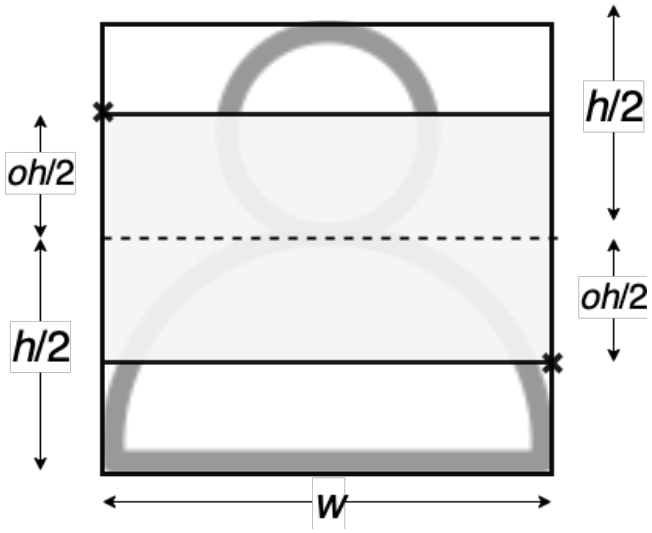


Figure 3.4: Image occlusion sketch, with height h, width w and occlusion factor o.

The occlusion factor, a value in the range [0, 1], is used to determine the diagonal coordinates of the occlusion rectangle. The coordinates are expressed by $(0, \frac{h}{2} - \frac{oh}{2})$, $(w, \frac{h}{2} + \frac{oh}{2})$. These are passed into `cv2.rectangle()`, which superimposes a filled rectangle on top of the image.

There are three equivalent IPython notebooks (`plotter_algo.ipynb`, `plotter_dist.ipynb`, and `plotter_repimg.ipynb`) which were used

to produce the plots seen across this report. Pandas conveniently has a function to swiftly load CSV files into dataframes. Matplotlib's state-based interface, pyplot [36], was used to perform data visualisation.

3.4 Model

3.4.1 Networks

To determine an ideal network design to begin with, three leading DCNN architectures were considered: AlexNet, VGG-Face and ResNet50. However, for context, it is first worthwhile to gain an understanding of some network elements generally found within DCNNs.

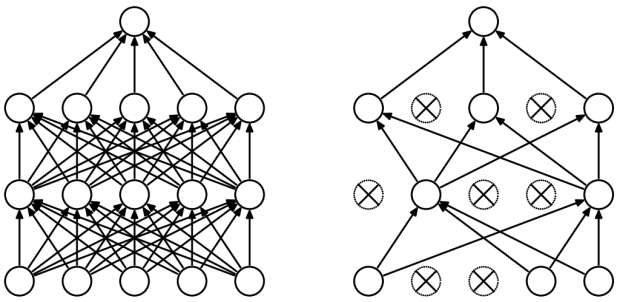
- **Rectified Linear Unit (ReLU)**: ReLU is a piecewise linear function that directly outputs the input if it is positive, and zero if the input is negative [8]. This is characterised by Equation 3.3:

$$\text{ReLU}(x) = \begin{cases} x & \text{for } x > 0, \\ 0 & \text{for } x \leq 0 \end{cases} \tag{3.3}$$

  ReLU is regarded as the default activation function when developing most types of neural networks. The function is straightforward to implement (simply requiring a max() function), boasting computational simplicity compared to alternative activation functions such as $\tanh(x)$ and $\text{sigmoid}(x)$ - both of which require exponential calculations [21]. Furthermore, the function mostly exhibits linear behaviour (for $x > 0$) which is ideal. In general, a neural network is easier to optimise when its behaviour is linear, or close to linear [21]. The linear behaviour of the activation function also eliminates the problem of vanishing gradients, as it does not have an error that can be back-propagated through the network, unlike the aforementioned exponential calculations.

- **Dropout**: DCNNs with a large number of parameters prove to be very powerful systems, however they are prone to overfitting. Also, these large systems are slow to use, making it hard to deal with overfitting by combining the predictions of many large neural nets at the same time. Dropout is a technique for seeing to this problem. The premise of the dropout layer is to randomly drop nodes (and their respective connections) from

the neural network during the training stage [56]. This stops the nodes from co-adapting too much. During training, the dropout layer samples from an exponential number of different "thinned" networks. Figure 3.5 shows the an indicative neural network before and after dropout is applied to it. Srivastava *et al.* found that it is easy to approximate the effect of averaging the predictions of all these thinned networks, by using a single unthinned network that has smaller weights [56]. This greatly reduces model overfitting and yields major improvements over alternatives such as L1 and L2 regularisation.



(a) Standard Neural Net (b) After applying dropout.

Figure 3.5: Standard neural network; before (a) and after (b) applying dropout [56].

- **Batch Normalisation**: this is a data pre-processing tool used to bring the data to a common scale without distorting its shape. This allows the model to generalise appropriately, as data input into a machine may have varying sizes. Batch normalisation extends this concept; the layer performs the standardising and normalising operations on a batch input of a layer, coming from a previous layer. Processing the data in batches makes neural networks faster and more stable. Furthermore, it solves the problem of internal covariate shift, ensuring that the input for each layer is distributed about the same mean and standard deviation [29].
- **Max Pooling**: Max pooling performs sample-based discretisation. It reduces the dimensionality of an input by allowing for assumptions to be made about features contained in the binned sub-regions. The size of the sampling sub-region is dictated by the stride, typically 2×2. This is illustrated in Figure 3.6. For each sub-region, the maximum value is taken, and then they are passed into a new output matrix where each element is the max of the corresponding region in the original input. This reduces

the network's overfitting ability, as well as the computational cost by reducing the number of parameters to learn. Other pooling variants include general pooling and average pooling. Max pooling is the most commonly used type, because it picks up the most prominent features, and so most of the features are retained in the resultant image. Furthermore, yet minor, it provides a relative saving on computational cost by simply finding the maximum, and not needing to compute averages of several sub-regions.

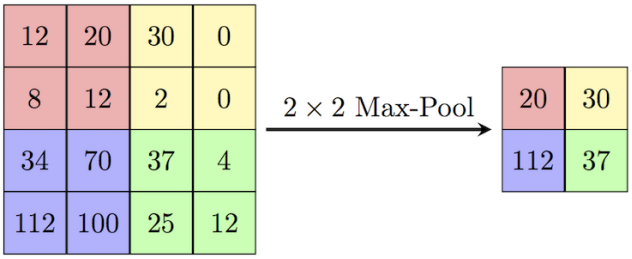


Figure 3.6: Pictoral representation of a max pooling layer with stride (2, 2) [12].

AlexNet

AlexNet is an architecture designed to classify images into 1000 different classes, as an entry for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012. The neural network, designed by Alex Krizhevsky, won the competition with a top-5 error rate of 15.3% - almost twice as better than the 2nd place's rate of 26.3%. AlexNet contains eight learned layers, of which five are convolutional layers, two are fully-connected layers and one is a fully-connected 1000-wide softmax layer. The network also features a local normalisation strategy, which allows for generalisation. The architecture contains overlapping pooling layers to reduce computation and control overfitting.

ResNet50

The victor of the 2015 ILSVRC was ResNet50 - a residual deep learning neural network model with 50 layers. DCNNs are typically hard to train because of the vanishing gradient problem. Repeated multiplication makes for a negligible gradient, causing the model to stop learning. Residual networks solve the problem with skip connection. This allows an input to bypass layers and instead flow through an identity function, $\mathbf{x}$, to preserve the gradient [24].

ResNet50 features 48 convolution layers, 1 max pooling layer and 1 average pooling layer.

VGG19

VGG19 is a variant of the classical CNN architecture, VGG. Very similar to AlexNet, it only has $(3, 3)$ size convolutions, but many filters. VGG19 also has a convolution stride size of $(1, 1)$, to retain the whole context of the image under analysis. However, this does lend to its larger computational requirements. VGG19 has 16 convolution filters, and features max pooling and softmax activation.

The key features of the three CNN architectures are summarised in Table 3.5.

Architecture	**Trainable Parameters (in millions)**	**Number of Layers**	**Floating Point Operations (in billions)**
AlexNet	60	8	0.72
ResNet50	25.6	50	3.8
VGG19	23	19	19.6

Table 3.5: Key characteristics of the AlexNet, ResNet50 and VGG19 architectures.

3.4.2 Output Branches

As the model is multi-output (race, age and gender), three output branches are required. Since different attributes are being predicted, it is necessary for the model to have additional layers which funnel the data into an appropriate format, dependent on the desired prediction metric. To accustom the network architectures to the output (of race, age or gender), a final set of functions were added to each branch:

Race: Dense layer of dimension 6, followed by a ReLU activation layer. This was then followed by categorical cross-entropy - the loss function chosen for the race output branch. During the training phase, the model aims to minimise:

$$\text{Categorical Cross-Entropy Loss} == -\sum_{i=1}^{n} y_i \cdot \log(\hat{y}_i), \qquad (3.4)$$

where $\hat{y}_i$ is the i-th scalar value in the model output, y_i is the corresponding target value, and n is the number of scalar values in

the model output. Categorical cross-entropy is a good measure of how distinguishable two *discrete* probability distributions are from each other. The negative sign ensures that the loss reduces as the distributions become closer to each other.

Age: Dense layer of dimension 1, followed by a linear activation layer, and finally, a mean squared error loss function, as seen in Equation 2.6.

Gender: Dense layer of dimension 2, followed by a ReLU activation layer, and a binary cross-entropy loss function. Equation 3.5 is the formula for this function:

$$\text{Binary Cross-Entropy Loss} = -\frac{1}{n}\sum_{i=1}^{n} y_i \cdot \log(y_i) \cdot \log(1 - \hat{y}_i) \quad (3.5)$$

Formally, this loss is equivalent to the average result of the categorical cross-entropy loss function applied on two-category tasks with target probabilities y_i and $(1 - y_i)$.

4 Discussion

This chapter examines the quantitative results of our three investigations, and seeks to draw valuable insights that are considered from the perspective of public facial recognition. First, we put each of the algorithms under test in order to find the best performer. This will be used as the algorithm for performing the subsequent investigations into sub-population distributions and representative images.

4.1 Algorithm

The three models under test were evaluated with an 80/20 train-test split of the Balanced dataset variant. Adam was the optimiser of choice, with a batch size of 128.

4.1.1 Model Loss

When training models, it is useful to run additional pieces of logic continually during training. Keras models conveniently accept callback functions; that is, functions that run at the start and/or end of every training and test epoch. In this investigation, the `EarlyStopping` callback was used to terminate model training once the loss had stopped increasing. To account for any anomalous behaviour between the training epochs, a tolerance of 2 was set. This means that if a model loss improvement is not detected for 2 epochs, the model training will then terminate. `EarlyStopping` is not only advantageous because it protects against model overfitting, but because it also reduces the number of training epochs required, leading to faster training times.

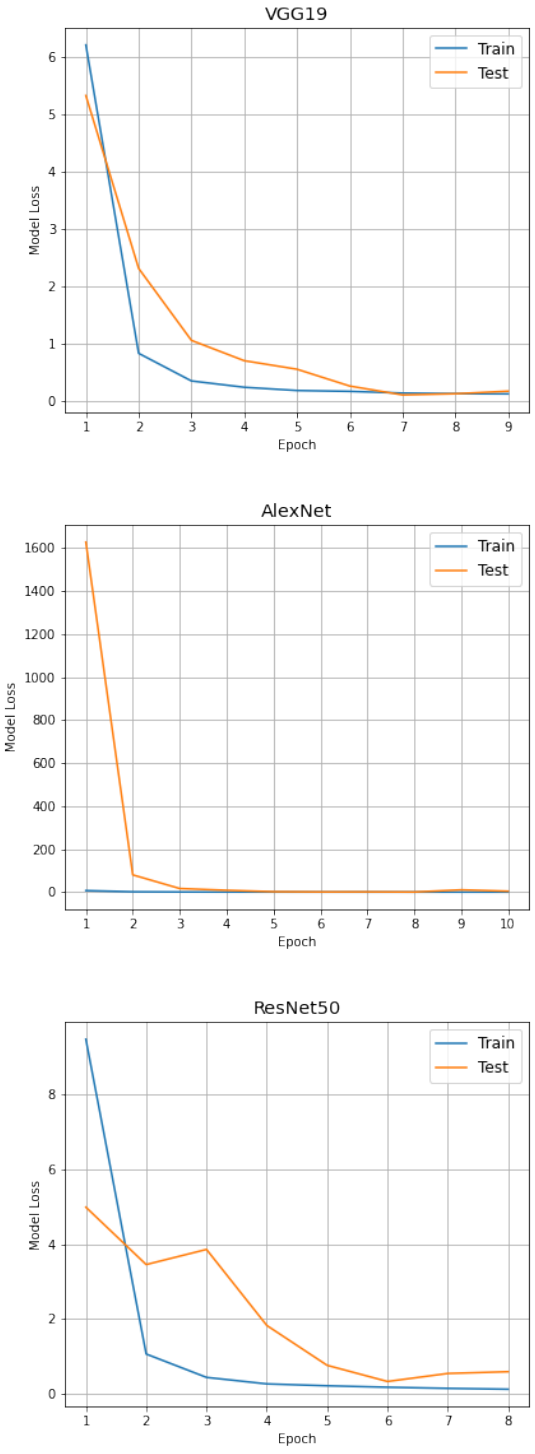


Figure 4.1: Training and test loss plots of the three architectures.

Figure 4.1 shows the model loss plots for the three architectures under scrutiny.

For the VGG19 model, the test data loss shares the same sentiment as the training loss - both respectively starting at 8.244 and 9.293, decreasing and reaching convergence between epochs 6 and 8. The effect of setting the `EarlyStopping` patience can be seen between epochs 2 and 4; the loss increases from epoch 2 to 3, but does not increase between 3 and 4. If the patience was not set at all, the model would have finished training at epoch 3, yielding a loss of 3.78. Observing the rest of the graph, it is clear that the model improved over the rest of the epochs, with the test data loss culminating at 0.35. This brings about a loss improvement of about 1000%, compared to if `EarlyStopping` was not used at all. It would be useful to experiment with different values of patience to assess the possibility of obtaining an even lower loss. However, from epoch 6, the train and test curves become relatively flat, and at a similar loss. This suggests that with a higher patience value, and thus larger number of epochs, the model is at risk of overfitting.

In the AlexNet architecture, the model converges very quickly. The large spike at epoch 1 is likely due to the gradient descent optimisation algorithm being used. This erratic behaviour is exhibited in Adam, which has an exponential moving average used in the algorithms. Reddi *et al.* discovered that algorithms with a stochastic nature, such as Adam (alongside other optimisers such as RMSProp and Adadelta) tend to display these erratic characteristics [51]. However, this is not too problematic and is likely just an anomalous instance, as the initial loss of 1982 is quickly neutralised during epoch 2, where the loss reduces fifteen-fold to 129.4, and then to 11.14 in epoch 3. The curve then begins to mimic the training loss curve, only diverging at most 1.66 units across epochs 4 and 10.

ResNet50 however, performs in a seemingly erratic manner. Although the initial test loss is not so large like in AlexNet and instead reflects the behaviour of VGG19, it is clear that the validation process is fairly unstable. Plus, this corroborates the idea that the first epoch of AlexNet was anomalous. The training curve appears sensible, and the test curve does follow its general trend, however, compared to the other two architectures, it is relatively unstable. A possible reason for this is the disagreement between the residual nature of the architecture, and the gradient descent optimisation method. To mitigate this effect, momentum may be applied to the gradient descent optimisation algorithm. The momentum extension allows the optimisation search to build inertia in the direction of the search space and overcome the oscillations of noisy gradients, thus

not getting 'stuck' in local minima, and also not falling stationary on local maxima. Between epochs 7 and 10, the test loss appears to converge and reach a minimum of 0.2135. However, it then shoots up at epoch 11 to 1.788, and then to 2.322 in the subsequent epoch. Setting a higher `EarlyStopping` patience would not help in this situation, as it is clear from the training loss curve that the loss should not need to rise again.

4.1.2 Race Output Accuracy

Figure 4.2 displays the race accuracy plots for the three networks.

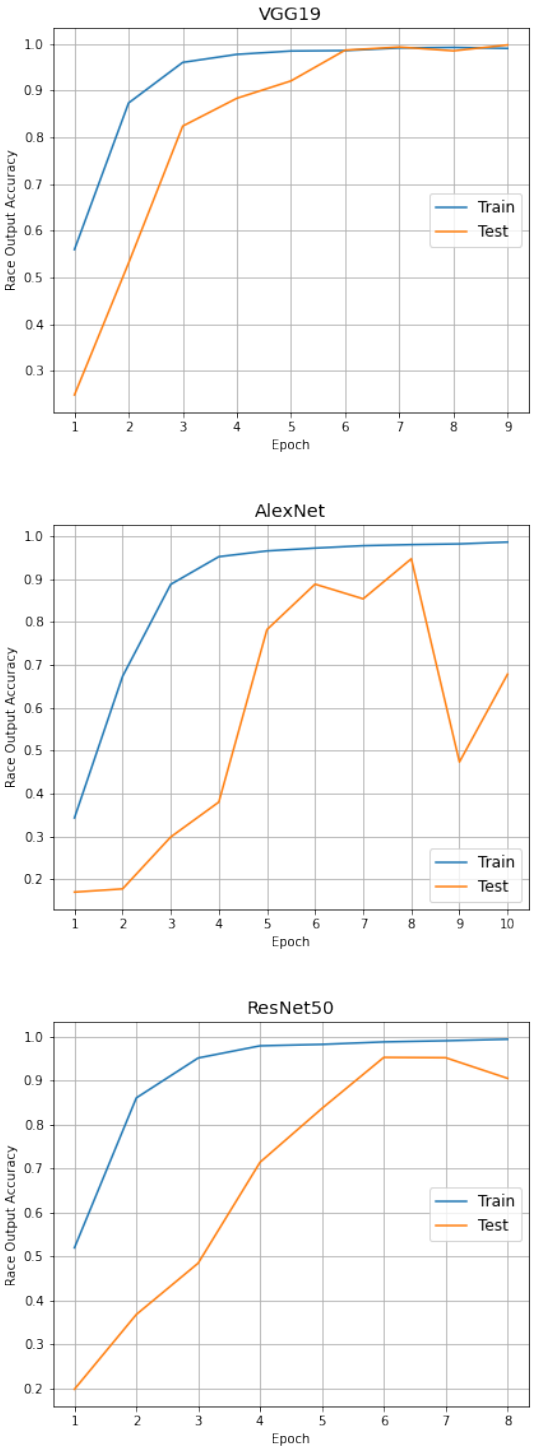


Figure 4.2: Race accuracy plots for the three architectures.

For VGG19, the test curve followed the general trajectory of the train curve. This indicates a good level of data fitting. The first reported test accuracy was 0.6541. It is worthy to note that this is the only point during the model evaluation at which the test accuracy was greater than the train accuracy. This is likely due to the randomness of the train/test data, leading to a 'lottery' effect in which the images seen were of the races seen in abundance during the model training stage. However, as proven in the literature review, accuracy is not the ideal metric when dealing with imbalanced classes. This steadily increased over the next 3 epochs, to 0.9286. From there, the accuracy begun to plateau. Between epochs 4 and 7, there were slight fluctuations (of magnitudes no more than 0.03293), before reaching epoch 8, where the test curve finally agreed with the train curve at 0.9873. As race accuracy was not the metric under `Early Stopping` scrutinisation, the gradient polarity of the test curve did not determine whether the model should stop or not. As mentioned, it was instead the model loss metric. Because of this, the race accuracy obtained may not be optimal. 0.9873 is still a remarkably good accuracy in the scope of this examination, however, in a real-world context where such technology would be applied to millions of people, the 1.27% misclass rate can amount to an exceedingly large amount of faces being incorrectly classified in race.

In the AlexNet architecture, the spike behaviour seen in the model loss curve is also seen here. Epoch 1 details a race accuracy of 0.6701, which subsequently falls to 0.6195 in epoch 2. As mentioned, this spike is likely the effect of the gradient descent algorithm under use. This anomalous result is quickly rectified however, as the test curve begins to follow the direction of the train curve. The test curve monotonically increases between epoch 2 and 7, and then fluctuates slightly while at its final convergence stage. Between epoch 8 and 9, the train accuracy slightly decreased from 0.9813 to 0.9749. Due to the sensitivity of the machine learning process, it was more than likely a fluctuation. If there were more epoch datapoints to be observed, it could be determined whether the race accuracy was actually going to continue decreasing. However, this is unlikely, seeing that it is the training curve. The final reported test accuracy was 0.9954 - at a misclass rate of 0.46%, this is more than twice as better as the misclass observed in the VGG19 evaluation.

In a similar fashion to the model loss behaviour, ResNet50 exhibits difficulty to smoothly converge. By epoch 3, the train and test curves had already diverged 84% relative to the accuracy difference in the first epoch. Although this is quickly recitifed in epoch 4,

straight after that the accuracy decreases. Epoch 5 to 6 is then has an ever-so-slight increase. As the `EarlyStopping` metric is model loss and not race accuracy, this is not alarming since it is not critical to the operation of the model. However, this type of behaviour would suggest a convergence, and so if race accuracy was the metric of choice, the patience would definitely have to be increased. The model finally stabilises at 0.9871. Although, the smoothness of the stabilisation was to a lesser extent than that in AlexNet, but it was not as erratic as the VGG19 convergence.

4.1.3 Gender Output Accuracy

Figure 4.3 is a series of gender accuracy plots for each model.

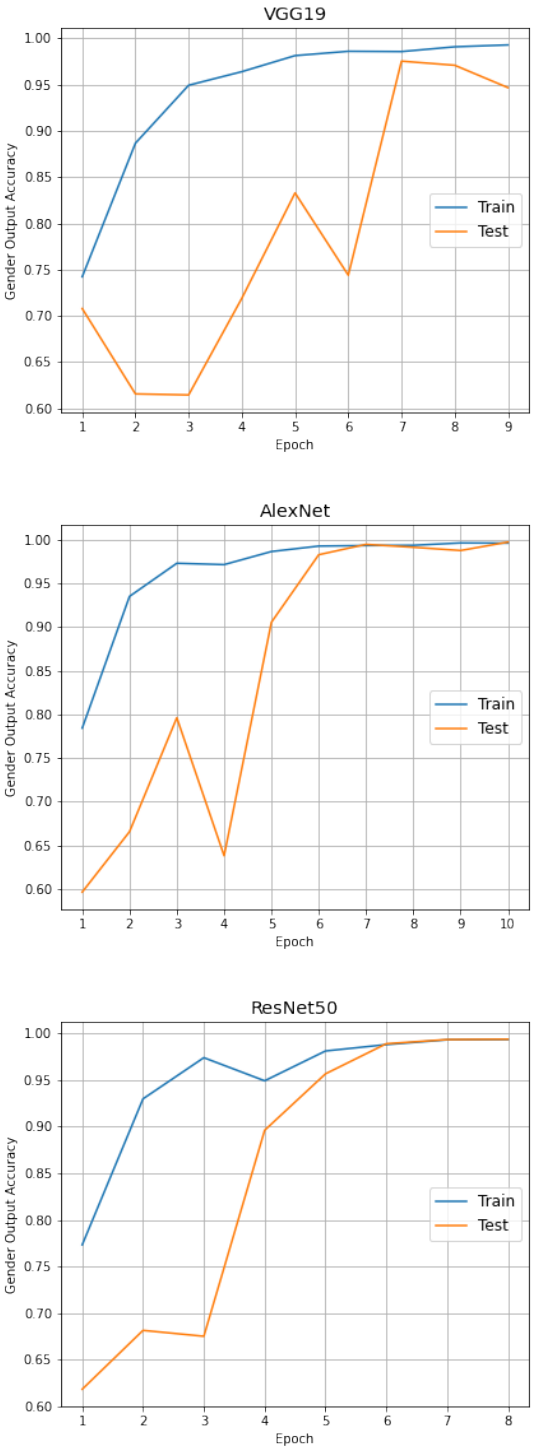


Figure 4.3: Gender accuracy plots for the three architectures.

For VGG19, the test gender output accuracy begins at 0.7080. It instantly decreases to 0.6157 by epoch 2, and then virtually remains the same at the following epoch; only differing by 0.001116. The output accuracy then sharply increases over the next two epochs to 0.8430. This is then met by a decline of magnitude 0.09900, to 0.7440. Another increase is observed between epochs 6 and 7, where the model appears to plateau. Although this plateau is negative (observing epochs 7 to 9), the low magnitude of its gradient suggests that the model is converging. The final reported value of the test gender output accuracy for VGG19 was 0.9468. It is a -4.6% delta from the final train gender output accuracy of 0.9927.

The AlexNet curve also has an intermediate peak during the testing phase. Beginning slightly lower (compared to VGG19), the initial gender output accuracy of 0.5964 rises to 0.7961 by epoch 3. This then drops to 0.6380, and subsequently inclines steeply to 0.9055 at epoch 5. From epoch 6 onwards, the model appears to reach convergence as the subsequent output accuracy values do not differ more than 0.45%. Also, at these latter epochs, the train and test curves are in agreement with each other, suggesting suitable data fitting. The final gender output accuracy was 0.9974.

A dip is noticeable between epochs 3 and 4 in the ResNet50 train curve. This is an interesting observation, seeing as the training curves in the other two architectures increase monotonically. This might be due to a random error; regardless of which, it is quickly resolved as the train curve converges to a final value of 0.9938. The same can be said for the test curve; only dipping slightly between epochs 2 and 3 before converging at a very similar gender output accuracy value of 0.9940. Rounded to 3 s.f., the final train and test gender output accuracy values are identical - indicating a stellar balance between bias and variance, which is ideal.

4.1.4 Age Error

Figure 4.4 contains plots of the age error curve.

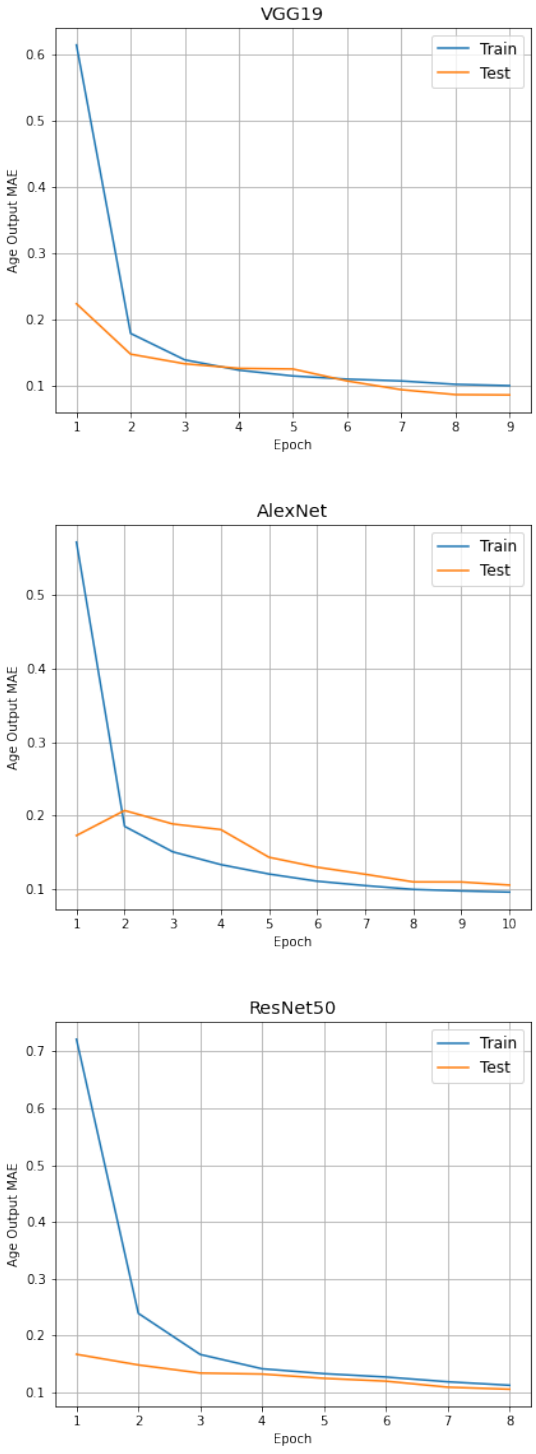


Figure 4.4: Age mean average error curves for the three architectures.

4.1.5 Conclusions

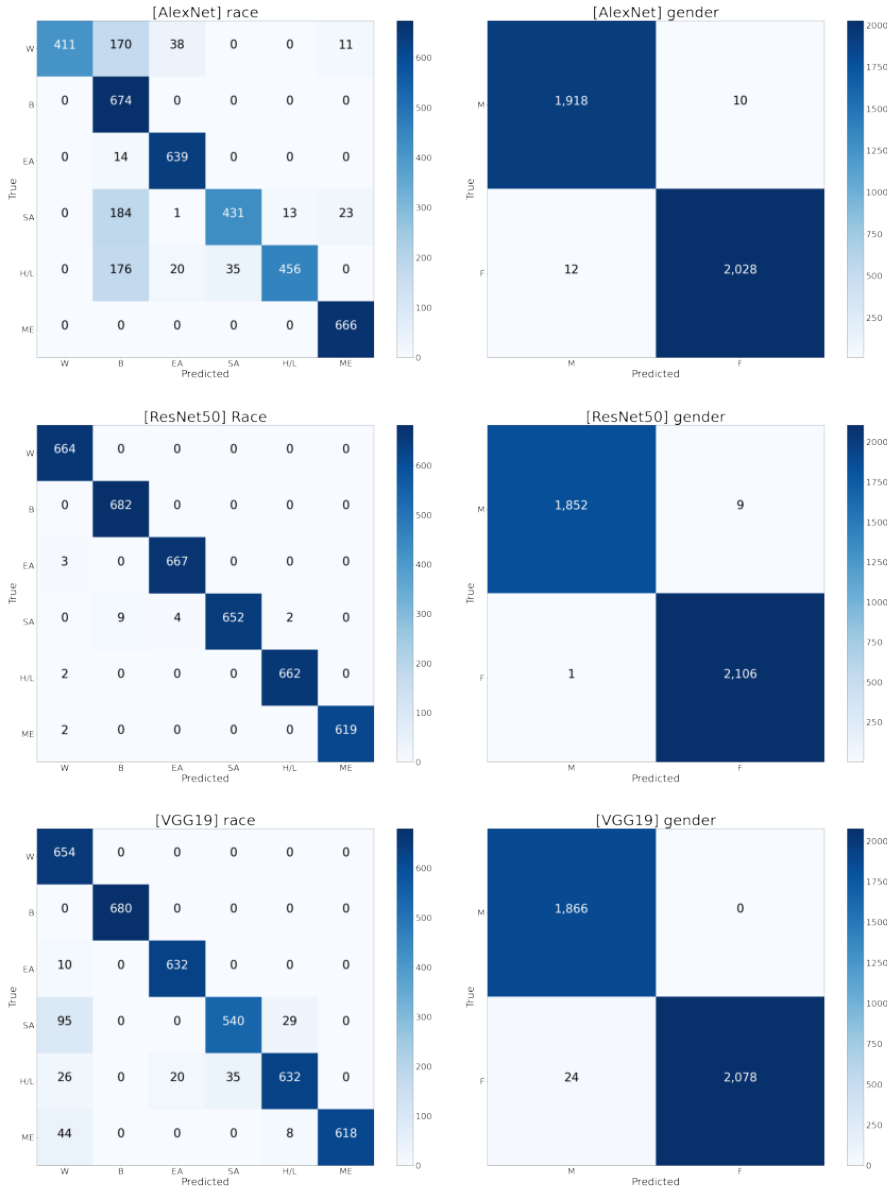


Figure 4.5: Confusion matrices of the three CNNs for race and gender.

The predictive capabilities of the each model across race and gender are depicted in the confusion matrices in Figure 4.5. Observing the colour intensity of the confusion matrices (where a darker blue represents a higher intensity), it is clear that ResNet50 performs the best in race and gender classification.

CNN Architecture	Output Accuracy	
	Race	Gender
AlexNet	0.8271	0.9945
ResNet50	**0.9945**	**0.9975**
VGG19	0.9336	0.9940

Table 4.1: Race and gender classification test accuracies.

In the race classification process, AlexNet performs significantly worse than the other two architectures. Therefore, to better the classification accuracy, it may be worth investigating various ways of tweaking the architecture, as well performing hyperparameter optimisation. The model incorrectly predicts 184 South Asian instances, 176 Hispanic/Latino instances and 170 White instances as Black. This harks back to the PPB data authors' idea that race is an unstable construct and why they opted to use skin colour as a proxy to race. It is plausible that the number of incorrect (Black) classified faces decreases as skin complexion tends to white, since they are on opposite ends of the spectrum colour-wise. Perhaps this is also due to the in-the-wild nature of the dataset images - if the images were ICAO-compliant, or at least had less background noise in general, then the machine learning model would be able to better focus on races' distinct facial structures which are typically more exclusive for a respective race, as opposed to naïvely evaluating the colour of a subject's skin. The effects of representative image quality on model performance will be discussed further in this section.

Table 4.1 tabulates the race and gender test output performance. Across the architectures, there is a large race output accuracy disparity of 16.74%. This is understandable as there is a wider set of classes for the model to select between, whereas all instances of gender output accuracies are extremely high - all at over 99%. However, the table shows the unequivocal truth that ResNet50 is the best performing model as it obtains both the highest race gender output accuracies. Predictably, this model would have the best performance out of the three architectures as it has a more rigorous learning procedure. For this reason, as well as its exceptional performance across-the-board, the architecture will be used as the algorithm of choice in the upcoming investigations.

4.2 Sub-population Distributions

In this subsection, we examine the effect of varying proportion of dataset sub-populations on numerous performance metrics.

4.2.1 Model Loss

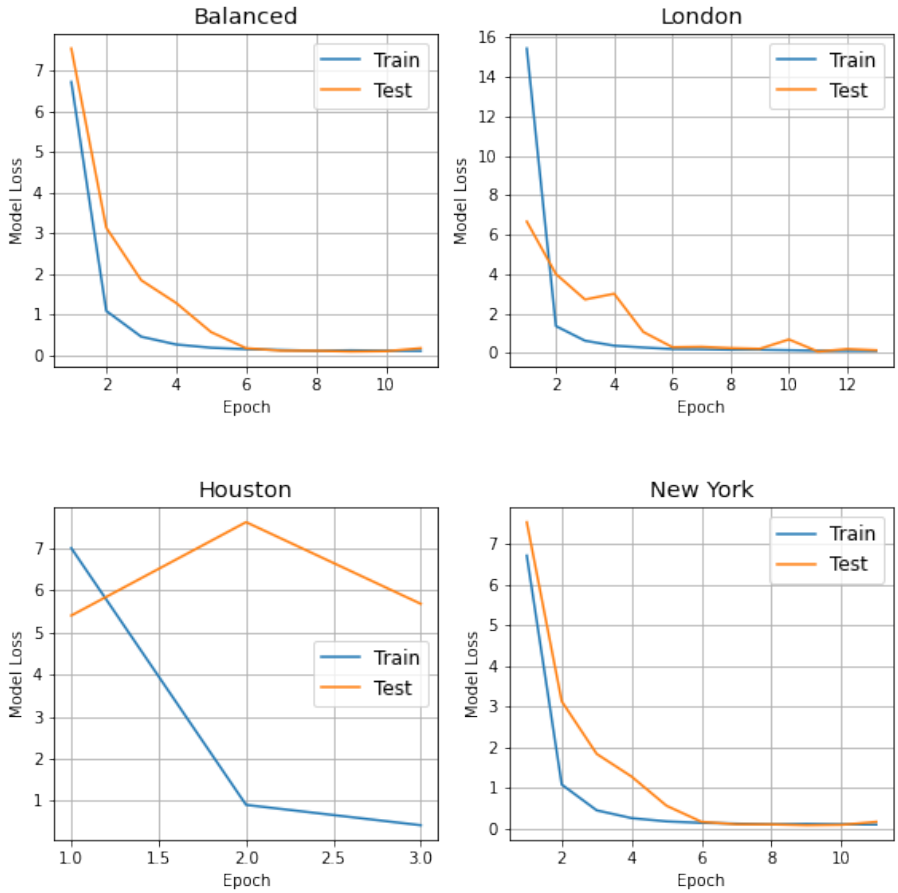


Figure 4.6: Model loss plots for the four distributions.

Figure 4.6 shows the model loss plots for the four dataset variants under scrutiny.

The Balanced variant has the most sensible performance out of the four. Both the train and test loss begin at respectable magnitudes of 6.372 and 7.023. By the net epoch, the test loss then decreases by over a factor of 2, to 3.205. This continues to decrease monotonically until between epochs 8 and 9, where there is a slight increase of 0.003. Our minimum `EarlyStopping` delta is set to 0.001, and therefore the increase was big enough to not necessitate the model to consider this as the (pen)-ultimate epoch. The trend then resumed in the expected course, finishing at a loss of 0.1039. This test loss is fine, especially considering that the final training loss is 0.90981 - a 7.76% decrease.

The same behaviour holds true for the New York dataset variant, exhibiting a very similar test and train performance to that of the Balanced dataset. The first recorded train and test losses are again in agreement with each other, differing only 10%. The test loss continues to decrease over the majority of the epochs. It is interesting to note that between epochs 6 and 10, where the train and test plots by eye almost inseparable, the same phenomenon occurs in the Balanced and New York model loss plots. Other than the White and Arab race groups, the races in the New York variant are spread in a somewhat similar fashion (relative to the other datasets). The race group deviating most from the mean (having excused the White and Arab outliers) is South Asian, at 148 instances, or a percentage difference of 7%. Since the dataset is more balanced than unbalanced, there are less unseen races in the dataset, meaning that during testing there will be less misclassification penalties, and thus a lower loss.

To a lesser extent, the London variant follows this train-test pattern. The initial train loss is higher than the other variants, at 15.44. This is more than double of all of the other variants' initial train losses. This sharply reduces to 1.359 in the next epoch however, and then continues to decrease and perform in a similar fashion to the two aforementioned variants. The London variant also had generally similar performance between epochs 2 and 6. This is where, area-wise, there is the greatest delta between the train and test curves. The training stage performs in an expected manner, with no erratic, cross-epoch spikes. However, the test plot had trouble reaching equillibrium in an equally smooth manner. Starting at a loss of 6.661, the test loss did definitely converge but it had spikes between epochs 3 - 4, and 9 - 11. If there was another increase between epochs 11 and 12, the model would have terminated due to `EarlyStopping` and resulted in a subpar loss result. Again, this demonstrates the utility of setting the callback's patience argument to 2.

The poorest performing composition was the Houston variant. The performance was so erratic that the training and test periods were only 3 epochs long. Although there are such few datapoints (the other variants have 3 - 4 times more), the general trend of the train curve is sensible. It begins at 7.008, and then decreases over seven-fold to 0.8994 by epoch 2. At epoch 3, it decreases 39.5% to 0.4153. In an absolute sense, 0.4153 is a good score; especially only 3 epochs into training. At this same point, the Balanced, London and New York variants had respective loss scores of 0.3664, 0.6163 and 0.4515. Predictably, the Balanced variant would have the lowest

loss, but Houston having the second-best loss at this point does appear promising. However, the issue is that this behaviour does not transcend into the test stage. The first recorded test loss is 5.404. Although the lowest initial test loss out of all four variants, this subsequently increases to 7.627. This already indicates that the test stage is unstable. The loss does decrease in epoch 3, however at this point the model was discarded because of how poorly it performed relative to the other datasets. For comparison, the final recorded losses for each variant is illustrated in Figure 4.7.

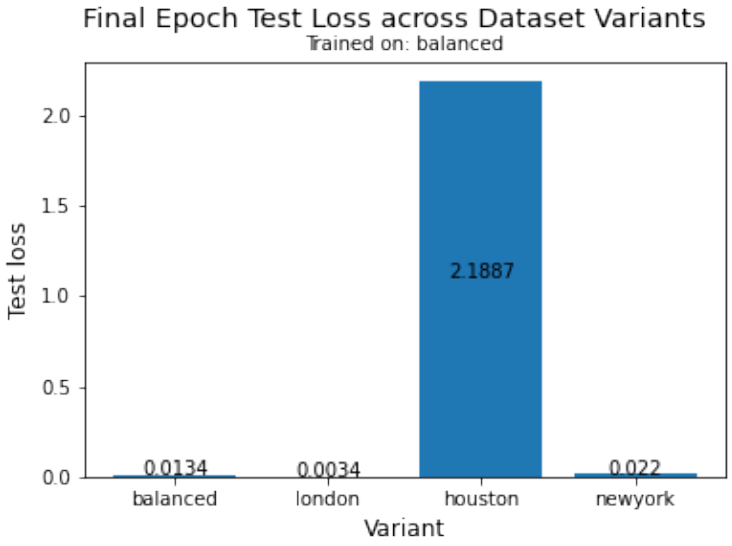


Figure 4.7: Final model loss for the four variants.

4.2.2 Race Output Accuracy

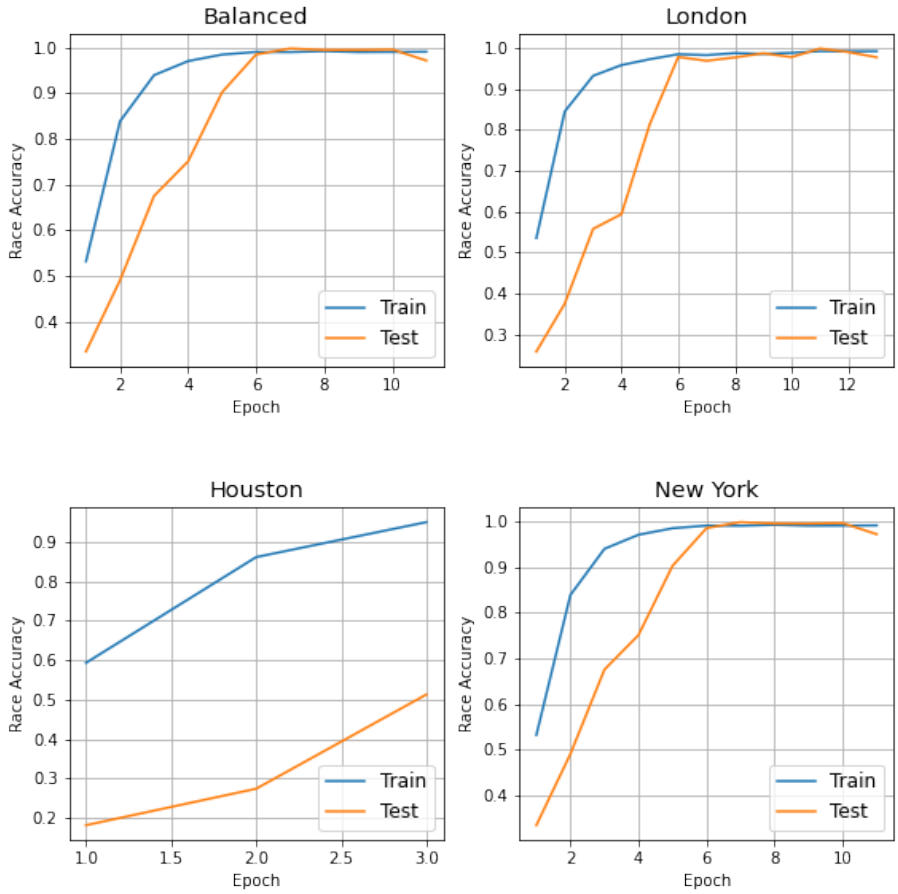


Figure 4.8: Race accuracy curves of the four dataset variants.

Figure 4.8 shows the race accuracy train-test curves for the four dataset variants.

Again, the Balanced, London and New York variants appear similar. Beginning with the Balanced variant; the first recorded test accuracy is 0.3281. The accuracy then rises to 0.4658 in epoch 2. The accuracy between epochs 3 and 4 still fits a positive trend, albeit at a lower gradient. This can also be observed in the test curves of the London and New York variants. This common behaviour allows for the assumption that this is an effect of the model architecture, ResNet50. The positive trend still continues until epoch 6, where it begins to plateau. Minor incremental increases are still observed though, rising 6% from 0.9606 to 0.9952 from epoch 6 to 10. However, past this, there are sustained decreases in the subsequent

epochs which led to the model terminating at a final accuracy of 0.9892. This is still a very respectable score.

As mentioned, the London variant exhibits a similar behaviour. Starting at 0.2574, the race accuracy increases sharply until epoch 6. From this stage onwards however, there is slightly more discrepancy between the train and test curves, compared to the curves in the Balanced dataset. There is an observable dip of 0.009301 between epochs 7 to 8. The model continues to strive for convergence, which it generally achieves albeit the slight variation. An accuracy decline can be seen in epochs 11 and 12. As there was a decrease in accuracy across two contiguous epochs, the model was terminated early in line with the stopping callback. The final recorded race accuracy was 0.977679.

The New York variant starts with an accuracy of 0.3348, and then doubles to 0.6752 in just two epochs. The characteristic lower, yet still positive gradient again occurs between epochs 3 and 4, and then the model trends towards convergence from epochs 6 onwards. Again, there is expectedly slight variations of around 0.001 across these epochs. The characteristic tail-end accuracy decline seen in the Balanced and London plots is also visible in the New York plot. The final two epochs see a decrease of 0.002, leading to an ending accuracy of 0.9714.

The Houston variant again has the worst race accuracy between all of the variants. The training curve does follow the trend as those of the other variants, starting at an initial accuracy of 0.5300, where the other variants' respective accuracies were 0.5522 (Balanced), 0.5353 (London) and 0.5322 (New York). The test accuracy is predictably where results diverge. The first recorded test accuracy was 0.1808, where the next closest competitor was the London dataset with 0.25. The accuracy increased to 0.2734 in the next epoch, and then finally 0.5126.

Figure 4.9 is a bar plot of the final race output accuracies of the variants under scrutiny.

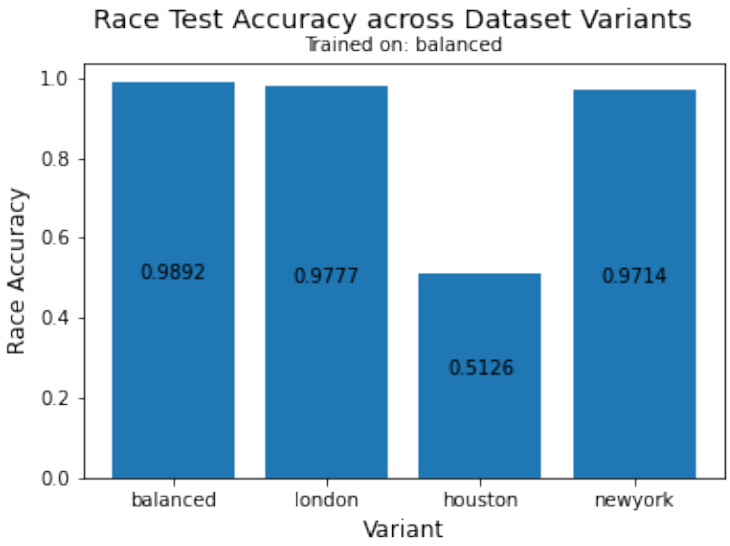


Figure 4.9: Final race accuracy for the four variants.

All instances were trained on the Balanced variant for linearity as this is the benchmark. For the sole case of where the Balanced variant was tested on itself, a train/test/validation split of 80/10/10 was used. All other variants had an 80/20 train/test split. Figure 4.9 shows that the three variants outside Houston achieved a stellar accuracy, corroborating the ideas posed earlier. Houston's final accuracy of 0.5126 is very poor both in an absolute sense, and in comparison to the other accuracies. This would almost certainly be discarded as an option in a real-world implementation.

4.2.3 Gender Output Accuracy

Figure 4.10 shows the gender output train-test accuracy curves of the four variants.

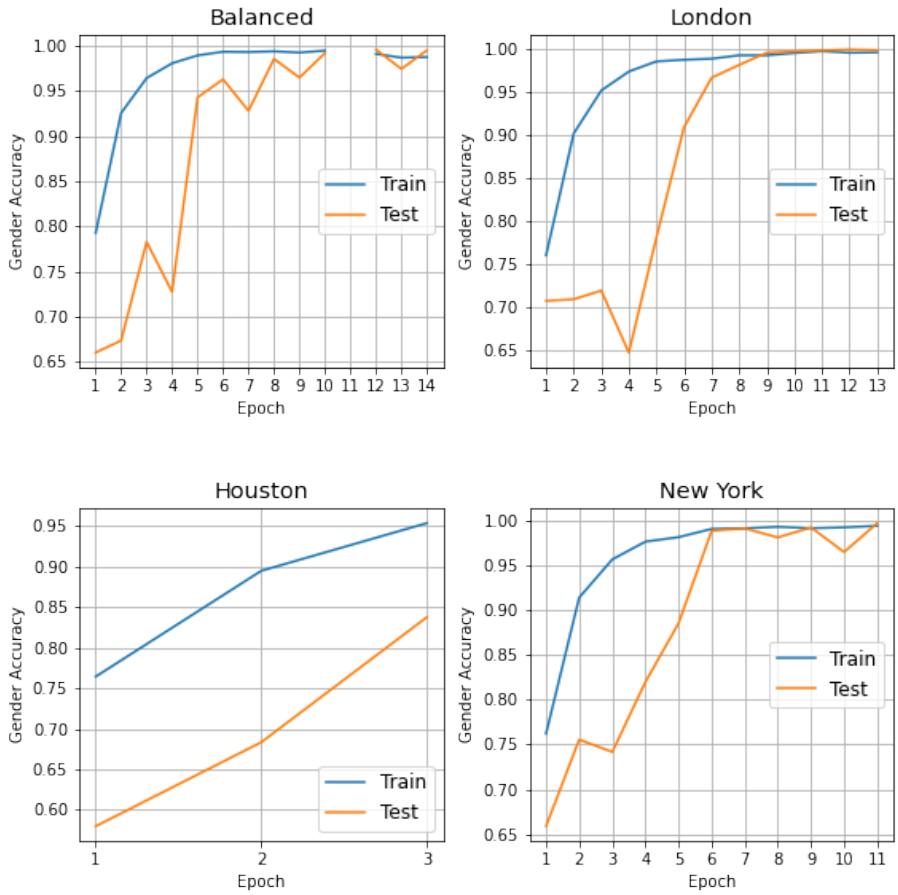


Figure 4.10: Gender accuracy curves of the four dataset variants.

For the Balanced dataset, there is evidently a bit of difficulty for the test accuracy to smoothly converge. The accuracy at epoch 1 is 0.6600, and it rises to 0.6734 in the subsequent epoch. However, the first dip occurs between epoch 3 and epoch 4, where the accuracy drops from 0.7827 to 0.7273; a 7.1% decrease. This decrease is subsequently met by a sharp upwards rebound in epoch 5, where the accuracy is 0.9431. It then increases slightly to 0.9632 before being met by another slight decrease of 3.5%. It is promising that the magnitude of the declines are decreasing as the process continues. This can also be seen between epochs 12 and 13 - the drop from 0.9963 to 0.9747 represents an even smaller decrease of 2.1%. Therefore, although the gender output accuracy does not converge as smoothly as some of the other metrics, it does still perform respectably. The final recorded gender output accuracy was 0.9952.

The London variant seemingly reaches convergence in a smoother

manner than the Balanced variant. The test accuracy begins at 0.7072 and slightly increases over the next two epochs to 0.7191. This is then followed by a sharp decrease to 0.6473. However, after this the gender output accuracy only monotonically increases. From epoch 9, with an accuracy of 0.9962 , the test curve begins to agree with the training curve - only differing a maximum of 0.0013 at epoch 12. The final gender output accuracy for the London variant was 0.9978.

Ranking by stability, the New York dataset can broadly fit in the middle of the two aforementioned variants. It does not converge as smoothly as the London variant, but it is relatively less chaotic than the Balanced dataset. The gender output accuracy begins at 0.6589 and increases to 0.7556 at epoch 2. A slight dip to 0.7418 is observed in epoch 3. An interesting observation is that in the Balanced and London variants, the first dip was one instance 'later' - the decline was instead between epochs 3 and 4. The accuracy continues to rise sharply until epoch 6 and epoch 7 where there is a slight plateau. The accuracies here are 0.9889 and 0.9914 respectively. This is then met by a dip to 0.9814, but rebounds in epoch 9 to 0.9926. The final dip then occurs, sending the accuracy down to 0.965. However, this is negated in the last epoch which yielded an accuracy of 0.9974.

Again, the Houston dataset does not have many datapoints to draw a strong conclusion from. Over the three epochs, the test accuracy rose from 0.5800 to 0.8374. The other variants' test accuracies at this epoch were 0.7827 (Balanced), 0.7191 (London) and 0.7418 (New York). So, it does have the greatest accuracy at that point, but all other models continue to increase in accuracy and ultimately perform better than the Houston variant. Figure 4.11 shows the final gender output accuracies for each variant, all trained on the Balanced dataset.

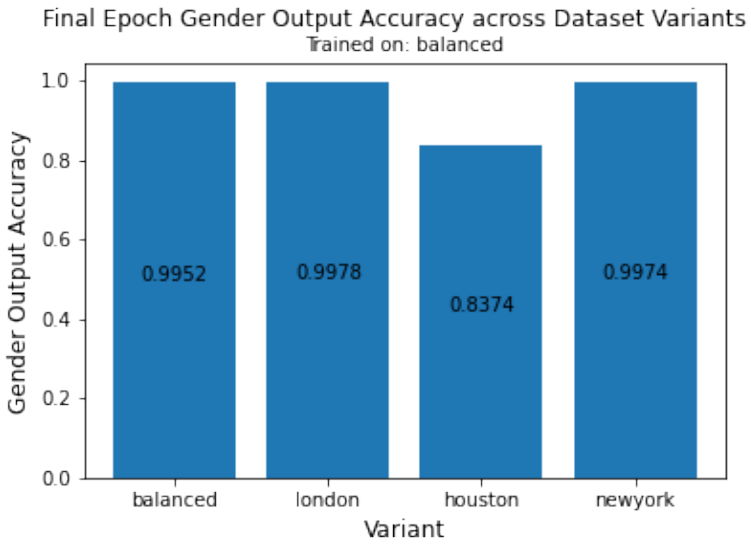


Figure 4.11: Final gender accuracy for the four variants.

4.2.4 Age Error

Figure 4.10 shows the age output train-test error curves of the four variants in discussion.

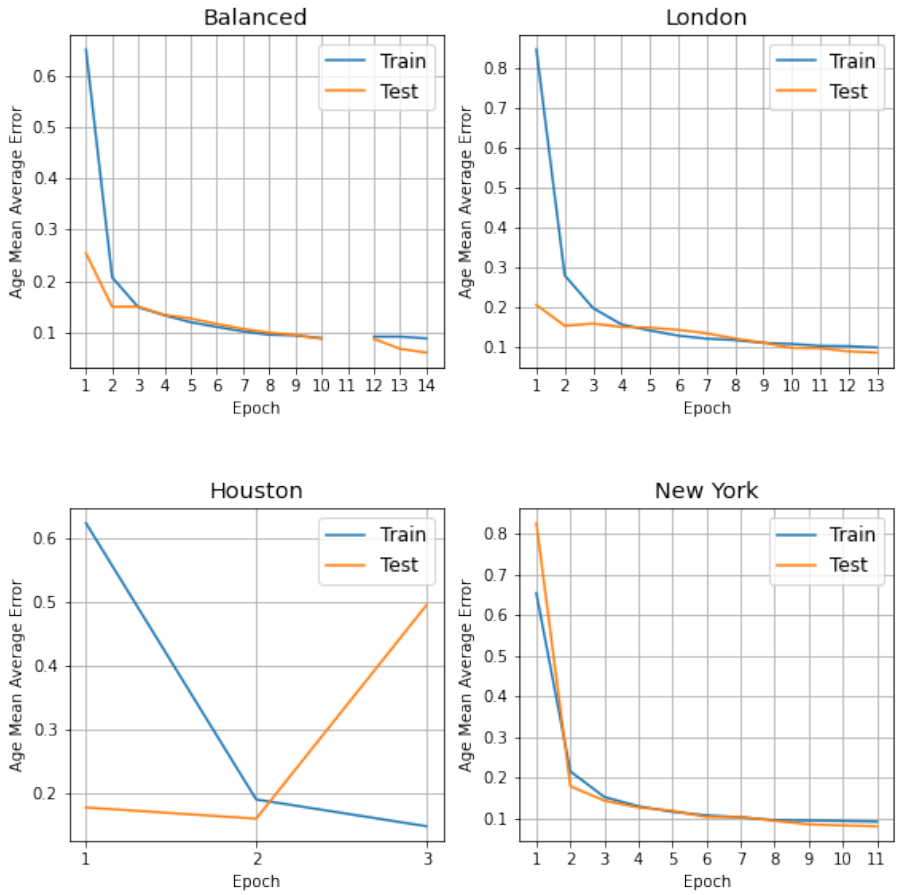


Figure 4.12: Age mean average error curve for the four distributions.

The Balanced dataset test mean average error begins at 0.2541, and then drops to 0.1502. After this, the error steadily decreases by 0.01 - 0.02 per epoch, before reaching a final mean average error of 0.06110. This behaviour indicates that the architecture is already good for age estimation, as it improves slowly over the 14 epochs. No large changes in error are necessary in order to stabilise the model. This is largely due to the loss weights - the age output has a loss weight of 4, compared to 1.5 for race and 0.1 for gender. There appears to be an inverse relationship between the loss weight and variance, as it is by no means a coincidence that the gender output plots as per Figure 4.10 feature the most noise between itself, race output accuracy, and age error plots.

The London dataset corroborates this sentiment even further. The first instance of the age mean average error is 0.2052. This then drops to 0.1531 by the next epoch, followed by a slight increase to

0.1585. From then onwards, there is a monotonic decrease of 0.005 - 0.01 across each epoch. This series of minute deltas indicate a convergence.

The same idea is reflected in the New York variant. However, it is worthwhile to note that it is the only dataset out of the three so far discussed, that has an initial test mean average error larger than the train mean average error. This isn't problematic however, as this drops from 0.8262 to 0.1788 by the second epoch. The mean average error continues to decrease by ≈0.01 per epoch, before reaching a final value of 0.08056.

Again, awkward results surface from the Houston dataset. It is the only variant in which the final epoch's mean average error was higher than the initial epoch's mean average error. The error rose over two-fold, from 0.1773 epoch 1 to 0.4948 in epoch 3. Although, there was a slight improvement at epoch 2 to 0.1597, but of course sharply negated by the next epoch. Due to the consistent bad performance across the metrics, it is best to regard the variant as anomalous. There was likely a random error during training and so the best resolution would be to run the model again. If the model still behaved in this unexpected manner, further investigation would need to be done to determine the root cause of this erratic behaviour.

Figure 4.13 plots the final age mean average errors of each dataset on a bar chart.

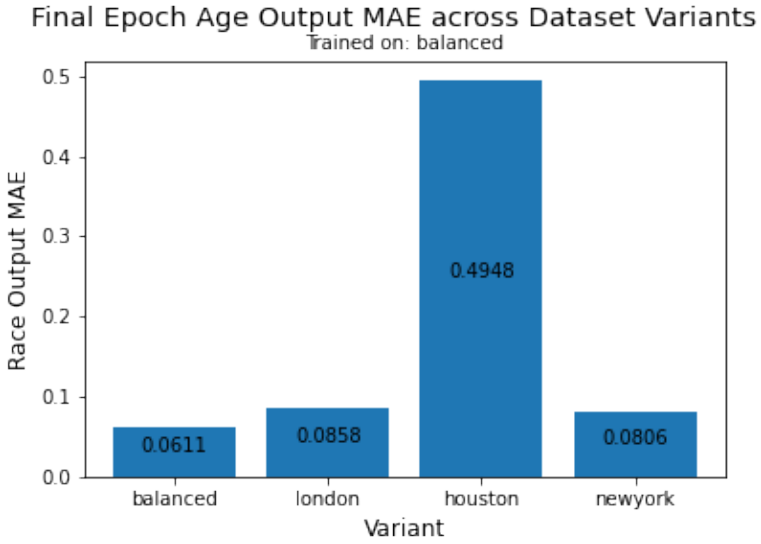


Figure 4.13: Final age output mean average error for the four variants.

Inspecting this chart evidences that the Balanced, London and

New York variants are suitable candidates for age prediction, due to the very low respective MAE values obtained.

4.2.5 Conclusions

The plots shown above were all trained on the Balanced dataset variant, therefore it is worthy to understand how the models fare as they are trained with and tested on a combination of different variants. Tables 4.2 to 4.5 tabulate the cross-dataset performance for race output accuracy, gender output accuracy, age output mean average error, and model loss.

Train	**Test**				μ	σ
	Balanced	London	New York	Houston		
Balanced	**0.9892**	0.9777	0.9714	0.5126	0.8627	0.2022
London	0.6198	**0.6797**	0.6484	0.6146	0.6406	0.002145
New York	0.3880	**0.5495**	0.3047	0.4870	0.4323	0.9350
Houston	**0.4661**	0.4062	0.4115	0.1250	0.3522	0.1333

Table 4.2: Cross dataset race output accuracies.

Train	**Test**				μ	σ
	Balanced	London	New York	Houston		
Balanced	0.9952	**0.9978**	0.9974	0.6719	0.9106	0.1380
London	0.6510	**0.7891**	0.7005	0.5781	0.6797	0.07673
New York	**0.6719**	0.6406	0.6615	0.5938	0.6420	0.03000
Houston	0.4635	**0.7292**	0.5703	0.6406	0.6009	0.09728

Table 4.3: Cross dataset gender output accuracies.

Train	**Test**				μ	σ
	Balanced	London	New York	Houston		
Balanced	**0.06110**	0.0858	0.08060	0.4948	0.1806	0.1817
London	0.3574	**0.1961**	0.2015	0.1986	0.2384	0.006873
New York	0.4522	0.2035	2.3529	**0.1902**	0.7997	0.9028
Houston	**0.1384**	0.1816	1.6245	1.0332	0.7444	0.6209

Table 4.4: Cross dataset age output mean average errors.

Train	**Test**				μ	σ
	Balanced	London	New York	Houston		
Balanced	**0.09880**	0.1350	0.1685	5.681	1.521	2.402
London	4.245	**2.173**	4.137	3.358	3.478	0.8276
New York	6.057	5.955	27.94	**5.129**	11.27	9.631
Houston	**3.974**	5.236	19.79	13.84	10.71	6.473

Table 4.5: Cross dataset model losses.

Broad inspection of the tables show that the balanced dataset performs the best. This is as expected, because present in the dataset are several examples of each race group and gender, as well as a wide variety of ages. Other variants have such sparse representation; for example, Houston has 17 South Asian faces and 16 Arab faces. Gaining a strongly representative range of faces in such few instances is near impossible - particularly when pit against the 950 Hispanic/Latino faces present in the variant. This is likely the reason behind the consistent poor performance observed in the Houston tests.

An interesting pattern can be found when observing the 'winning' test variants for each row. These top-performing variants are boldened in the tables. The best race output and gender output accuracies occur when the test variants under consideration are the Balanced and London datasets. Whereas, the mean average errors and model losses follow the same pattern of cities: Balanced performs best on Balanced, London performs best on London, New York performs best on Houston and Houston performs best on Balanced. Perhaps this is due to the polarity of the optimisation goal in question - race and output accuracies are maximised, and mean average error and model loss are minimised.

For each training dataset, the mean gender accuracies tend to be larger than the race accuracies. This is understandable, due to the binary nature of the classification. Because there is a larger set of possible outputs for race, intuitively, the accuracy would be lower given the finite (and relatively low) number of training examples.

4.3 Representative Images

In this subsection, we look at how modifying test images affects the predictive power of the facial recognition system.

4.3.1 Occlusion

Figure 4.14 is an indicative series of occlusions applied to the same face.

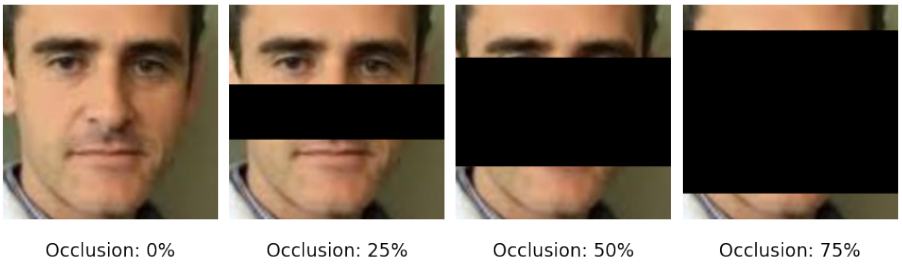


Figure 4.14: Various intensities of occlusion applied to a face.

The occlusion will be added to the test images, and the training dataset will remain unchanged. A geometric rectangle expanding from the image centre was selected as the occlusion shape of choice, because we made the grand assumption that the face in each image is centred about the nose. If this wasn't the case, the coordinates of the nose would have to be determined so that the occlusion rectangle could begin from that point. This is a lengthy feat however, necessitating manual labelling of the several-thousand large dataset, or the use of dlib. This procedure would run outside the scope of the project timeline, hence why the nose centre assumption was made.

Furthermore, such linear and monotonous occlusions might not be the norm in real life. For example, a CCTV may pick up three-quarters of a person's face, but the remaining quarter may be covered by an awkward object, like an umbrella or in more recent times, a face mask. In light of this, it might be ideal to consider superimposing random objects on top of the subject's face, or adding filters such as a layer of snow or rain, in order to simulate performance in an adverse lighting/weather situation. Alternatively, as a starting step, vertical or diagonal occlusion rectangles may be considered, in order to determine how different types of occlusion affects model performance.

Figures 4.15, 4.16 and 4.17 are respective plots of the race output accuracy, gender output accuracy and age output mean average error for the various occlusion amounts under consideration.

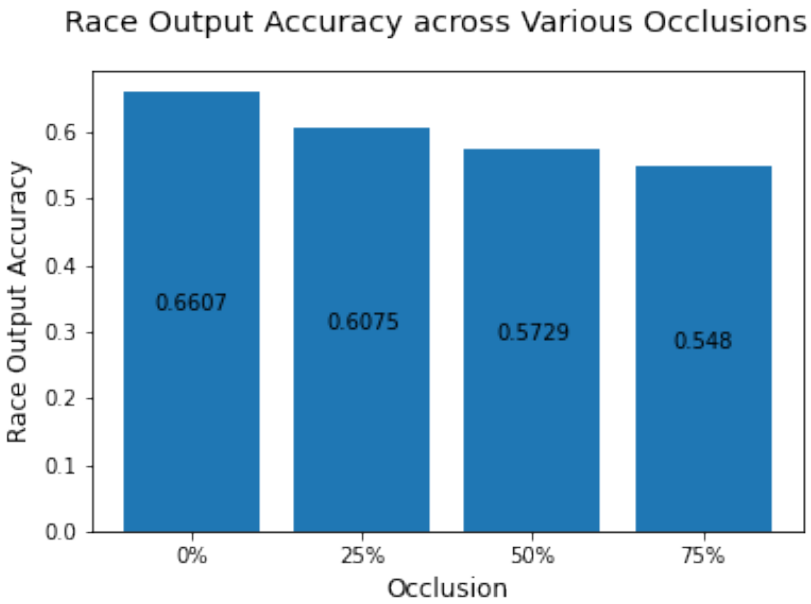


Figure 4.15: Race output accuracy for different occlusion amounts.

The race output accuracy results for the unedited, 0% occlusion test data are the highest amongst the other occlusion variants. This is expected since the test subject is completely bare and uncovered, therefore the machine has the most available information in order to make a prediction for race.

The race accuracies are all in close succession to each other, dispersed only by 0.1127. Understandably, as more of the face is removed, the accuracy slightly decreases. This can be seen in the $\approx 3\%$ accuracy decrease per 25% occlusion increase. However, as mentioned, there is not a significant level deviation between the accuracies. It can be said that this is due to the mechanism behind intelligent learning - as we are predicting race, the model would intuitively, although somewhat naïvely, assess the skin colour as a proxy to race, similar to what the PPB dataset authors do [35]. No matter the level of occlusion (so long as it is not excessively high to the point where the subject's skin is no longer visible), the skin colour can offer an insight to the machine to predict the race.

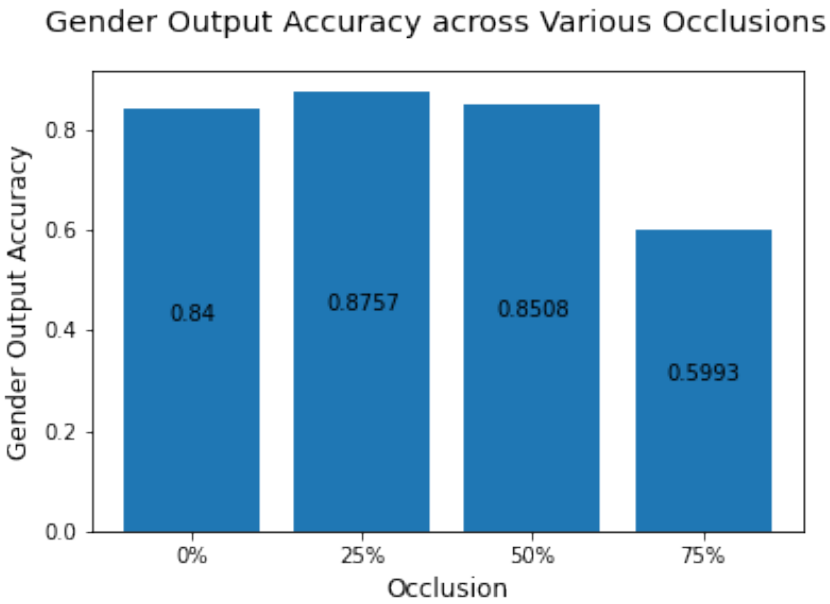


Figure 4.16: Gender output accuracy for different occlusion amounts.

For 0%, 25% and 50% occlusion tests, the gender output accuracies are fairly close, at 0.8400, 0.8757 and 0.8508 respectively. This is understandable, being that humans are dimorphic beings. This means that characteristics in appearance (e.g. hair length, facial structure) are typically very distinct for male and female. At these relatively low occlusion levels, the distinct characteristics are still very much visible to the computer. This offers an explanation the sharp drop in accuracy to 0.5993 for 75% occlusion. At this level of occlusion, it is very difficult to determine any gender-distinct facial features. On inspection of the 75% occlusion instance in Figure 4.14, identifying the subject as male might be a straight-forward and almost intuitive task. However, this is due to our unparalleled ability as humans to predict gender, also paired with our knowledge bias from having already seen the un-occluded image. But, to a computer, only seeing a fraction of a forehead and a chin substantially increases the difficulty to predict the gender of the subject in question because it has such few reference points.

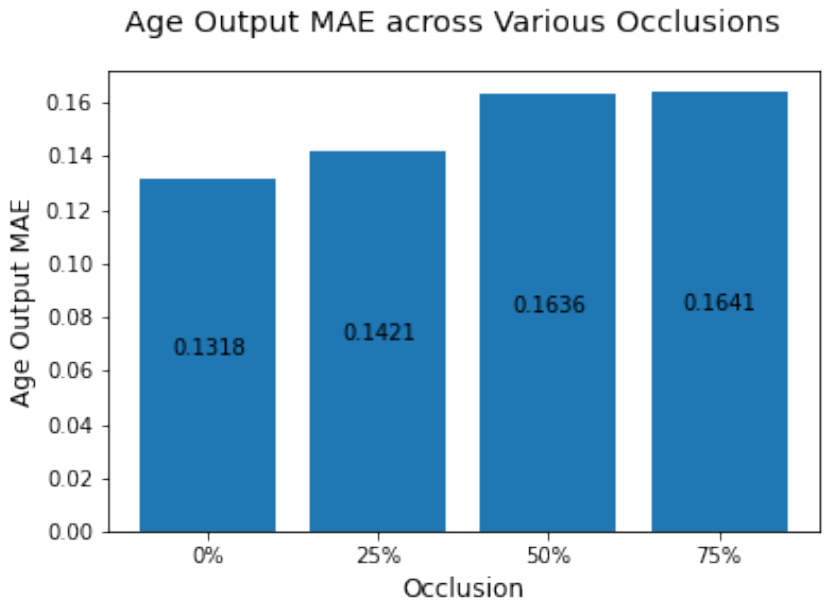


Figure 4.17: Age output MAE for different occlusion amounts.

Figure 4.17 is a plot of the age output mean errors for the different occlusion amounts. We see that as the occlusion amount increases, so does the mean output error. At just 0.005, the difference in error between 50% occlusion and 75% occlusion is minor. Using Figure 4.14 as a reference, we can theorise that the telling characteristics of age in a person's face include the bagginess of their eyelids and extent of definition of their nose. These are visible in the 25% occlusion instance, but not in the 50% or 75% counterparts. This provides a plausible explanation for why the mean average error increased significantly between 25% and 50% occlusion tests. Since there are much fewer critical datapoints to assess the age from in the 50% and 75% datapoints (the only difference between the two being the visibility of the eyebrows, which is not very insightful), the model behaved similarly in both cases, hence the close output errors.

4.3.2 Blurring

Figure 4.18 is a series of blur transformation of differing intensities applied to a face.

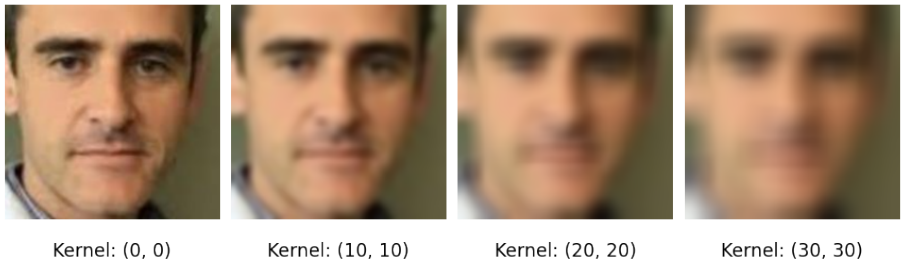


Figure 4.18: Blur kernels of various dimensions applied to a face.

The blur was added to the test images, while the training images remained unedited.

Figures 4.19, 4.20 and 4.21, respectively, are plots of the race output accuracy, gender output accuracy and age output mean average error for the various blur kernel sizes under consideration.

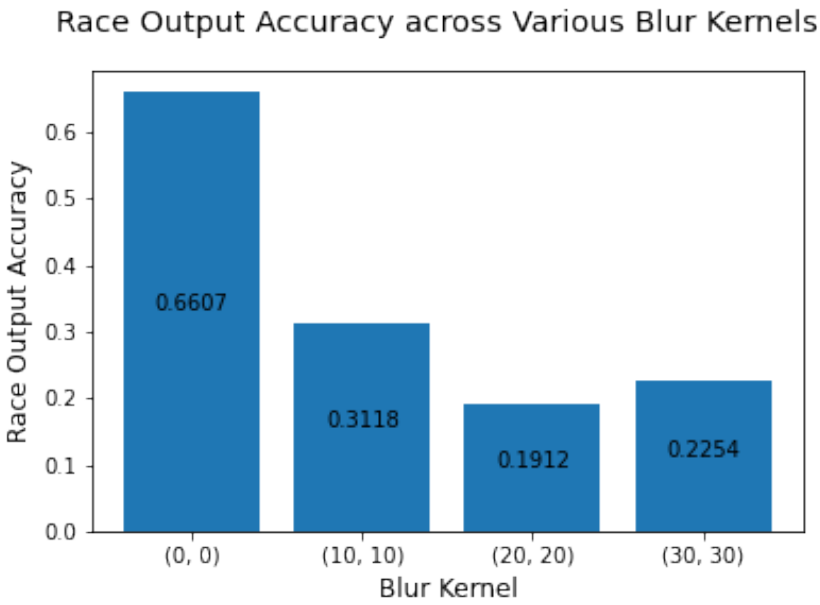


Figure 4.19: Race output accuracy for different blur kernel sizes.

Inspecting Figure 4.19, it is clear that blur has a serious impact on the race output accuracy. With zero blur, the system scores an accuracy of 0.6607. However, all other instances perform consider-

ably worse than this. The (10, 10) blur performs at a 52% deficit to the zero blur, and the (20, 20) and (30, 30) blurs perform even worse at 71% and 66% deficits, respectively. Because the box blur takes the average RGB values of surrounding pixels, it is likely that the background colours are incorporated in this calculation. The background colours are not predictable; neither should they be, as this is an in-the-wild test. Thus, these colours introduce an error in the model's ability to predict race. Furthermore, although we are analysing two-dimensional images, faces are three-dimensional - our noses protrude and eye-sockets sink inwards. These variations in depth give rise shadows across the face. The effect of the colour difference due to the blurs is only multiplied due to the averaging effect of the blur. As well as this, blurring facial hair adds a layer of confusion for similar reasons, since the footprint of the facial hair on the image is technically increased by virtue of the blur.

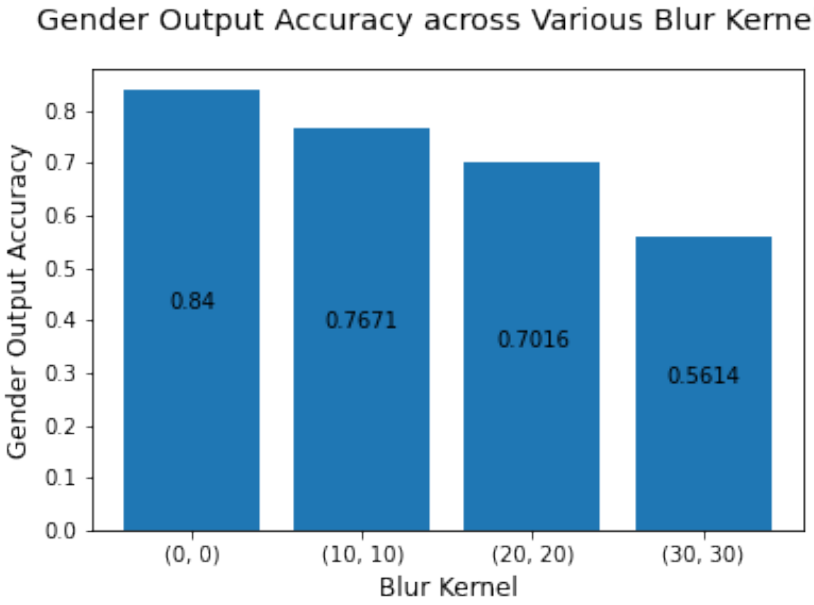


Figure 4.20: Gender output accuracy for different blur kernel sizes.

Figure 4.20 shows an inverse relationship between gender output accuracy and blur kernel size. The gender output accuracy consistently decreases, however in all cases, the accuracies are higher than its race output equivalents. This is a similar trend to that observed in the occlusion analysis; regardless of the blur, the model can still detect telling features such as hair length. Of course, this is a broad generalisation, as men can have long hair, and women can have short hair. However, it appears that for the most part, each

gender maintains mutually exclusive hair styles. Between kernels $(20, 20)$ and $(30, 30)$, a significantly larger delta in accuracy occurs. It has the largest decrease of 0.1402, compared to the 0.0729 and 0.0655, in order, seen across kernels $(0, 0)$, $(10, 10)$ and $(20, 20)$. It seems that past a certain kernel size threshold, the model begins to perform poorly. In light of this, it would be worth experimenting with numerous kernel sizes within this range to better understand the trend between blur kernel size and gender output accuracy.

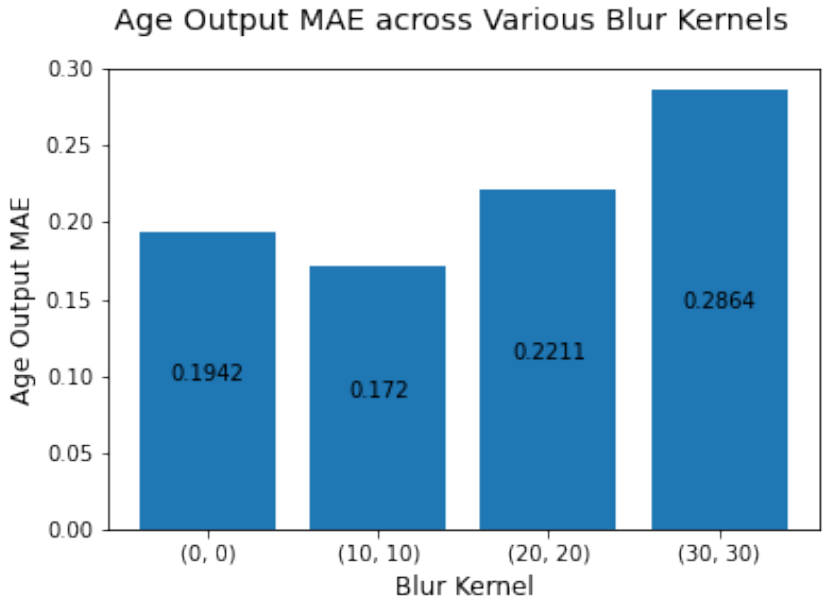


Figure 4.21: Age output MAE for different blur kernel sizes.

The general trend observed in Figure 4.21 is in line with intuition. Sans the age output mean average error for kernel size $(10, 10)$, the error increases as the kernel size increases. Physical features which are indicative of age, such as wrinkles, eyebags and roundness of face are much less distinguishable at higher blur intensities. This is the most probable explanation for this error trend. Again, it appears that the delta between size $(20, 20)$ and $(30, 30)$ is the largest observed, at 0.0653. This corroborates the idea that there is a critical kernel size within this range, at which above said size, the model performance decreases significantly.

4.3.3 Lighting

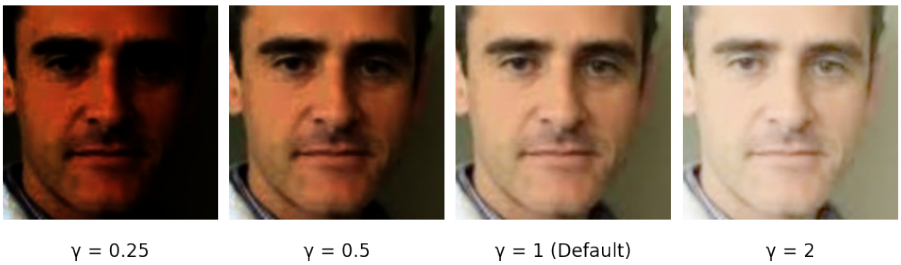


Figure 4.22: Varying gamma correction intensities applied to a face.

The decision was made to use a geometrically-increasing set (with constant ratio, $r = 2$) of gamma correction intensities, in place of an arithmetically increasing range. This is because as the intensity increased by a fixed amount, there was a less noticeable visual difference in the subject's face. Therefore, a geometric offset was required in order to provide results worthy of analysis.

Figures 4.23, 4.24 and 4.25 are respective plots of the race output accuracy, gender output accuracy and age output mean average error for the various gamma intensities under test.

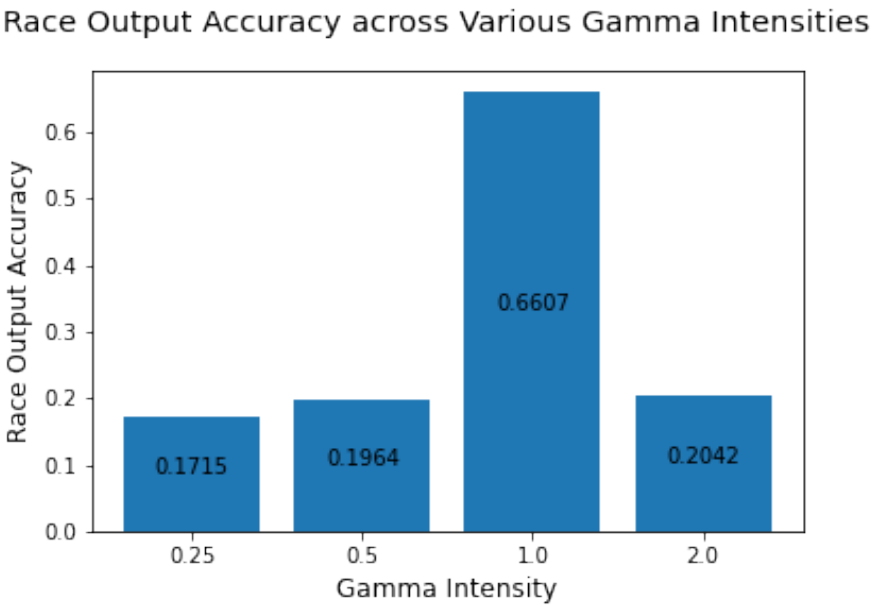


Figure 4.23: Race output accuracy for different gamma intensities.

Inspecting Figure 4.19 The unedited ($\gamma = 1.0$) instance yields the greatest accuracy of 0.6607. Whereas the other three instances are associated with significantly lower respective accuracies. Similar to the conclusions drawn in the occlusion and blurring tests, the machine learning model likely predicts race by using skin colour as a proxy. Applying a gamma correction to an image modifies the luminance of every pixel, thus the spectrum of colours under consideration by the model are shifted. The model was trained on the unedited variant and so it is accustomed to the original race colour range. Then showing the model a shifted spectrum in the test would source confusion in the prediction process, which explains the lower accuracies.

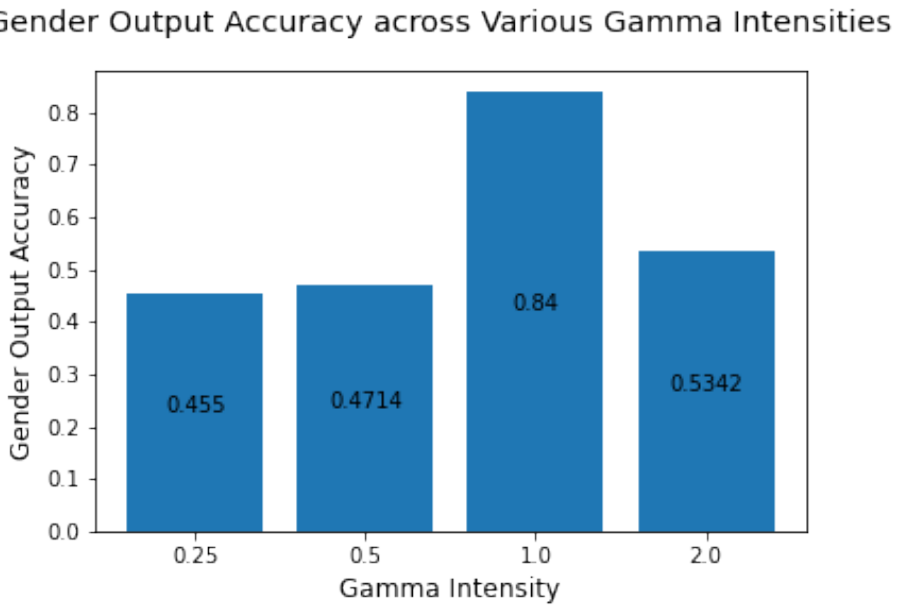


Figure 4.24: Gender output accuracy for different gamma intensities.

The gender output accuracy is under the same skin colour proxy effect. This follows from the unedited dataset reaching an accuracy of 0.8400, and the 0.25, 0.5 and 2.0 intensities scoring 0.4550, 0.4714 and 0.5342 respectively. But as aforementioned, due to the human being's dimorphic nature, telling characteristics such as hair length are still visible to the machine which can be used to make a more intelligent prediction. This is why each of the non-one γ intensity gender output accuracies are higher than its race output accuracy counterparts. What is striking in Figure 4.24 is the close agreement of the non-one gamma intensity accuracies. This trend can also be seen in Figure 4.23. Although there is an accuracy increase of 0.164 between the 0.25 and 0.5 intensity instances, the difference is not significant enough to confidently define a trend. The data suggests that for any non-one value of γ, the output accuracies will result as roughly similar, however this finding is subject to further scrutiny. A natural progression for this work would be to test a wider range of γ values to determine the relationship between gamma and accuracy. Intuitively, one can expect the accuracy to fit a bell curve, and so with a wider set of observed values, a χ^2 test might be used to evaluate the goodness of fit for a normal distribution.

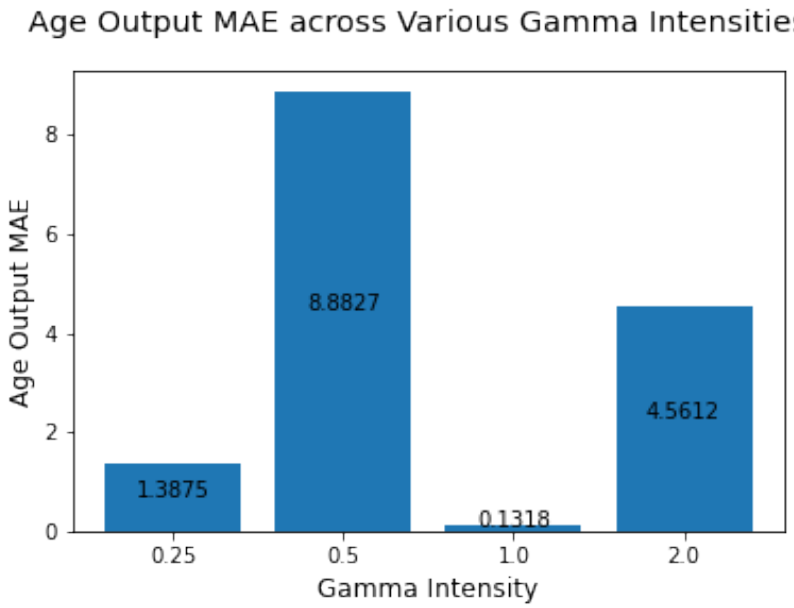


Figure 4.25: Age output mean average error for different gamma intensities.

Figure 4.25 shows an interesting set of results. Predictably, the age output error at $\gamma = 1.0$ is the lowest observed, because the test images' lighting is unchanged and so the model has the best opportunity to predict age. The largest error arises at $\gamma = 0.5$, with a value of 8.8827. This is significantly higher than the other errors observed. We would expect the error to increase the further we move away from $\gamma = 1.0$, and so this does not fit our prediction. As further investigation, more readings of γ should be taken in order to confidently establish a trend.

4.4 Testing

4.4.1 System Requirements Evaluation

Table 4.6 contains the evaluation of the system requirements set out earlier in the paper.

Requirement	Priority	Completion
Resize raw images into a new dimension (e.g. 224x224)	M	Completed
Train specified neural network on a given training set	M	Completed
Modify the training set to fit a given proportion	M	Completed
Test specified neural network on a given test set	M	Completed
Apply occlusion of a given amount to a given test instance	M	Completed
Apply blurring of a given amount to a given test instance	M	Completed
Apply gamma correction of a given intensity to a given test instance	M	Completed
Handle valid or corrupt test images	D	Completed
Easily integrate with an external system to be used in real-world practice	D	Completed
Evaluate algorithmic bias by accuracy and error	M	Completed
Evaluate algorithmic bias with an extended set of metrics and plots	D	Partially Completed
Design code with OOP principles	D	Partially Completed

Table 4.6: System requirement evaluation.

We have a respectable coverage of the requirements - no requirements are without attempt; only 2 of the 12 defined are not fully completed. The system evaluates algorithmic bias by assessing the accuracy of the race and gender outputs, and the mean average error of the age. It would be ideal to have a few more metrics such as mean squared error for the age output. We do attempt visualisation of performance by generating confusion matrices, however it would be beneficial to produce ROC curves for the race and gender output performance.

A Python script was made for each factor of investigation. As the scripts share a considerable amount of the same logic, there is an opportunity to refactor the code to follow an object-oriented programming (OOP) pattern. At present, defined object classes do exist, but not throughout the entire codebase, hence why this requirement is 'Partially Completed'. There is a `FaceDataGenerator` class which contains all the logic used to preprocess the images, generate train/test split indexes and apply any necessary effects (e.g. blur). There is also a `MultiOutputModel` which stores the logic for building the model architecture.

5 Conclusions

In this section, we conclude our findings and discuss opportunities to build on the work carried out in this study.

5.1 Summary

We understood the problem of artificial intelligence from a sociopolitical perspective; its common use-cases and the applications of facial recognition in societal domains. We argued the timeliness and necessity of this investigation by providing examples of malicious use of this technology. We then performed a literature review of the history and evolution of neural networks in the domain of facial recognition, the current state-of-the-art and where novelty can be introduced. We systematically researched and selected a starting dataset (UTKFace), and justified the introduction of a more representative set of race groups, and then modified the UTKFace dataset annotations to fit these groups. Additional photos were scraped from Flickr and subsequently stitched together with the UTKFace photos to create an amalgamated dataset with a better representation of Arab faces. This dataset was used to train models based on three leading facial (image) recognition algorithms; VGG19, AlexNet and ResNet50. We examined their respective abilities to predict race, age and gender by evaluating the associated accuracies and errors. We experimented with the dataset's race group compositions by changing it to mimic the ethnicity proportions in diverse cities such as London, New York and Houston, and put these carefully constructed dataset variants under test in a machine learning model. To then understand how image quality affects model performance, we manipulated the test images by performing blur, occlusion and gamma correction operations on it. We presented the data from each of these three investigations in a series of plots, and provided critical evaluations and explanations for all of the results.

5.2 Further Work

5.2.1 Algorithm

ReLU was the main activation function used when the different network architectures were being scrutinised. Variants of ReLU do exist - namely PReLU and LeakyReLU. LeakyReLU allows a small gradient for negative x values, the magnitude of which is determined

by a constant gradient α. This is seen in Equation 5.1:

$$\text{LeakyReLU}(x) = \begin{cases} x & \text{for } x > 0, \\ \alpha x & \text{for } x \leq 0 \end{cases} \tag{5.1}$$

Using large values of α have been found to increase model accuracy, as well as the learning rate [64]. Hence, it would be insightful to experiment with this activation layer and see how it affects the model performance. This leads onto further opportunities for hyperparameter tuning and refinement. Some hyperparameters that could be investigated for their effect on model performance are:

- Learning rate
- Gradient descent algorithm (Adam was used in this study)
- Momentum
- Number of hidden layers
- Number of neurons

To perform hyperparameter tuning, applying GridSearch to k-fold cross-validation would be the method of choice. Cross-validation is a data partitioning strategy which effectively uses the dataset to build a more generalised model [52]. The objective of machine learning is generalisation; that is, to develop a model that reliably predicts any unseen instances. The general process of k-fold cross-validation is as follows:

1. The entire dataset is randomly split into independent k-folds, *without* replacement.
2. $k - 1$ folds are used to train the model, and 1 fold is used for evaluating performance.
3. The procedure is repeated k times, so that we obtain k number of performance estimates (e.g. accuracy, MAE).
4. These performance estimates are then averaged, giving us a more generalised understanding of performance.

Because our dataset is relatively small (see Table 3.1), the special case method of Leave-one-out-cross-validation (LOOCV) would be used here. In this instance, we set $k = n$, where n is the number of observations in the dataset. Therefore, only one training sample is used for testing during each iteration. The cross-validation would then be applied to a grid of test hyperparameter values.

GridSearch would be applied to the cross-validation procedure to complete the hyperparameter optimisation. The premise of GridSearch is to look for an optimal combination of hyperparameter values in a way that can further improve the model's performance. Evaluating the model for each combination of hyperparameters with k-fold cross-validation results in a more robust and accurate model. Figure 5.1 is a high-level illustration of 5-fold cross-validation with GridSearch for hyperparameter tuning.

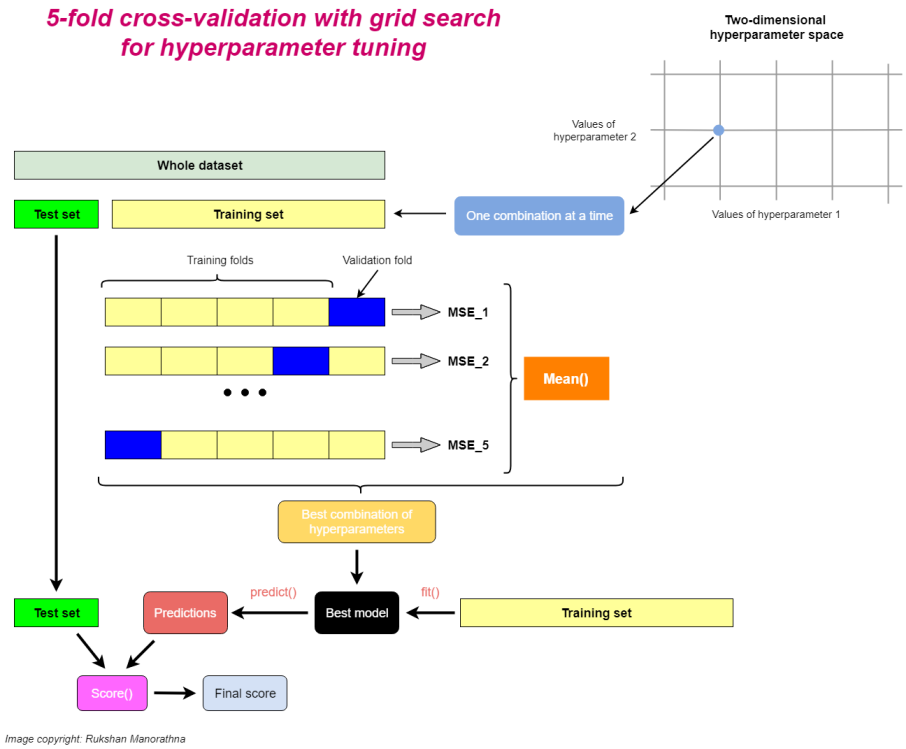


Figure 5.1: High-level overview of the cross-validation + GridSearch process [52].

5.2.2 Distribution

To further mimic a real-world context, it would be beneficial to test the model exclusively on novel datasets. As we want to measure the model's effectiveness on diverse races, the most reliable way to ensure a wide representation is to use images with verifiable locations of origin. Geo-tagged tweets are the best source for these. Images uploaded by Twitter users can be identified by their longitude and latitude [57]. Thus, after picking a set of countries, tweets (and ultimately images) originating from the desired locations can be found.

However, due to time constraints, this was not possible. Because raw, unannotated images would be collected, it would have to go through a classification and labelling process. To do this in-house would require an incredible number of hours, and to outsource the work to a service such as Amazon Mechanical Turk would be fairly expensive. This is the same reason why UTKFace was selected as the base dataset of choice, opposed to other contenders such as FFHQ.

If these time and cost restrictions didn't exist, further datasets would be compiled. It would be interesting to compile a dataset of faces found at protests. In 2017, Won *et al.* developed a novel visual model which can recognise protesters, describe their activities by visual attributes and estimate the level of perceived violence in an image [63]. This is a real-world example of where facial recognition technology is used, and so it would be worth incorporating the dataset Won *et al.* created for their study. The authors collected most of the data from Google Images by using keywords such as "London riots" or "football game" (for undeniably negative cases). The dataset contains a wide range of race and gender groups, making for a suitable dataset candidate.

5.2.3 Representative Image Quality - Blurring

Box blur, perhaps the simplest blurring transformation, was used in this investigation. Other types of blurring exist, such as Gaussian blurring, which is the result of blurring an image by a Gaussian function. The effect of this blurring technique resembles that of viewing an image through a semi-opaque screen. In a real-world context, particularly in reference to the ongoing COVID-19 situation, this could mimic CCTV taking a photo of someone wearing a full covering face visor. Thus, it would be useful in the examination of change in model performance as the blurring type is varied.

5.2.4 System Design

As mentioned previously, OOP design is prevalent throughout the majority of the codebase, but not everywhere. Although forgoing OOP is useful for quickly prototyping code, it skips out on the pattern's associated benefits. OOP has multiple advantages:

- **Code Maintenance**: when writing logic for objects, it only needs to be written once, and it will globally update wherever used. This provides homogeneity across all instances of the object in the codebase, reducing the likelihood of broken code.

- **Troubleshooting**: building on the previous point, writing code in an OOP manner makes the process of troubleshooting and debugging much easier. As there isn't duplicate code scattered around the project, it is easy to track and source the location of errors.
- **Code Reuse**: reusing code allows for quick horizontal and vertical code iteration. This increases the project development throughput significantly. Reusing code also promotes extensibility as the codebase can be incorporated into a wider system.

6 Evaluation

This section contains an evaluation of the project as a whole; an insight on the operational aspect of completing the project, and a reflection on the dissertation from conception to delivery.

6.1 Project Management

6.1.1 Deliverables

The project consisted of numerous deliverables submitted mostly across the latter half of the MSc course timeline:

- **Research Proposal**: a preliminary discussion into how research into race, gender and age bias would be conducted. This was submitted as coursework in partial fulfilment of the CS908 Research Methods module, however it contributed greatly towards the research direction of this project overall.
- **Presentation**: a short presentation illustrating the necessity and timeliness of the investigation, followed by a briefing on the main project goals and it's various considerations, as well as work done so far.
- **Interim Report**: a scholarly pit-stop providing an overview of and reflection on the progress to date, accompanied by an action plan to complete the remaining work in the project.

6.1.2 Management and Documentation Tools

A range of tools were used for managing the project:

- **Notion**: an all-in-one digital workspace for notes and tasks [39] has been used as a logbook. Organised into weekly sections, the logbook has been used as a point of reference for useful resources, brainstorms, and summary of the progress made during the week. This has proved useful in collating the report.
- **GitHub**: a Git-based version control system which tracks code versions over time [11]. GitHub allows for documentation of code progression (by making frequent repository commits) and code sharing with collaborators and other users. GitHub was selected over alternative version control platforms such as Perforce and SVN, due to familiarity obtained by using it in prior endeavours. This saved development time as less time would be invested into learning to become well versed in a new technical product.

- **Overleaf**: an online LaTeX editor and compiler [45]. This was used over desktop alternatives such as TeXMaker because of its web-based nature; Overleaf can be accessed from any modern web browser; allowing the opportunity to write the dissertation report from virtually anywhere.

6.1.3 Timeline

Figure 6.1 is a Gantt chart detailing the project timeline.

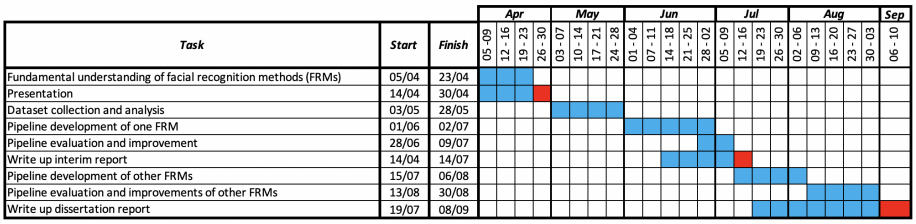


Figure 6.1: Project Gantt chart.

This is chart is largely unedited from its previous mention in the interim report, indicating an adequate estimation of the times required for each task. Project work coincided with MSc examinations during mid-May to mid-June and so the decision was made to not have several dissertation-related tasks running simultaneously in this period.

6.1.4 Risk Management

Risk Event	Consequence(s)	Probability	Impact	Risk Response Plan
Postural problems	Back ache and upper limb disorders	Low	Medium	• Assume a correct seating position, as per HSG57 guidelines • Regularly perform posture checks • Consider using a standing desk
Visual problems	Eye strain and headaches	Medium	High	• Wear computer glasses to reduce glare and blue-light transmission • Work in a well-lit environment
Fatigue and stress	Reduced focus and productivity, impacting the quality of the project work	Medium	High	• Work in 25-minute sessions accompanied by 5-minute breaks
Scope creep	Inability to deliver project in full by the deadline	Low/Medium	High	• Perform weekly review sessions to verify that the research is not exceeding project constraints and objectives.
Loss of project data	Delay in project competition by having to redo work	Low	High	• Create frequent backups in two cloud locations
No access to training datasets	Unable to perform machine learning operations	Low	High	• Prioritise using freely available datasets • Move onto bespoke and private datasets if permission is received

Figure 6.2: Project risk assessment.

Figure 6.2 is a summary of the risks associated with this project.

As the project is worth 60 CATS ($\approx$ 600 hours of study), these risks had to be considered from a long term perspective. For example, loss of project data occurs near instantly, but postural problems may slowly develop over a period of weeks or months. These less quickly noticeable risks were prioritised for mitigation. All work on the project was done in 25-minute sessions accompanied by 5 minute breaks, and computer glasses were worn to avoid fatigue and stress, as arguably this is the most important risk. If fatigued, performing to a decent standard would not be possible. The risk assessment was reviewed fortnightly to serve as a reminder and also ensure that there was a diligent adherence to it.

6.2 Project Reflection

6.2.1 Challenges

From the middle of July onwards, the dissertation was carried out while working full-time at an internship. At the time of constructing the Gantt chart, the internship role was not yet confirmed, and so the additional workload due to this was unforeseen. To accommodate for this, work on the dissertation was carried out typically in the uninterrupted hours of the early mornings, and late nights where possible. This is because the bulk of the daytimes were committed to internship work.

While developing the technical solution, in late June, a memory issue was encountered. As we are training thousands of images and passing them into memory, the system frequently raised an Out of Memory (OOM) error. Solving this one issue took about two weeks. At first, a tuple of RGB values of each pixel from each image was extracted and saved into a 'master' Pandas dataframe as a NumPy array, alongside each image's race, age and gender annotations. This was done so that the raw image dataset could be scrapped as all the necessary data was contained in the dataframe, ready for ingestion by the model. However, this was not feasible, because the dataframe itself was so big that it also raised an OOM error. It was then decided to save this dataframe into CSV files, in chunks of 500 images at a time. The image data would be loaded chunk-by-chunk and stitched together, so that it does not instantly deplete the computer memory. However, CSV stores plaintext contents; therefore with more than 2×10^9 integer values to store, stitching again led to memory limit problems. Pandas offers the ability to save dataframes to a plethora of formats, some in binary form (e.g. .h5, .pickle), which offer encoding and compression. These were all tested, however to no avail.

After more research, it was found that keras, the deep learning library used in this project, has a data generation capability which solves this exact problem. Memory issues are the norm when working with large amounts of information; particularly images, which are so dense in data. Data generators allow for real-time consumption of data into the keras model, so that images don't need to be loaded into memory prior to training the model. This fixed the issue, while also keeping memory usage at a very low level. Fortunately, the 2 weeks spent on this issue was not problematic, as the project was ahead of schedule. The decision was made to frontrun a lot of project work well before the summer period as the internship had already been confirmed at this point, and busyness was anticipated.

6.2.2 Ethical Implications

This project involves the scrutinisation of thousands of faces. Only publicly available datasets have been sourced, therefore this research does not infringe on the privacy of anyone, or raise any ethical implications. The source of the photos in the datasets are Google Images and Flickr. Research ethics reviews are generally not required in projects which perform secondary analysis of publicly available data, however this does not extend to the use of social media data in research. As discussed in the Further Work section, considering geo-tagged tweets to find faces reliably classified in race is a good idea. However, this requires primary collection of data. Should this course of action be taken, the University's Biomedical and Scientific Research Ethics Committee (BSREC) should be contacted, and an appropriate ethical review should be conducted in tandem with them.

References

[1] Amazon Mechanical Turk, Inc. Amazon Mechanical Turk. https://www.mturk.com/. [Online - accessed 3-September-2021].

[2] Amrita Mazumdar. Understanding Box Blur. http://amritamaz.net/blog/understanding-box-blur. [Online - accessed 31-August-2021].

[3] Anaconda Inc. Anaconda — The World's Most Popular Data Science Platform . https://www.anaconda.com/. [Online - accessed 06-August-2021].

[4] Arab America. Arab American Cultural Center — New York. https://www.arabamerica.com/new-york/. [Online - accessed 2-September-2021].

[5] BetterRetailing. Facewatch facial recognition technology provides a proven crime deterrent in-store. https://www.betterretailing.com/legislation-competition/retail-crime/facewatch-facial-recognition-is-proven-crime-deterrent/. [Online - accessed 07-September-2021].

[6] BetterRetailing. Facewatch facial recognition technology provides a proven crime deterrent in-store. https://www.betterretailing.com/legislation-competition/retail-crime/facewatch-facial-recognition-is-proven-crime-deterrent/. [Online - accessed 07-September-2021].

[7] Bjarne Stroustroup. Foundations of C++ — ETAPS Corrected Draft. https://www.stroustrup.com/ETAPS-corrected-draft.pdf, 2012. [Online - accessed 05-Aug-2021].

[8] J. Brownlee. A Gentle Introduction to the Rectified Linear Unit (ReLU). https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/, 2019. [Online - accessed 14-Aug-2021].

[9] J. Buolamwini and T. Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In S. A. Friedler and C. Wilson, editors, *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 77–91. PMLR, 23–24 Feb 2018.

[10] D. Chen, X. Cao, F. Wen, and J. Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[11] chigbojk (Jordan Chigbo). GitHub Repository. `https://github.com/chigbojk?tab=repositories`. [Online - accessed 10-July-2021].

[12] Computer Science Wiki. Max-pooling / Pooling . `https://computersciencewiki.org/index.php/Max-pooling_/_Pooling`. [Online - accessed 14-August-2021].

[13] C. M. Cook, J. J. Howard, Y. B. Sirotin, J. L. Tipton, and A. R. Vemury. Demographic effects in facial recognition and their dependence on image acquisition: An evaluation of eleven commercial systems. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 1(1):32–41, 2019.

[14] dlib. dlib C++ Library . `http://dlib.net/`. [Online - accessed 12-March-2021].

[15] FashionUnited. Facial recognition at Ruti captures new retail landscape. `https://fashionunited.uk/news/retail/facial-recognition-at-ruti-captures-new-retail-landscape/2020030947884`. [Online - accessed 07-September-2021].

[16] Flickr. Flickr. `https://www.flickr.com/`. [Online - accessed 31-August-2021].

[17] D. J. Fuchs. The dangers of human-like bias in machine-learning algorithms. *Peer 2 Peer*, 2(1), May 2018.

[18] K. Fukushima and S. Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. 1982.

[19] N. Furl, P. J. Phillips, and A. J. O'Toole. Face recognition algorithms and the other-race effect: computational mechanisms for a developmental contact hypothesis. *Cognitive Science*, 26(6):797–815, Nov. 2002.

[20] G. Givens, J. Beveridge, B. Draper, P. Grother, and P. Phillips. How features of the human face affect recognition: a statistical comparison of three face recognition algorithms. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer*

Vision and Pattern Recognition, 2004. CVPR 2004., volume 2, pages II–II, 2004.

[21] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In G. Gordon, D. Dunson, and M. Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[22] P. Grother, M. Ngan, and K. Hanaoka. Face recognition vendor test part 3. Technical report, Dec. 2019.

[23] V. Gupta. Face Detection – OpenCV, Dlib and Deep Learning (C++ / Python) — learnopencv.com. `https://learnopencv.com/face-detection-opencv-dlib-and-deep-learning-c-python/`, 2018. [Online - accessed 05-Aug-2021].

[24] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

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

[26] HireVue. Video interview software & platform — hirevue, Dec 2020.

[27] J. J. Howard, Y. B. Sirotin, and A. R. Vemury. The effect of broad and specific demographic homogeneity on the imposter distributions and false match rates in face recognition algorithm performance. In *2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–8, 2019.

[28] ImageNet. ImageNet Large Scale Visual Recognition Challenge (ILSVRC). `https://www.image-net.org/challenges/LSVRC/`. [Online - accessed 7-September-2021].

[29] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[30] James Ives. Scientists use facial recognition technology to develop tool for monitoring ICU patient's safety. `https://www.news-medical.net/news/20190603/Scientists-use-facial-recognition-technology-to-develop-tool-`

for-monitoring-ICU-patients-safety.aspx. [Online - accessed 07-September-2021].

[31] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014.

[32] H. E. Khiyari and H. Wechsler. Face verification subject to varying (age, ethnicity, and gender) demographics using deep learning. *Journal of Biometrics & Biostatistics*, 07(04), 2016.

[33] B. F. Klare, M. J. Burge, J. C. Klontz, R. W. Vorder Bruegge, and A. K. Jain. Face recognition performance: Role of demographic information. *IEEE Transactions on Information Forensics and Security*, 7(6):1789–1801, 2012.

[34] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th International Conference on Computer Vision*, pages 365–372, 2009.

[35] A. J. League. Request Data Sets - AJL. `https://www.ajl.org/connect/request-dataset-for-research`. [Online - accessed 02-March-2021].

[36] Matplotlib. matplotlib.pyplot - matplotlib documentation. `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.html`. [Online - accessed 2-September-2021].

[37] Message Passing Interface Forum. Mpi: A message-passing interface standard, version 2.2. Specification, September 2009.

[38] NEC. NeoFace Watch. `https://www.nec.com/en/global/solutions/biometrics/face/neofacewatch.html`. [Online - accessed 07-September-2021].

[39] Notion. Notion - The all-in-one workspace for your notes, tasks, wikis, and databases. `https://notion.so`. [Online - accessed 02-March-2021].

[40] NtechLab. Face recognition system for airport and public transport. `https://ntechlab.com/solution/transport-security/`. [Online - accessed 07-September-2021].

[41] NumPy. NumPy. `https://numpy.org/`. [Online - accessed 3-September-2021].

[42] NVIDIA. CUDA Zone — NVIDIA Developer . https://developer.nvidia.com/cuda-zone. [Online - accessed 12-March-2021].

[43] OpenCV. *The OpenCV Reference Manual*, 2.4.13.7 edition, April 2014.

[44] A. J. O'TOOLE, K. DEFFENBACHER, H. ABDI, and J. C. BARTLETT. Simulating the 'other-race effect' as a problem in perceptual learning. *Connection Science*, 3(2):163–178, Jan. 1991.

[45] Overleaf. Overleaf, Online LaTeX editor. http://overleaf.com/. [Online - accessed 2-September-2021].

[46] pandas. pandas - Python Data Analysis Library. https://pandas.pydata.org/. [Online - accessed 2-September-2021].

[47] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[48] P. Phillips, P. Grother, R. Micheals, D. Blackburn, E. Tabassi, and M. Bone. Face recognition vendor test 2002. In *2003 IEEE International SOI Conference. Proceedings (Cat. No.03CH37443)*, pages 44–, 2003.

[49] P. J. Phillips, F. Jiang, A. Narvekar, J. Ayyad, and A. J. O'Toole. An other-race effect for face recognition algorithms. *ACM Transactions on Applied Perception*, 8(2):1–11, Jan. 2011.

[50] Race Disparity Unit, UK Cabinet Office. List of ethnic groups. https://www.ethnicity-facts-figures.service.gov.uk/style-guide/ethnic-groups. [Online - accessed 02-March-2021].

[51] S. J. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

[52] Rukshan Pramoditha. k-fold cross-validation explained in plain english. https://towardsdatascience.com/k-fold-cross-validation-explained-in-plain-english-659e33c0bc0. [Online - accessed 01-September-2021].

[53] Runai Labs Ltd. Understanding Deep Convolutional Neural Networks - Run:AI. https://www.run.ai/guides/deep-learning-for-computer-vision/deep-convolutional-neural-networks/. [Online - accessed 8-August-2021].

[54] K. K. S, K. Vangara, M. C. King, V. Albiero, and K. Bowyer. Characterizing the variability in face recognition accuracy relative to race. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2278–2285, 2019.

[55] Slurm. Slurm Workload Manager. `https://slurm.schedmd.com/overview.html`. [Online - accessed 3-September-2021].

[56] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, Jan. 2014.

[57] Z. C. Steinert-Threlkeld. *Twitter as Data*. Cambridge University Press, Jan. 2018.

[58] I. Sárándi, T. Linder, K. O. Arras, and B. Leibe. How robust is 3d human pose estimation to occlusion?, 2018.

[59] The MathWorks, Inc. What is a Neural Network? - MATLAB Simulink . `https://uk.mathworks.com/discovery/neural-network.html`. [Online - accessed 7-August-2021].

[60] TheAILearner. Power Law (Gamma) Tranformations . `https://theailearner.com/2019/01/26/power-law-gamma-transformations/`. [Online - accessed 31-August-2021].

[61] A.-C. Tsai, Y.-Y. Ou, W.-C. Wu, and J.-F. Wang. Occlusion resistant face detection and recognition system. In *2020 8th International Conference on Orange Technology (ICOT)*, pages 1–4, 2020.

[62] U.S. Department of Commerce. Explore Census Data. `https://data.census.gov/cedsci/`. [Online - accessed 2-September-2021].

[63] D. Won, Z. C. Steinert-Threlkeld, and J. Joo. Protest activity detection and perceived violence estimation from social media images, 2017.

[64] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network, 2015.

[65] D. Zeng, R. Veldhuis, and L. Spreeuwers. A survey of face recognition techniques under occlusion, 2020.

[66] S.-Y. Zhang, Zhifei and Q. Hairong. Age progression/regression by conditional adversarial autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

A GPU Batch Specifications

- **CPU**: Core i5-8500 @ 3.00GHz (6 Cores)
- **RAM**: 64GB
- **GPU**: RTX 2080Ti 11GB
- **Total Cores**: 12