Real-Time Application of Deepfake for De-Identification: Privacy Preservation and Data Protection

Muhsin Mustapha Inuwa

University of Salford

School of Science, Engineering & Environment (SEE)

A thesis submitted in partial fulfilment of the requirements of the University of Salford for the Degree of Doctor of Philosophy

July 2023

ABSTRACT

In an era marked by mounting concerns over data privacy and protection, conventional regulatory measures have proven inadequate against cyber-attacks, complicating data sharing for research and development. Meanwhile, traditional face de-identification methods often result in the complete erasure of facial information, hampering facial behaviour analysis. This thesis addresses these challenges by proposing a real-time deepfake deidentification for privacy preservation and data protection. Leveraging a first-order motion model and Mediapipe model of deidentification, the study investigates methods to accurately identify multiple faces within a single image, crucial for comprehensive deepfake models. Three distinct models were developed and tested to achieve deidentification of created deepfakes. Experimentation from various angles revealed differing levels of success, with considerations such as processing power, model openness, and training data quality influencing outcomes. Despite challenges, the study demonstrates the feasibility of real-time deepfake technology for privacy preservation and data protection. The proposed pipeline offers potential solutions to ethical concerns associated with data sharing, with implications extending to healthcare, autonomous vehicles, and unmanned aerial vehicle technology.

.

ACKNOWLEDGEMENT

I wish to express my sincerest gratitude to those who have contributed to my journey throughout this PhD programme.

First and foremost, my profound thanks go to my supervisor, Professor Mo Saraee. His tireless academic guidance, invaluable advice, consistent encouragement, and unwavering support have been instrumental in the successful completion of my studies. I am deeply indebted to him for his mentorship.

I would also like to extend my gratitude to the other members of the supervisory panel, notably Dr Meisam Brabie from the University of Leeds. Their contributions to my research have been invaluable.

Within the School of Science, Engineering & Environment, I'm thankful for the camaraderie and intellectual stimulation provided by my colleagues, particularly the Data Science Research and Teaching Team. Special acknowledgement goes to Princewill Obuzor, whose consistent support and accountability have greatly enriched the final stages of my PhD.

I am profoundly thankful for Khadijah (Sugrah). Her encouragement and support, especially in times when I needed it the most, have been invaluable to me.

My family has been a pillar of support throughout this journey. I am particularly indebted to my mother, Hajiya Rakiya Shehu Runka, whose moral support and wise counsel have been unwavering. My father, Dr Mustapha Muhammad Inuwa's constant encouragement and his habit of challenging me to strive for more have been significant motivators. I also owe thanks to my sisters, Mufida, Muhsina, and Mumtaza Mustapha Inuwa, as well as to my brothers Sadique Naseer and Salim Ahmad and my friends and family back home.

The love, tolerance, and inspiration from my wife, Jalila Buhari, and daughter, Ruqayya Naia, have been my greatest source of motivation and strength. I am profoundly grateful for their unwavering support. I Love you so much.

Lastly, I wish to thank the Petroleum Technology Development Fund in Nigeria for funding this research. Their financial support has been pivotal in bringing this research to fruition.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Overview

This chapter provides the foundation for the research conducted in this study. It commences with an exploration of the research background, framing the context within which the study is positioned. This allows for a clear understanding of the pre-existing knowledge and conditions relevant to the research. Next, the problem statement, which defines the specific issues the research seeks to address is presented. The problem statement lays the groundwork for the research questions, which guide the overall trajectory of the study. These research questions formed the core which helped narrow the focus and define the research boundaries.

The justification for this research is then elucidated, detailing why the study is both necessary and timely. This section underscores the significance of the research in the broader academic and societal context, highlighting its potential impact and relevance. Next, the chapter presents the aims and objectives of the research. This section provides a clear vision of what the research seeks to achieve, detailing specific goals and intended outcomes. Following the aims and objectives, the unique contribution of the research is illustrated. This highlights the novel aspects of the study and how it adds to the existing body of knowledge in the field.

Lastly, the chapter concludes with an outline that presents a road map for the remaining sections of the study. This gives an overview of what each subsequent chapter will cover, providing a comprehensive guide to the structure of the research. In summary, this chapter sets the stage for the rest of the research, establishing the groundwork upon which the study is built and providing a clear path for the reader to follow.

1.2 Background

The rapid growth of technological advancement, with smart cities, autonomous vehicles, and artificial intelligence has transformed tremendously with its realisation clearly hinging on data availability. However, as information is involved, possible data breaches regarding the data collected and its usage have grown to attract several interests. The regulatory bodies are still having a field day regarding creating regulations on data protection acts. Moreover, it is

believed that regulations alone are insufficient because they do not protect people's data from cyberattacks. According to Hickey (2012), vice president of software security firm Vincula, current cyber-attacks are more common acts of espionage (gaining unauthorised access to a system for information gathering) than sabotage (actively compromising a system's regular operation). While images and video data are essential for the operation of these technologies, they would be a significant source of threat if misused. Recent advances in technology have made manipulations of images and videos (deepfake) very easy and highly accessible using application software, selling users' data and information without their consent (Guera and Delp, 2019).

Deepfake is a state-of-the-art application of deep learning techniques for the manipulations of image or video data (Dolhansky et al., 2019; Masood et al., 2023). The advent of deepfake technology marks a significant milestone in the evolution of artificial intelligence (AI) and digital media manipulation. Coined from the combination of "deep learning" and "fake," deepfakes refer to synthetic media generated using deep neural networks, particularly generative adversarial networks (GANs) and autoencoders (AE). These algorithms enable the seamless alteration of audiovisual content, including videos, images, and audio recordings, with astonishing realism and accuracy. While this technology is usually misused in different cases, it also holds positive potential, which may help give live feed data an extra layer of security, and sometimes are deployed for entertainments. The manipulation and fabrication of digital images and videos are not new. Nonetheless, the proliferation of deepfake with numerous applications built using deep neural networks has made deepfake a significant concern when wrongfully deployed (Dolhansky et al., 2019).

Deepfakes typically involve the intrinsic association of expertly crafted identities that could be used to create an impression of being present at an event that never happened or being absent from an actual event. Several deepfake algorithms have been proposed, and they are mainly based on modern machine learning techniques, usually involving the training of a large set of data sets (Dolhansky *et al.*, 2019, 2020; Nowroozi *et al.*, 2021). Initially, deepfake technology garnered attention for its novelty and entertainment value, often employed in creating amusing videos and impersonations. However, its potential for misuse quickly became apparent, leading to widespread concerns regarding its implications for privacy, security, and trust in digital media. Over time, deepfake applications have expanded beyond mere mimicry to encompass a

spectrum of uses, including political manipulation, celebrity impersonation, and fraudulent activities such as financial scams and revenge porn.

The proliferation of deepfake technology has raised numerous ethical and societal concerns, stemming from its ability to fabricate convincing yet entirely fictitious content. From the manipulation of public figures' speeches to the creation of counterfeit evidence for legal proceedings, deepfakes pose a threat to the integrity of information and the authenticity of digital communication. Moreover, the ease of access to deepfake tools and resources has democratized the creation of synthetic media, amplifying the potential for malicious actors to exploit and manipulate public perception for nefarious purposes. The rapid advancement of deepfake technology has outpaced the development of regulatory frameworks and legal mechanisms to address its challenges adequately. While some jurisdictions have introduced legislation targeting specific forms of deepfake manipulation, such as non-consensual pornography or election interference, the broader implications for privacy and data protection remain largely uncharted territory. As such, there is a pressing need for comprehensive legislative measures and policy interventions to safeguard individuals' rights and mitigate the risks associated with deepfake manipulation.

Despite the growing awareness of deepfake-related threats, there exists a notable gap in research focusing on proactive approaches to mitigate these risks while harnessing the potential benefits of synthetic media. While much attention has been devoted to detecting and debunking deepfakes post hoc, relatively fewer efforts have been directed towards leveraging deepfake technology for constructive purposes, such as privacy preservation and data protection. This research seeks to address this gap by exploring the real-time application of deepfake techniques for de-identification, offering a novel approach to enhancing privacy and security in the digital age.

1.3 Problem Statement

While deepfake technology holds promise for de-identification purposes, enabling the anonymization of individuals' identities within multimedia content, the lack of robust and real-time methodologies poses a significant challenge. Existing approaches often suffer from limitations in scalability, computational efficiency, and preservation of visual realism, hindering their practical applicability in scenarios requiring timely and reliable deidentification. Furthermore, the potential for adversarial attacks and the ethical implications

of deploying deepfake-based de-identification techniques raise concerns regarding their effectiveness, reliability, and ethical considerations. Consequently, there is a pressing need to develop innovative and efficient deepfake frameworks tailored specifically for de-identification purposes, addressing these challenges and facilitating the widespread adoption of privacy-preserving technologies in the digital landscape.

Therefore, this study provides an option for Deepfake as another De-identification Technique and Application in real-time to aid in privacy preservation and data protection of the face data and answer several research questions arising from the identified research gaps.

1.4 Research Questions

Addressing the issues raised under the problem statement would require identifying the research questions. The main research question is how people's faces could be obfuscated in real-time to preserve privacy and improve data Protection. The specific research questions are:

- What are the latest techniques, models, or algorithms for deepfake?
- How are the parameters of such models or algorithms selected for the best performance?
- How does Deepfake perform as a de-identification technique compared to the standard de-identification techniques?
- How can a deepfake model be deployed to perform in real-time?

1.5 Justification, Aims and Objectives

This work aims to encourage privacy preservation and promote data protection. The proposed models of real-time application of deepfake as a de-identification technique will heighten privacy preservation and data protection. Hence, the research objectives are:

- Develop and evaluate a real-time deepfake technology for privacy preservation and data protection using first order motion and the mediapipe models.
- Incorporate a privacy consideration through masking .
- Investigate the impact of various model characteristics on deepfake performance.
- Assess the feasibility and practical implications of real-time deepfake technology on single and multiples faces on a single image.

1.6 Contributions

The Main Contributions of the thesis are as follows:

1. A framework for real-time application of deepfake for de-identification, privacy preservation, and promoting data protection compliance is presented and developed.
2. Real-time applications of deepfake models were developed. It consists of a generator (that creates fake data) and a discriminator (that differentiates the real from the fake data).
3. Through the real-time application of the models, deepfake learns more features and produces better performance by training the target face with multiple images and videos rather than a single image.

1.7 Thesis Outline

The thesis is structured into six chapters aimed at comprehensively addressing the research objectives. Chapter 1 provides an introductory overview, giving the research background, questions, justification, aims, and objectives, concluding with an outline of the thesis. Chapter 2 gives an extensive literature review, analysing some previous studies on facial recognition, general data protection regulations, Deepfake applications, de-identification methods, and potential industries for deployment. In Chapter 3, the methodology adopted for the research, detailing the framework, models deployed, and datasets chosen, is presented. Subsequently, Chapter 4 delineates the step-by-step experiments conducted with three models, explaining their outcomes. In Chapter 5, the results of the deployed models are scrutinized, encompassing comparisons and discussions on their performance, and encountered challenges, thereby validating the privacy preservation approach. Finally, Chapter 6 encapsulates the thesis with a succinct summary of the main findings, challenges encountered, and prospects for future research directions

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter presents a comprehensive examination of the literature on Face Detection and Recognition systems, De-identification techniques, and advancements in deepfake technologies. Additionally, it explores various industries poised for application of the proposed model, including autonomous vehicles, healthcare, and Unmanned Aerial Vehicles (UAVs). Furthermore, this review also presents the various machine learning methodologies and their diverse applications within the contexts of the research. Through an in-depth reviews and analysis, this chapter aims to expose the current landscape of these critical domains and pave the way for further investigation and innovation in the field.

2.2 Overview of Deepfake Technology

Deepfake technology refers to the creation of synthetic media, primarily through the use of deep learning algorithms, to manipulate or replace existing content with fabricated elements (Kirchengast, 2020; Masood et al., 2023). The term "deepfake" originates from the combination of "deep learning" and "fake." Deep learning, a subset of artificial intelligence, involves the training of neural networks with vast amounts of data to perform specific tasks, such as image or video generation, with remarkable accuracy (Mahmud & Sharmin, 2021) . Therefore, a deepfake is an AI-based tool that can create or manipulate image or video content to depict something that did not happen. Celebrity faces appearing on other people's bodies and politicians appearing to utter things they never said are the most common examples of deepfake on the Internet (Kirchengast, 2020). Even though digital editing technologies such as Photoshop, avatar creation software, and facial animation software have been around for a while, this more widely used - deepfake production – has grown simpler, cheaper, and faster to make (Wojewidka, 2020). Most of the deepfake manipulation has taken place in entertainment, such as merging human and dog faces or politicians speaking bizarre things on late-night television. However, a quickly rising dark side generates recordings of leaders and influencers for criminal purposes, raising legitimate concerns about how this affects individuals, organisations, and society. Figure 2.1 shows a sample deepfake image and its original image.

Figure 2 - 1 Images showing the original image, the target image and the output image with deepfake.

2.3 Key Components of Deepfake Technologies

Deepfake technologies have gained significant attention in recent years due to their potential to produce highly realistic yet entirely fictional content. These technologies rely on advanced machine learning techniques, particularly generative adversarial networks (GANs) and autoencoders, to generate and manipulate multimedia content seamlessly (Khoo et al., 2022).

a. **Generative Adversarial Networks**: GANs consist of two neural networks, the generator and the discriminator, which compete against each other in a game-like scenario. The generator generates synthetic content (e.g., images, videos), while the discriminator evaluates the authenticity of the generated content. Through iterative training, GANs can produce increasingly convincing deepfake media.
b. **Autoencoders**: Autoencoders are neural networks designed for dimensionality reduction and data compression. In the context of deepfakes, autoencoders can encode and decode images or videos, enabling the generation of altered or entirely new content based on the learned representations.

2.4 Applications, Challenges and Ethical Issues of Deepfake Technologies

Deepfakes have found applications in various fields but represent a double-edged sword with both beneficial applications and significant risks. Understanding their capabilities, limitations, and ethical implications is essential to help derive the maximum benefits therein.

2.4.1 Applications of Deepfake

The following are some applications of deepfake (Kwok & Koh, 2021; Masood et al., 2023; Westerlund, 2019):

a. **Entertainment:** Deepfake technologies are widely used in the entertainment industry for creating digital doubles of actors or inserting them into scenes where they were not originally present. This application has implications for filmmaking, gaming, and virtual reality.

b. **Social Media Content Creation:** The application of deepfake technology for social media content creation has seen rapid growth, offering users novel ways to engage with digital platforms. By seamlessly blending facial reenactment, voice synthesis, and image manipulation, deepfake tools enable users to produce highly realistic videos, memes, and other multimedia contents. Social media influencers and content creators leverage deepfake technology to enhance storytelling, create entertaining skits, and mimic popular personalities, amplifying their reach and engagement with audiences. Additionally, deepfake-based filters and effects add an element of fun and creativity to user-generated contents, fostering community interaction and user-generated memes. However, the proliferation of deepfake content on social media also raises concerns about misinformation, privacy infringement, and digital manipulation, underscoring the need for robust detection mechanisms and ethical guidelines to safeguard against potential harm.

c. **Political Manipulation:** Deepfake videos have been used to create misleading or false political contents, such as speeches or interviews, with the potential to influence public opinion or sow discord. This poses significant challenges for media authenticity and trust.

d. **Identity Theft and Fraud:** Deepfake technologies raise concerns regarding identity theft and fraud, as they can be used to create convincing forgeries of individuals in compromising or illegal situations. This has implications for cybersecurity and digital identity verification.

e. **Privacy Preservation:** Despite their negative connotations, deepfake technologies also hold promise for privacy preservation and data protection. By anonymizing or obfuscating individuals' identities in multimedia content, deepfakes can be used for de-identification purposes in sensitive contexts, such as surveillance footage or medical imaging.

2.4.2 Challenges and Ethical Considerations of Deepfake

Despite their potential benefits, deepfake technologies also present several challenges and ethical considerations:

a. **Misuse and Misinformation:** The widespread availability of deepfake tools raises concerns about their misuse for spreading misinformation, defamation, or harassment.
b. **Detection and Authentication:** Detecting deepfake content and authenticating multimedia sources are ongoing challenges, requiring advancements in forensic analysis and verification techniques.
c. **Consent and Privacy:** The creation and dissemination of deepfake content without individuals' consent raise serious privacy concerns and may infringe upon their rights to control their own image and likeness.
d. **Bias and Representation:** Deepfake technologies may perpetuate biases and stereotypes, particularly if used to manipulate or misrepresent individuals from marginalized or underrepresented groups.

With advancement in technologies, the deployment of deepfake will only continue to bather momentum. For security and privacy, various techniques have nee explored in the detection of deepfake, but before the discussion on the detection techniques, there is the need to understand how deepfake is created.

2.5 Creating Deepfake

Creating deepfake content involves a multi-step process that combines several stages basically involving extraction of information from the original file through the other stages of the processing. This is discussed in the next subsections. Creating deepfake is a process involving several stages such as face detection, alignment, embedding, generation.

2.5.1 Face Detection

Face detection in the context of deepfake refers to the process of identifying and locating human faces in a digital image or video frame. This is a crucial initial step in creating deepfake as it allows the system to focus specifically on the facial region for subsequent manipulations (Viola & Jones, 2001). Face detection algorithms, such as the Viola-Jones algorithm or deep learning-based methods like Multi-task Cascaded Convolutional Networks (MTCNN), are typically used to locate faces within an image or video frame. These techniques are designed

to recognize patterns consistent with human faces, considering factors like the relative position and size of eyes, nose, mouth, and other facial features (Xiong & De La Torre, 2013). Once the faces have been detected and located, the region of the image containing each face is often normalized and aligned to a standard format, typically by cropping and rotating the image so that the face is centred and upright. This facilitates the subsequent stages of the deepfake process, where the facial features are manipulated or swapped (Thies et al., 2020).

In the context of deepfake, accurate face detection is vital. It allows for the precise manipulation of facial features and contributes to the overall realism of the final output. However, it also adds to the challenge of controlling and detecting misuse of this technology (Thies et al., 2020).

2.5.2 Face Alignment

Face alignment in the context of deepfake refers to the process of standardizing images of faces based on size, position, and orientation so that facial features align to predefined locations on a standard template. This step is crucial to ensure that faces can be accurately manipulated and swapped in deepfake creation (Xiong & De La Torre, 2013). The goal of face alignment is to transform the detected facial region in an image or video frame to a canonical pose, where key facial landmarks (such as the eyes, nose, and mouth) are located at predetermined positions. This is usually done by identifying specific facial landmarks in the original image, such as the corners of the eyes and the mouth, and applying a geometric transformation (such as scaling, rotation, or translation) to align these landmarks with the ones on a standard face template (Thies et al., 2020).

A common method of face alignment involves using an algorithm like Active Appearance Model (AAM) or Supervised Descent Method (SDM), but more recent approaches leverage deep learning models to achieve even more accurate results. Face alignment plays a crucial role in the deepfake process, as it ensures that the manipulated face will have the correct size, position, and orientation in the output video. Moreover, it allows the deep learning model to focus on learning the appearance of faces rather than variations due to pose or lighting (Thies et al., 2020).

2.5.3 Face Embedding

Face embedding in the context of deepfake refers to the process of representing a face in an image or video frame as a compact, high-dimensional vector in an abstract feature space. This vector, or "embedding," captures the essential characteristics of the face and allows for comparisons between different faces (Parkhi et al., 2015). The goal of face embedding is to capture a representation of a face that retains the most important information while filtering out irrelevant variations due to lighting, pose, and other non-essential factors. This is typically done using a deep learning model that has been trained to distinguish between many different faces (Thies et al., 2020).

A common method for generating face embeddings is to use a type of neural network called a convolutional neural network (CNN). The network is trained on a large dataset of face images, using a training process that encourages the network to create embeddings where faces of the same person are close together (in terms of Euclidean distance or some other distance metric), while faces of different people are further apart (Schroff et al., 2015). In the context of deepfake, face embeddings can be used to guide the generation process. For instance, the deepfake model can be trained to transform the embedding of a source face to match the embedding of a target face, thereby ensuring that the resulting fake face has the same identity as the target (Schroff et al., 2015).

2.5.4 Face Generation (GAN)

A variety of approaches can be used to create synthetic audio-visual media. (Verdoliva, 2020) (Tolosana et al., 2020) have recently proposed an outline of Media Forensics with a particular focus on Deepfakes. Due to its diverse uses and realistic results, the Generative Adversarial Network (GAN) is currently the most popular of these techniques. (I. J. Goodfellow et al., n.d.) was the first to introduce generative adversarial networks (GANs) in 2014. They present a new framework for estimating generative models based on a malicious process in which two models are trained at the same time: a discriminative model D capable of assessing the likelihood that a sample came from the training data rather than G, and a generative model G that reflects the data distribution. The goal of G's training is to increase the chances of D making a mistake. This framework is equivalent to a two-player min-max game. In the instance of Deepfake, the G represents a group of counterfeiters attempting to create counterfeit currency, while the D represents the police trying to discover the criminal activity. G and D can be made using any

generative model; however, deep neural networks were used in the original form. (Guarnera et al., 2020) The deepfake generation process is as shown in Figure 2.2.

Initially, the generator pulls data from the latent space at random and attempts to create fake images that look like genuine images in the training dataset. After then, the discriminator will try to distinguish between actual and fraudulent photos. As both models compete, improvements are achieved simultaneously, leading to convergence or equilibrium. The generator improves to the point that the discriminator has difficulties distinguishing between actual and fake images (see Brock et al., 2018).

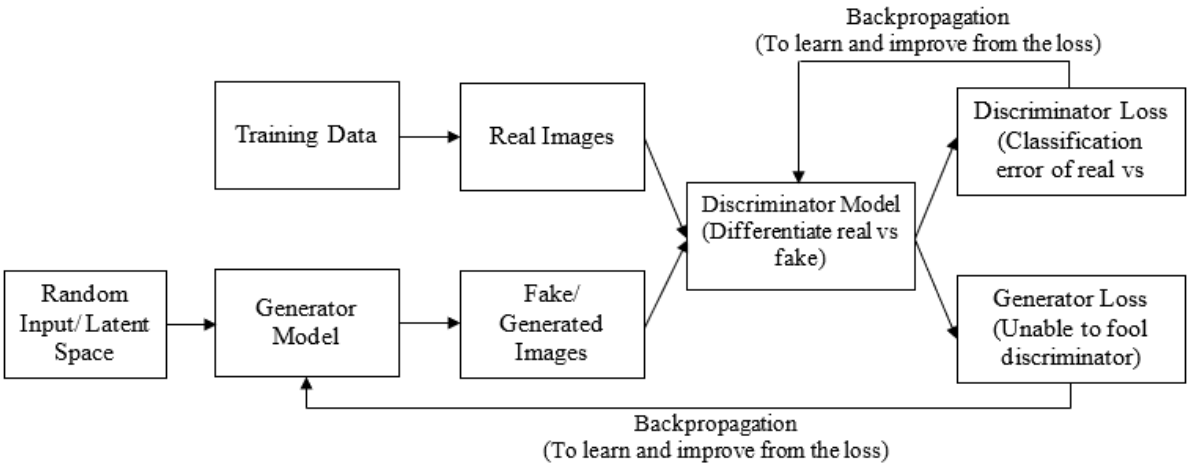


Figure 2 - 2 shows how the generative adversary network works in creating a deepfake(Brock et al., 2018).

2.5.5 Face Blending

Face blending in the context of deepfake refers to the process of seamlessly integrating the generated synthetic face with the original image or video frame. This is a crucial final step in the deepfake creation process that ensures the replaced face does not appear artificially superimposed but naturally fits into the scene (Pérez et al., 2003). Face blending involves adjusting the synthetic face generated by a machine learning model (like a Generative Adversarial Network) to match the lighting, colour, and texture of the surrounding area in the original image or video frame. It also includes smoothing the boundaries between the inserted face and the rest of the image, so that there are no visible seams or sharp transitions that could give away the manipulation (Thies et al., 2020).

Techniques used for face blending may include Poisson blending, a popular method for image editing that works by solving a certain type of differential equation (Pérez et al., 2003), or more advanced neural network-based methods that can learn to blend the faces in a way that is

visually indistinguishable from a real image. Face blending is a crucial aspect of deepfake creation because even the most realistic synthetic face can look artificial if it is not properly blended with the rest of the image. At the same time, the quality of the blending is often a giveaway in deepfake detection, as errors in this step can lead to visible artefacts (Thies et al., 2020).

2.5.6 Output

The output in deepfake refers to the final product generated by the deepfake creation process, typically in the form of a static image or video sequence where one or more person's face has been convincingly replaced with another's (Thies et al., 2020). The output of a deepfake process involves a synthetic representation of an individual, usually in video form, that realistically mimics their appearance and mannerisms. The degree of realism in the output can be highly variable, depending on the sophistication of the deepfake model and the quality and quantity of input data (Rossler et al., 2019).

The quality of a deepfake output is determined by a variety of factors: the realism of the generated face, the accuracy of the face replacement, the seamless integration of the replaced face into the original video (face blending), and the consistency of the replaced face's movements, lighting, and texture with the rest of the scene (Rossler et al., 2019). High-quality deepfake outputs are often incredibly convincing and can be difficult to distinguish from authentic videos. However, they may also exhibit certain tell-tale signs of manipulation, such as unusual lighting or colour effects, inconsistent facial movements, or blurry or flickering areas around the replaced face (Thies et al., 2020). While deepfake have found legitimate use in areas like entertainment, advertising, and synthetic data generation, the potential misuse of this technology raises serious ethical and legal concerns, particularly relating to issues like non-consensual pornography, identity theft, and the spreading of disinformation (Thies et al., 2020).

2.6 Face Detection and Recognition

According to (Othman & Aydin, 2018), "Face recognition is defined as comparing two faces and storing known people in a database to classify them as known or unknown. This can be accomplished by comparing invariant features obtained via algorithms that capture the delegate variability of the faces of the structure." The Face can reflect a person’s emotions, making it

an essential part of a person's body. The nose, eyes, and mouth, linked to facial expression, are the most significant components of the face image. (Ojala et al., 2002).

Face Detection and Recognition systems are commonly categorized into two groups: Face Detection and Face Recognition (Jafri & Arabnia, 2009). Various methods exist for implementing face detection, including Fisher-face, Eigen-face, and Haar-like features, which are followed by a geometric analysis of facial features such as the eyes, mouth, and nose. In terms of face recognition, one prominent technique is the Local Binary Pattern (LBP) method, which was developed by Ojala et al. (1996). The identification and recognition process typically involve several steps, as illustrated in Figure 2.3.

LPB describes the shape and texture of a digital image. This technique provides good results and is efficient for real-time applications. Haar-like features and LBP are robust when compared to the others. According to many studies (Huang et al., 2019; Ahonen et al., 2004), the LBP approach was used for face recognition because of its quick discriminatory performance and good results. LBP generates the binary code that describes the local texture pattern. The nose and eyes are extracted from the LBP face image, and LBP histograms are created for each image pixel. (Ojala et al., 2002) (Ahonen et al., 2006).

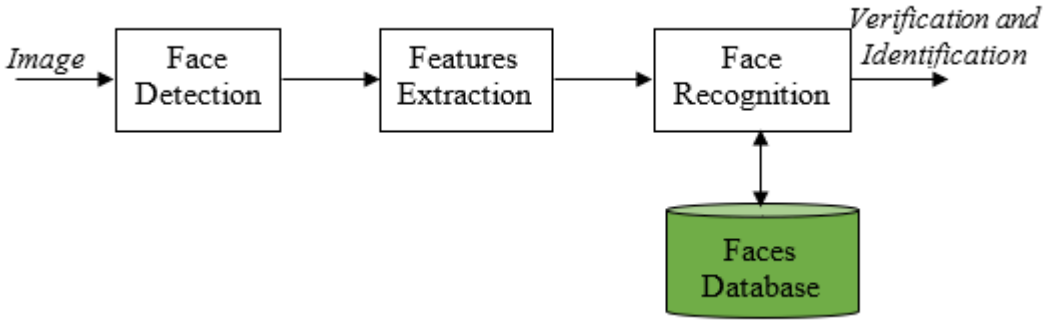


Figure 2 - 3 General steps for identification and recognition procedure(Ojala et al., 2002)

2.6.1 Face Representation

The first phase, face representation, describes how to display a face and determines which methods can be used for detection. Face detection is achieved using the Haar-like feature and the AdaBoost classifier (Sun et al., 2016). The brightness, redundant data, contrast, and image size of an acquired image are some of the factors that affect a face recognition system's

precision. As a result, preparing the captured image is much more critical than face recognition. The input face is converted to grayscale and normalized for this purpose.

2.6.2 Feature Extraction

A face image's most valuable and unique aspects are extracted during the feature extraction step. The face image will be compared to the images from the shop to gain features. Face recognition systems are built around the feature extraction process. These attributes can be used to sample the face and measure image matching. (Othman & Aydin, 2018). Face recognition algorithms exist to extract and bring out the most important features from photos of faces. LBP is a programming language that may describe the shape and texture of digital images. This method is effective for real-time applications and produces good results. When contrasted with the others, Haar-like characteristics and LBP stand out. Partitioning a picture into several minor parts is how the LBP process is done, as shown in Figure 2.4. Each region's features are retrieved separately. These attributes are coded into binary patterns to depict the surroundings of pixels in the areas. To calculate the characteristics, each region is processed. All estimated features are integrated into a single feature histogram, which visually represents the image.

LBP works on grayscale pictures, with the 3∗3 neighbourhoods of each pixel interested in the centre value and considering the result as a binary number. The texture description is then the histogram of these 28 = 256 individual labels. Unsupervised texture segmentation performance was excellent when this operator was used with a primary local contrast measure (Ojala et al., 1996). Following that, various related texture and colour texture segmentation algorithms were developed. The LBP operator was expanded to include different-sized neighbourhoods (Ojala et al., 2002). Any radius and number of pixels in the neighbourhood can be achieved using a circular neighbourhood and bilinearly interpolating values at non-integer pixel locations. The local neighbourhood’s grayscale variance might be an additional contrast metric. The LBP code is computed for the middle pixel with coordinates (x, y) v.

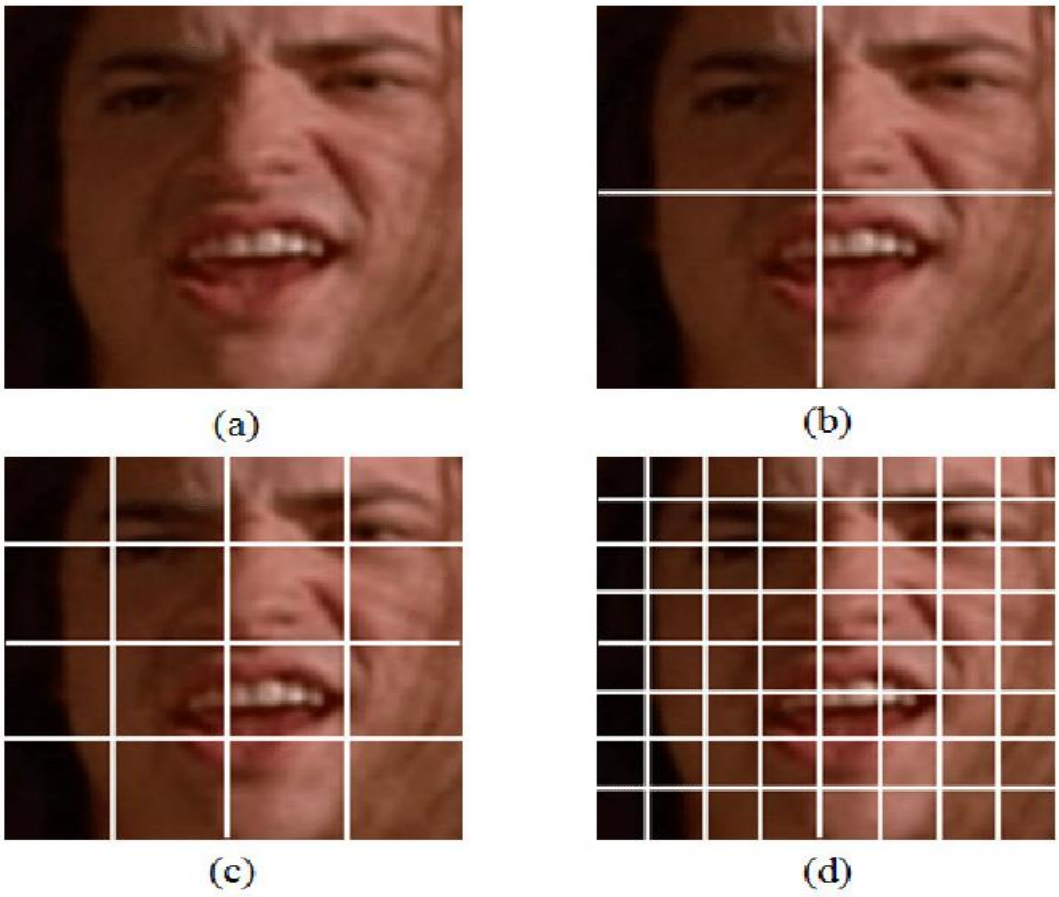


Figure 2 - 4 A face image is divided into different areas neighbourhoods (Ojala et al., 2002).

Figure 2.4 above is the LBP value. In the image's window, there are eight surrounding pixels. To calculate the LBP value for a pixel in a grayscale image, the pixel is compared to its eight neighbours on the left-middle, left-top, left-bottom, right-bottom, and right-top axes. When the value of the centre pixel is greater than the value of the neighbouring pixel, write zero. Something else; come up with one. This generates a binary number with eight digits. As a decimal number, a binary value is generally kept in the centre pixel location of the output picture. Figure 2.6 illustrates the entire procedure. The present pixel, for example, has a value of 157. The comparison begins with the pixel next to it, whose label is zero. The neighbouring pixel with label zero has a value of 150. Because it is less than the current pixel value of 157, the 0th-bit position in the 8-bit binary array will be zero. The process will then be performed in reverse. The following label position one value is 165, which is greater than the current pixel value. As a result, the array's first-bit position is set to one. If the nearby pixel's value is the same as the current value, write 1.

Local features are formed in the local feature extraction procedure by generating histograms of LBP across local picture areas once the LBP operator has been calculated. The feature vector of the picture can be produced by computing the LBP histogram for each cell after the LBP mask has been calculated for each pixel. As a 256-dimensional feature vector, this histogram can be visualised. After that, the LBP histogram will be standardised, and the concatenated histograms of each cell will be completed. This provides an image-wide feature vector. For a single cell, the length of the feature vector decreases from 256 to 59. The image of a person's face is divided into 64 pieces.

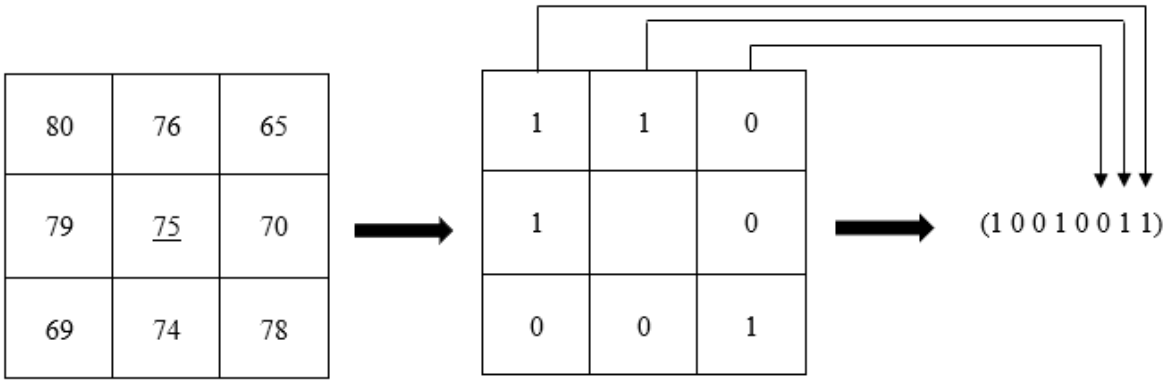


Figure 2 - 5 Example of LBP calculation

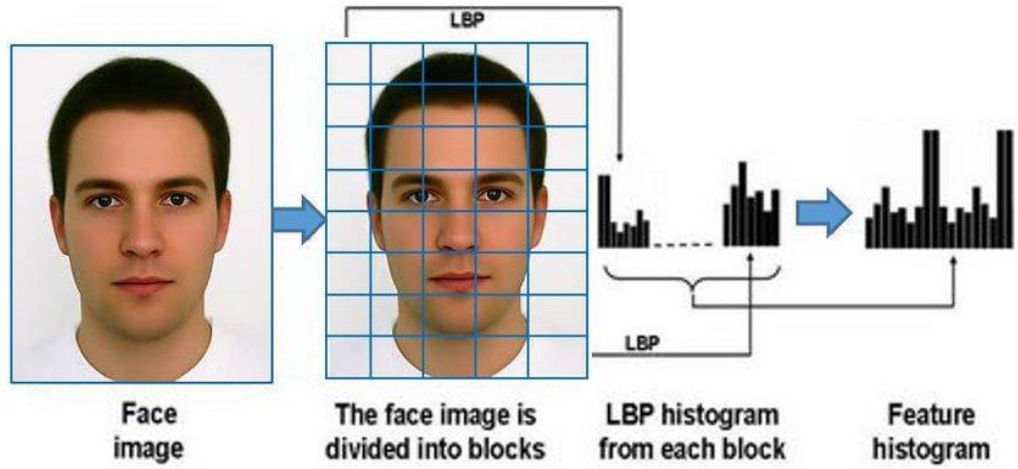


Figure 2 - 6 An example of extracting LBPH feature vector (Abuzneid et al 2018).

For each part, a histogram with all potential labels is created. This means that each bin in a histogram represents a pattern and stores the number of times it appears in the section. The feature vector is then created by joining the regional histograms into a single large histogram. Finally, estimating the distance between the histograms is used to determine the match between

the photos. Every pixel's LBP code is aggregated into a histogram, similar to a texture description for an input picture. The histogram of the picture LBP (x, y) can be characterised as:

$$LBPH(i) = \sum x, yI\{i, LBP(x,\ y)\}, i = 0, \dots, n-1 \quad (1)$$

2.6.3 Classifier

The face image will be compared to the store images in the classification stage to determine the features. The local features gathered in the algorithm's initial phase are compared. The output of the classification portion will be the maximum matching score based on the face image. Following the LBPH feature vector extraction, face recognition is performed using the K-Nearest Neighbour (KNN) classifier based on histogram matching algorithms. For example, the commonly used Chi-square measure:

$$d_{x2}(M, S) = \sum i = 1B(M_i - S_i)2M_i + S_i \quad (2)$$

M and S are the galleries in (3), which explore histogram objects. The number of bins in the histogram is denoted by B. The diversity among the feature vectors must be quantified to compare two facial photos, a sample (S) and a model (M). Histogram features can be used to do this. After the Chi-square algorithm calculates the distance between elements, the KNN classifier is used to identify faces. In a face recognition system, KNN is a data categorization approach that can be used. The schematic diagram of the nearest neighbour is shown in Figure 2.7 below.

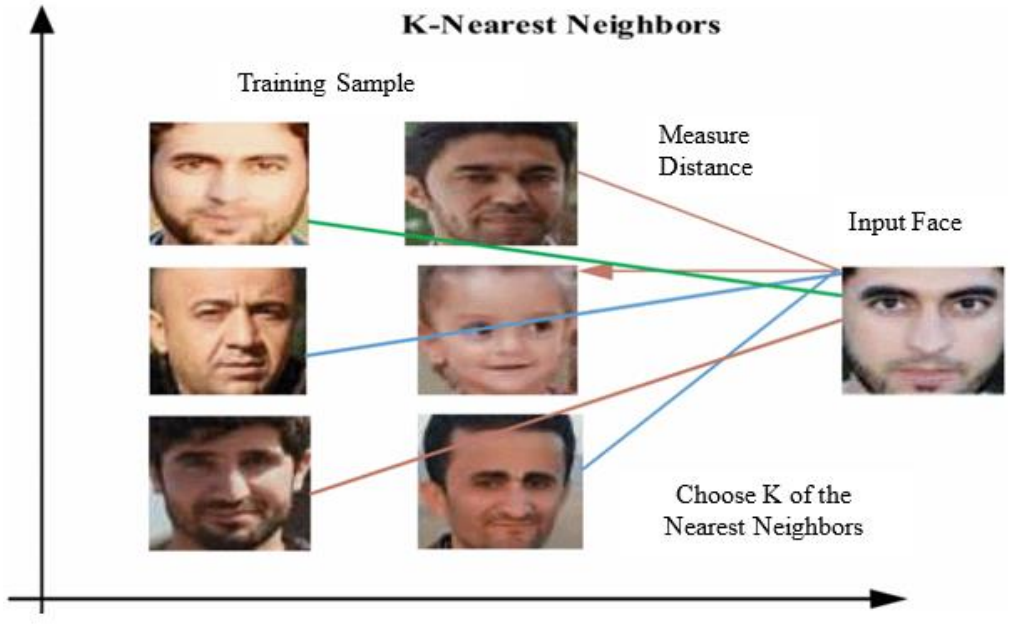


Figure 2 - 7 Graphic diagram of KNN classification

In KNN, each training data sample will be compared to the input face. The nearest neighbour technique refers to the selection of unique looks. It works on the principle that the query case should be close to the training examples. Every pixel on the front represents precise information.

2.7 The Rise of Facial Recognition Surveillance in The World

Biometric surveillance, particularly facial recognition technology, is on the rise in several parts of the world. According to an Amnesty International investigation, at least 64 countries around the world are currently utilising facial recognition technologies. (Feldstein, 2019) China is one of the most prominent adopters of the technology. Face recognition is used in Chinese schools to track library borrowing and produce annual nutrition reports for each student. (Emotional Entanglement: China's Emotion Recognition Market and Its Implications for Human Rights, 2021) According to reports, Chinese authorities use biometric identification, including facial recognition technology, to restrict the Uyghur minority's movements and activities. (Parliament, n.d.) Chinese businesses are taking the lead in establishing international technical standards for AI applications, such as facial recognition, at the International Telecommunication Union. (Chinese Tech Groups Shaping UN Facial Recognition Standards Financial Times, n.d.).

Face recognition cameras are increasingly being used in public settings in a variety of countries and regions throughout the world, including Kyrgyzstan, India, Latin America, Israel, the

United States, Australia, and Russia. (Parliament, n.d.) According to reports, AI-assisted surveillance methods are increasingly being used against political dissidents and human rights activists in Russia, and the pandemic has accelerated the installation of a network of 100,000 facial recognition cameras to track quarantined people. (Madiega & Mildebrath, n.d.) In light of this, policymakers throughout the world are debating the feasibility of enacting more or less stringent legislative frameworks to regulate the use of facial recognition technology.

2.7.1 Legislation Around Facial Recognition

Every country and government body has its own data protection and privacy policies in place as a regulatory body to regulate data protection, data sharing, and data privacy. However, with facial recognition systems becoming so sophisticated and advanced, governments are concerned about their application and people's privacy. An article was published in the House of Parliament on Wednesday, March 31, 2021, titled "Facial recognition technology: police powers and the Protection of Privacy." In the article, it was discussed that in August 2020, the Court of Appeal found that the use of facial recognition technology by the South Wales Police had been unlawful. The court stated that there was no clear guidance on its use of the system, and the South Wales police did not take reasonable steps. (Facial Recognition Technology: Police Powers and the Protection of Privacy, House of Lords Library, n.d.).

Moreover, at the time of publication, it was stated on April 12, 2021, that the use of facial recognition technology will be the subject of the following oral question in the House of Lords: "Lord Clement-Jones (Liberal Democrat) to ask Her Majesty's Government what assessment they have made of (1) the Council of Europe consultative committee's Guidelines on Facial Recognition, published on January 28; and (2) the Biometrics and Forensics Ethics Group's Briefing Note on the Ethical Issues Arising from Public-Private Collaboration in the Use of Live Facial Recognition Technology, published on January 21; and what (a) legislative or (b) regulatory changes they intend to make as a result".

Furthermore, The European Council published an in-depth analysis of "Regulating Facial Recognition in the EU". The document consists of discussions on Facial recognition and artificial intelligence technologies, usage, and economics. Concerns are raised by facial recognition, the current EU legal framework, the proposed EU Artificial Intelligence Act, facial recognition, and international aspects.

2.7.2 Concerns Raised on Facial Recognition

The prevalence of facial recognition technology and the difficulty of implementing human control are significant sources of concern. Facial recognition technology records features of the human body that cannot be changed (unlike mobile phone identifiers), a large number of images are already available (for example, on the internet), and facial images can be captured remotely without a person's knowledge while obtaining consent in public spaces is extremely difficult. (Castelluccia et al., n.d.) Furthermore, the application of deep learning algorithms allows for the collection of extremely sensitive information about a vast number of people. It makes manual verification and labelling nearly impossible as data sets grow. (This Is How We Lost Control of Our Faces | MIT Technology Review, n.d.) Moreover, Security risks the risks posed by collecting and keeping facial recognition data, including the danger of data breach and misuse, have been 30mphasized. (Algorithmic Bias Detection and Mitigation: Best Practices and Policies to Reduce Consumer Harms, n.d.).

2.7.3 Data Protection and Privacy

Face recognition technologies entail gathering, comparing, and storing facial photographs for identification. Face recognition technologies enabled by AI, particularly' second wave' biometrics, employ more complex technologies and algorithms, collecting sensitive and personal data. (Study Supporting the Impact Assessment of the AI Regulation | Shaping Europe's Digital Future, n.d.) More data, including personal data, is constantly collected, and analysed by devices (e.g. surveillance cameras or autonomous vehicles) as AI technology (e.g. facial recognition) improves, resulting in more invasive effects on individual privacy and data protection. ("Artificial Intelligence in Society," 2019) The difficulty in obtaining explicit agreement for Facial recognition systems is one of the major sources of concern. It has been reported that several vendors have been accused of stealing publicly available facial pictures from other websites to populate their biometric databases. (REGULATING BIOMETRICS Global Approaches and Urgent Questions, 2020)(Madiega & Mildebrath, n.d.) Even researchers in facial recognition systems have gradually stopped seeking people's permission. (This Is How We Lost Control of Our Faces | MIT Technology Review, n.d.).

2.7.4 Mass Surveillance and Concerns for Fundamental Rights

There have also been concerns raised about the potential for the use of facial recognition technologies to become more widespread. In the medium or long term, the possibility of expanding the use of facial recognition technologies outside their initially authorised and controlled purpose implies some concerns. Such extensions can occur, for example, by using data collected on social networks or databases created for different purposes, by using a database for purposes other than those intended, or by adding new functionalities to an existing system (for example, by extending facial recognition used for passport control to payments in an airport and then throughout the city). (Castelluccia et al., n.d.) It has been suggested that such an extension could be part of a planned strategy by proponents to employ facial recognition systems in circumstances where the goal appears legitimate at first, and then progressively expand their use (i.e. the "slippery slope" argument). (Madiega & Mildebrath, n.d.).

It is stated in the European Commission, Impact assessment accompanying the Proposal for a Regulation of the European Parliament and of the Council, 2021; Remote biometric identification technologies are increasingly being used in publicly accessible locations, and face recognition appears to be quickly becoming the norm in the EU. Investigations by the European Commission demonstrate that whenever such a system is in use, the whereabouts of people in the reference database can be tracked, jeopardising their personal data, autonomy, and dignity. As a result, a new set of societal issues like the inability to move anonymously in public places or conformism that are detrimental to one's freedom of choice could emerge as a result of the use of face recognition technology to create such a mass monitoring system. The Italian Data Protection Authority stated that the automated processing of biometric data for facial recognition might be considered indiscriminate mass monitoring in this regard. (Use of Facial Recognition Technology for Migrant Disembarkation in Italy, n.d.).

Furthermore, the use of Facial Recognition systems raises concerns about a variety of other civil liberties, including religious freedoms and children's rights – as vulnerable individuals deserving of greater protection, particularly when employed for law enforcement and border management objectives. (Fundamental Rights Report 2020 | European Union Agency for Fundamental Rights, n.d.) It has also been emphasised that employing face recognition technologies to process facial photographs acquired by video cameras in public places may

infringe on a person's freedom of expression and assembly, as well as hurt their freedom of association and assembly. (Fundamental Rights Report 2020 | European Union Agency for Fundamental Rights, n.d.; Madiega & Mildebrath, n.d.).

According to a United Nations Human Rights Council report, the use of face recognition technology to identify people in public places has significant negative consequences for the rights to privacy, freedom of speech, and peaceful assembly. (Statement by the Special Rapporteur on the Right to Adequate Housing, Leilani Farha, during the Interactive Dialogue at the Human Rights Council | OHCHR, n.d.). Furthermore, automatic person identification and tracking may have a significant impact on people's social and psychological behaviour, raising substantial ethical concerns about the usage of such technology. (Ethics Guidelines for Trustworthy AI | Shaping Europe's Digital Future, n.d.) Face surveillance infrastructures have the potential to exacerbate the effect, resulting in the formation of entrenched structural racism and the threat of modern democratic forms or life social solidarity (Madiega & Mildebrath, n.d.).

2.7.5 Respect For Private Life and Personal Data Protection

Because the use of Facial Recognition systems entails the processing of data to identify, it is a violation of the right to data protection enshrined in Article 8 CFR, as well as the right to privacy included in Article 7 CFR. The initial video capture, the subsequent retention of the material, and the comparison of footage with database records for identification (matching) are all examples of interferences with or limitations on this right. According to Article 52(1) CFR, any restriction on these fundamental rights must be strictly necessary and proportional. (Article 8 - Protection of Personal Data | European Union Agency for Fundamental Rights, n.d.).

However, these essential rights are still gaining shape in practice. (The EU Rights to Privacy and Personal Data Protection: 20 Years in 10 Questions — Vrije Universiteit Brussel, n.d.) and the scope of their application to private relationships is still up in the air. (Madiega & Mildebrath, n.d.) They rarely provide practical recommendations for the use of FRT on their own, and they frequently only indirectly contain and resolve disputes at the intersection of data protection and developing technologies. Their 'expression' in secondary law, on the other hand, provides a practical structure. (Ivanova, 2020).

2.7.6 Discussions On Global Norms

AI regulation is a hot topic these days given the myriads of applications which has opened up so may dimensions (The OECD Artificial Intelligence Policy Observatory - OECD.AI, n.d.) In the context of two international forums, the question of how to deal with FRTs has been raised. In 2020, the United Nations Human Rights Council passed a resolution condemning the use of FRT in the context of peaceful protests, stating that these technologies have a chilling effect on the right to protest by enhancing governments' ability to identify, monitor, harass, intimidate, and prosecute protesters. (Human Rights Documents, n.d.). The Council urged nations not to use facial recognition technology to track people taking part in peaceful protests.

The European human rights organization, the Council of Europe (COE), is situated in Strasbourg. In January 2021, the Guidelines on Facial Recognition were adopted. (CONSULTATIVE COMMITTEE OF THE CONVENTION FOR THE PROTECTION OF INDIVIDUALS WITH REGARD TO AUTOMATIC PROCESSING OF PERSONAL DATA CONVENTION 108 Guidelines on Facial Recognition Directorate General of Human Rights and Rule of Law 2 Contents, n.d.) Governments, facial recognition developers, manufacturers, service providers, and entities employing FRT should observe and apply these rules to guarantee that they do not infringe on anyone’s human rights and basic freedoms, including the right to human dignity and personal data protection.

The rules include the usage of FRTs in both the private and public sectors and have a broad reach. They advocate for restrictions on the use of highly intrusive FRTs as well as the implementation of safeguards. The Council of Europe's next work on developing a legal framework for AI is likely to include principles that apply to facial recognition. (Result Details, n.d.) There is also bilateral cooperation, such as the Trade and Technology Council, which the EU and the US chose to establish as a venue for transatlantic collaboration and standard-setting for developing technologies like artificial intelligence. (EU-US: A New Transatlantic Agenda for Global Change, n.d.).

Facial recognition systems are employed in various capacities across EU nations. In France, pilot projects utilizing facial recognition technology (FRT) were conducted at schools in Nice and Marseille. These projects aimed to assist safety agents in controlling access to high schools, thereby preventing intrusions, identity theft, and streamlining security checks. However, the administrative court of Marseille annulled the municipality's decision to authorize FRT testing

in the two schools. Additionally, in 2020, the French Ministry of Home Affairs introduced ALICEM (Certified online authentication on mobile phones), a smartphone application leveraging FRT. This application enables individuals to securely verify their identity online using their smartphone in conjunction with their passport or residence permit. Despite these developments, the French data protection authority (CNIL) issued a favorable opinion on a draft decree authorizing the creation of the ALICEM system.

In Germany, efforts to enhance crime prevention at train stations included a 2019 pilot implementation of facial recognition technology (FRT) by police at Berlin's Südkreuz train station. Similarly, during the 2017 G20 summit, authorities in the city of Hamburg deployed FRT for crime detection and investigation purposes. Despite the initial ruling, a first instance court overturned the order from the Hamburg Data Protection Authority (DPA) mandating the deletion of the police database containing biometric templates. However, the Hamburg DPA contested the judgment by filing an appeal. Initially, police authorities relied on Sections 161 and 163, in conjunction with Section 98c of the German Criminal Procedure Code (GCPC). Later, they referred to Sections 161, 163, or alternatively, Section 483 of the GCPC. Media reports have indicated plans by the Berlin Zoo to implement facial recognition technology (FRT) to streamline access controls, prompting an inquiry by the Berlin Data Protection Authority (DPA) into the matter. Moreover, biometric-ready cameras have been supplied to at least 19 cities in Germany. Notably, the Cologne Police Headquarters deployed such cameras capable of live facial recognition. However, on January 18, 2021, the Cologne Administrative Court issued an injunctive order against the Cologne Police, halting video surveillance of Breslauer Platz and its adjacent streets in Cologne.

In Spain, facial recognition technology has found diverse applications across various sectors including surveillance-supportive services at airports, immigration checkpoints, and supermarkets. For instance, Madrid's South Station implemented a live face recognition system in 2016 to combat vandalism and petty crimes. Additionally, major players like Aena and Iberia have adopted facial recognition systems for boarding processes at airports since 2019. Moreover, facial recognition is utilized to enhance border control and security at crossings in Ceuta, while Mercadona, a prominent Spanish supermarket chain, employs FRT to identify individuals subject to restraining orders or court bans from entering their premises. In Italy, the police have utilized the Automatic Image Recognition System (SARI) for identification purposes since 2019. However, on April 16, 2021, the Italian data protection authority

expressed concerns, stating that the SARI system could lead to indiscriminate or mass surveillance if used as intended. Consequently, a draft bill proposing a moratorium on the use of facial recognition technologies in public spaces was introduced.

In Ireland, the Department of Social Protection employs a facial recognition system within the Public Services Card to combat social welfare fraud. Moving to the Netherlands, facial recognition technology is utilized for event control purposes during carnivals and large gatherings, while the Dutch police have adopted the CATCH system since 2016 to identify suspects or convicts through a criminal justice database. Furthermore, police are exploring real-time facial recognition technology using smartphone pictures, body cams, and cloud-based systems. However, the Dutch Data Protection Authority has issued a critical recommendation concerning the existing biometric legal framework (Wet Biometrie Vreemdelingenketen) and disapproves of extending its application period.

2.8 De-Identification

De-identification, particularly in the context of deepfake, is a crucial process aimed at mitigating the harmful impacts of manipulated media. Deepfake, boosted by advanced machine learning algorithms, can convincingly alter or superimpose faces onto existing videos, creating realistic yet entirely fabricated content. Facial de-identification is a privacy-preserving technique that involves altering identifiable facial features in images or videos to prevent the recognition of individual identities. This process seeks to balance the use of visual data for technological applications, such as machine learning training, with the preservation of personal privacy (Ribaric & Pavesic, 2015).

Facial de-identification is a response to growing privacy concerns, especially with the rise of powerful machine learning algorithms capable of identifying individuals in images and videos with remarkable accuracy. This technique ensures the integrity of personal identities while still allowing the utility of visual data in various applications. The methods employed for face de-identification range from simple techniques such as blurring or pixelation to more complex approaches involving the substitution of original faces with synthetic or generic ones. (Ribaric et al., 2016). The generation of synthetic faces often aims to retain certain non-identifiable attributes such as emotion, age group, or gaze direction, while removing specific identity information. Some advanced techniques involve differential privacy, where a specific amount

of 'noise' is added to data to prevent the deduction of individual identities while still enabling accurate population-level analysis (Ribaric & Pavesic, 2015).

De-identification techniques play a pivotal role in combatting the spread of misinformation and protecting individuals' privacy and reputations in this digital age. By anonymizing or obfuscating sensitive visual data, de-identification serves as a vital defense mechanism against the proliferation of deepfake content. One approach to de-identification involves the removal or manipulation of identifiable facial features within a video or image. This process typically entails techniques such as blurring, pixelation, or even the replacement of facial features with synthetic or generic elements. By obscuring specific facial characteristics, de-identification aims to disrupt the accuracy of facial recognition algorithms employed by deepfake technology, thereby rendering the content less susceptible to manipulation or misuse. However, striking a balance between preserving anonymity and maintaining visual integrity remains a significant challenge in the de-identification process, as excessive alterations may compromise the usability and authenticity of the content.

Moreover, de-identification efforts extend beyond facial features to encompass various metadata associated with digital media, including timestamps, geolocation data, and device information. Stripping or modifying such metadata helps prevent the tracking of content origins and enhances user privacy. Additionally, advancements in artificial intelligence have led to the development of automated de-identification tools capable of efficiently processing vast amounts of media content. Nevertheless, ongoing research and collaboration are essential to refining de-identification techniques and addressing emerging threats posed by increasingly sophisticated deepfake technology. Ultimately, effective de-identification strategies serve as a critical defence against the harmful ramifications of manipulated media, safeguarding both individuals' privacy and the integrity of digital content ecosystems. Figure 2.8 illustrates various de-identification techniques.

Figure 2 - 8 8 Images of various de-identification techniques (Siu et al 2020).

2.8.1 Blurring

Blurring de-identification is a technique used in data anonymization that involves the reduction of image clarity to obfuscate identifiable information, thereby protecting individual privacy. Blurring stands out as a prominent de-identification technique for combating deepfake manipulation. In essence, blurring involves the intentional obscuring of specific areas within an image or video to render them unidentifiable while preserving the overall context of the content. In the context of deepfake studies, blurring is often employed to protect the identities of individuals featured in videos or images that may be vulnerable to exploitation or misuse. This method is commonly applied to personally identifiable visual data, including faces, license plates, or other unique identifiers within images or videos (Avidan & Butman, 2006).

One of the primary advantages of blurring as a de-identification method lies in its simplicity and effectiveness. By applying a blur filter to facial features or other identifiable elements within a video, researchers and content moderators can quickly anonymize sensitive information without fundamentally altering the integrity or visual coherence of the content. This approach is particularly valuable in scenarios where preserving the authenticity or integrity of the original media is paramount, such as in forensic analysis or academic research.

Blurring de-identification, as a component of the larger field of privacy-preserving techniques, is based on the process of deliberately decreasing the resolution of certain areas within an image or video. This decrease in resolution, or 'blurring', obscures the details necessary for identification. Despite the blurring, the overall context or essential non-identifying attributes of the image or video can still be preserved. (McPherson et al., 2016). One of the key strengths of the blurring de-identification method lies in its simplicity and accessibility; blurring can be implemented with relative ease compared to more complex privacy-preserving methodologies. However, a trade-off exists between the level of anonymization and the utility of the data. Excessive blurring may render data impractical for certain applications, such as machine learning or research (Y. Li et al., 2016; Ribaric & Pavesic, 2015). Figure 2.9 illustrates blurring.

Figure 2 - 9 images of blurring de-identification technique

It's worth noting that while blurring de-identification offers a level of privacy protection, its efficacy can vary depending on the complexity of the deepfake technology involved., and therefore, not fool proof. Techniques have been developed to reverse-engineer blurred images, raising potential concerns over its effectiveness as a standalone privacy measure. Consequently, it is often advised to combine blurring with other privacy-enhancing techniques to achieve robust de-identification (Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization (UCLA Law Review, n.d.). Continued research and innovation in the field of de-identification techniques are essential to staying ahead of evolving deepfake

threats. Collaborative efforts between researchers, policymakers, and industry stakeholders are crucial for developing robust frameworks and standards for de-identification that effectively mitigate the risks posed by deepfake technology while upholding principles of privacy, security, and authenticity in digital media ecosystems.

2.8.2 Blacked

Blacked de-identification is an anonymization method where identifiable information within an image or video is concealed by overlaying it with a solid, usually black, shape or area. This technique is commonly applied to visual data that includes personally identifiable details such as faces, license plates, or other unique visual identifiers (Y. Li et al., 2016). Blacked de-identification operates by entirely obscuring specified regions of an image or video. Unlike techniques such as blurring or pixelation, which reduce the resolution of the identifiable area, blacking out involves completely covering the targeted portion of the image, making it impossible to perceive any underlying details. (Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization (UCLA Law Review, n.d.; Gellert & Gutwirth, 2013). The primary advantage of blacked de-identification lies in its thoroughness. Because it completely obscures the targeted data, it leaves no room for potential re-identification through advanced image processing or machine learning algorithms. However, this level of thoroughness also results in a significant loss of data, potentially limiting the usefulness of the image or video for certain applications. (Y. Li et al., 2016). Figure 2.10 illustrates images of Blacked de-identification technique.

Figure 2 - 10 images of Blacked de-identification technique (Y. Li et al., 2016).

While blacked de-identification provides a robust level of privacy, it's important to consider the impact on data utility. The method is best suited for scenarios where the obscured

information is not necessary for the data's intended use, or where privacy concerns heavily outweigh the need for complete data (Gellert & Gutwirth, 2013; Meden et al., 2023).

2.8.3 Pixelate

Pixelation de-identification is a data anonymization method that involves replacing identifiable elements in an image or video with larger, less detailed pixels, obscuring the information to protect individual privacy. This technique is commonly used for personally identifiable data such as facial features, license plates, or other unique identifiers within visual media.(Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization | UCLA Law Review, n.d.). Pixelation de-identification operates by reducing the resolution of specified regions within an image or video to a point where individual elements become indistinguishable. Each pixel in the targeted region is enlarged, effectively transforming the detailed image into a low-resolution, blocky abstraction that retains the original colour scheme but lacks the clarity to identify specific details (McPherson et al., 2016; Meden et al., 2023).

The pixelation technique offers a straightforward approach to preserving privacy. Its strength lies in its simplicity and wide recognition; pixelated images are commonly understood to represent censored or anonymized content. (McPherson et al., 2016). However, like blurring, pixelation de-identification presents a trade-off between the level of privacy and the utility of the data. Too much pixelation can degrade the data to a point where it no longer serves its intended purpose. Furthermore, despite the de-identification, more advanced techniques have been developed that can partially reverse engineer pixelated images, calling into question the security of pixelation as a standalone privacy measure (McPherson et al., 2016). Therefore, it's recommended to pair pixelation with other privacy-enhancing techniques to ensure robust de-identification. Additionally, the level of pixelation should be carefully calibrated to maintain the balance between data utility and privacy. Face de-identification techniques have gained relevance in the era of stringent privacy laws like the General Data Protection Regulation (GDPR). They provide a crucial tool for using and sharing visual data in a responsible, privacy-conscious manner, thereby advancing technology without infringing on personal privacy (Meden et al., 2023; Ribaric et al., 2016; Ribaric & Pavesic, 2015). Figure 2.11 gives images of pixelation de-identification technique.

Figure 2 - 11 images of pixelation de-identification technique

2.9 Review of Existing Deepfake Technologies

With a focus on Deepfake images of the face, (Lample et al., 2017) introduced an "encoder-decoder" architecture (Fader Networks) capable of generating different realistic versions of an input image by varying the values of the attributes: given an input image x with its details y, the encoder maps x to a latent representation z, and the decoder is trained to reconstruct x given z. (z,y). A test picture is encoded in the latent space at inference time, and the user selects the values of the attributes y that are transmitted to the decoder. During training, a classifier learns how to predict the attribute given the latent representation z. The encoder-decoder is trained to ensure that the latent representation z contains enough information to allow input reconstruction while also preventing the classifier from correctly predicting attribute values. (Z. Liu et al., 2015) used the CelebA dataset to train and evaluate Fader Network, resulting in a model that can drastically change the perceived value of attributes while maintaining the natural appearance of the input images. Changing the colour of the flower while leaving the backdrop unchanged was also tested on the Oxford-102 dataset (Nilsback & Zisserman, 2008) (which has roughly 9,000 photos of flowers categorised in 102 categories). In this test, as well, excellent results were obtained.

Shen et al. present a novel approach for face attribute manipulation based on residual image learning. It can simulate picture alteration by learning the residual image, defined as the difference between the original input image and the desired altered image. Instead of focusing on the entire face, which contains many redundant and useless characteristics, the suggested work concentrates on the attribute-specific facial area. They devise a dual system capable of simultaneously learning two inverse attribute manipulations (one as the primal and the other as the dual). There are two image transformation networks called G0 and G1, as well as a discriminative network called D.G0 and G1 that replicate the primal and dual manipulations

for each face attribute manipulation. D divides the generated and reference images into three categories.

Several Deepfake-based algorithms that are now available are limited in their ability to manage more than two domains (for example, changing the hair colour, gender, age, and many other aspects of a face). They must develop separate models for each pair of picture domains. Choi et al. presented utilising a technique called StarGAN, a generative adversarial network, a method capable of executing image-to-image translations on many domains with a single model. The primary goal was to design a scalable image-to-image translation model that could be used in various domains while only requiring a single generator and discriminator. CelebA (Z. Liu et al., 2015) and RaFD dataset (Langner et al., 2010) were the authors' two types of face datasets. The network can perform an image-to-image translation operation using a random label (for example, hair colour, facial expression, etc.) as input. The findings were compared to those of other current approaches (J.-Y. Zhu et al., 2017), (M. Li et al., 2016), (Perarnau et al., 2016), demonstrating that StarGAN is capable of producing images of more excellent visual quality. The framework for face ageing called Identity-Preserved Conditional Generative Adversarial Networks (IPCGAN) consists of three components: a CGANs module (which uses an input image and a target age to create a new face with that age), an identity-preserved module (which ensures that the aged face retains the same input identity), and an age classifier (to make sure that the output has the desired age).

The Style Generative Adversarial Network, or Style-GAN (Karras et al., 2019), proposes several significant changes to the generator model, including the use of a mapping network to map points in latent space to an intermediate latent space, the use of style control at each point in the generator model, and the introduction of noise as a source of variation at each end in the generator model. The resulting model is capable of creating amazingly photorealistic high-quality images of faces and altering the style vectors and noise to modify the style of the created image at different degrees of detail. NVIDIA, a visual computing company, provided an open-source code for photorealistic face-generating software based on the StyleGAN algorithm in December 2018 (Karras et al., 2019).

Then, on February 11, 2019, Uber's computer engineer Phillip Wang constructed the website. https://thispersondoesnotexist.com/ and shared it with the public Facebook Artificial Intelligence and Deep Learning group. The goal of the StyleGAN algorithm is to make realistic

pseudo-portraits that are difficult to spot as fakes. StyleGAN also has trouble defining the teeth and identifying the backdrops. Furthermore, fluorescent dots are standard, which seem like water drops and can appear anywhere on the image. To address these flaws in StyleGAN, Karras et al. proposed StyleGAN2 (Karras et al., 2020), a new generator version that includes redesigned normalisation, multi-resolution, and regularisation algorithms.

Tools	Links	Findings
Faceswap	https://github.com/deepfakes/faceswap	● Using two encoder-decoder pairs. ● Parameters of the encoder are shared
Faceswap-GAN	https://github.com/shaoanlu/faceswap-GAN	Adversarial loss and perceptual loss (VGGface) are added to an auto-encoder archi-tecture
Few-Shot FaceTranslation	https://github.com/shaoanlu/fewshot-face-translation-GAN	Use a pre-trained face recognition model to extract latent embeddings for GAN processing.- Incorporate semantic priors obtained by modules from FUNIT [42] and SPADE [43]
DeepFaceLab	https://github.com/iperov/DeepFaceLab	Expand from the Faceswap method with new models, e.g., H64, H128, LIAEF128, SAE [44].- Support multiple face extraction modes, e.g. S3FD, MTCNN, dlib, or manual [44].
DFaker	https://github.com/dfaker/df	DSSIM loss function [45] is used to reconstruct face.- Implemented based on Keras library.
AvatarMe	https://github.com/lattas/AvatarMe	Reconstruct 3D faces from arbitrary “in-the-wild” images. - Can reconstruct authentic 4K by 6K-resolution 3D faces from a single low-resolutionimage [46].
DeepFaketf	https://github.com/StromWine/DeepFaketf	Similar to DFaker but implemented based on tensorflow.

MarioNETte	https://hyperconnect.github.io/MarioNETte	A few-shot face reenactment framework that preserves the target identity.- No additional fine-tuning phase is needed for identity adaptation [47].
DiscoFaceGAN	https://github.com/microsoft/DiscoFaceGAN	Generate face images of virtual people with independent latent variables of identity,expression, pose, and illumination.- Embed 3D priors into adversarial learning [48].
StyleRig	https://gvv.mpi-inf.mpg.de/projects/StyleRig	Create portrait images of faces with a rig-like control over a pretrained and fixedStyleGAN via 3D morphable face models.- Self-supervised without manual annotations [49]
FaceShifter	https://lingzhili.com/FaceShifterPage	Face swapping in high-fidelity by exploiting and integrating the target attributes.- Can be applied to any new face pairs without requiring subject specific training [50]
FSGAN	https://github.com/YuvalNirkin/fsgan	face swapping and reenactment model that can be applied to pairs of faces without requiring training on those faces.- Adjust to both pose and expression variations [51].
TransformableBottleneckNetworks	https://github.com/kyleolsz/TB-Networks	A method for fine-grained 3D manipulation of image content.- Apply spatial transformations in CNN

		models using a transformable bottleneck framework [52].
“Do as I Do”Motion Trans-fe	github.com/carolineec/EverybodyDanceNow	Automatically transfer the motion from a source to a target person by learning a video-to-video translation.- Can create a motion-synchronized dancing video with multiple subjects [53].
NeuralVoicePuppetry	https://justusthies.github.io/posts/neural-voice-puppetry	method for audio-driven facial video synthesis.- Synthesize videos of a talking head from an audio sequence of another person using3D face representation. [54].

Table 2- 1 Summary of existing studies on deepfake technologies

2.10 Comparison of Standard De-identification and Deepfake

Blurring, pixelation, blacking out, and other deepfake techniques are all methods used to modify digital images, especially human faces, but they are used for different purposes and have different effects on the resulting images.

Blurring	This technique obscures facial features by averaging the pixel values in a local neighbourhood. The resulting image is less identifiable, which can help protect privacy in certain contexts. However, the blurring process can be irreversible if done aggressively, meaning that the original image cannot be perfectly restored.
Pixelation	Pixelation also obscures facial features, but it does so by reducing the resolution of the image. This results in a grid-like appearance with large, blocky pixels. Like blurring, pixelation can also protect privacy, but it tends to produce more visually disruptive results.
Blacking-out	Blacking out involves covering the face or other identifying features with a solid colour (usually black). This effectively removes all identifiable information but also results in a loss of all facial details, which may not be desirable in certain applications.
Deepfake	Unlike the above methods, deepfake techniques are not primarily used for de-identification. Instead, they are used to replace one person's face with another's in an image or video. This process involves complex machine learning models and can produce very realistic results.

Table 2- 2 Comparison of de-identification techniques and deepfake.

In summary, blurring, pixelation, and blacking-out are simple techniques for de-identifying faces in images and can be effective at protecting privacy, but they also result in a loss of information. On the other hand, deepfake can create highly realistic modifications of images or videos.

2.11 Method Using First Order Motion Model

Aliaksar Siarohin and his team pioneered the development of open-source codes for deepfake technology, specifically focusing on the first-order motion model for image animation. These codes, meticulously detailed and written in the Python programming language, will serve as the foundational infrastructure for our system. Our future work will extend and build upon this established foundation. Figure 2.12 gives a diagrammatical explanation of the first order motion model.

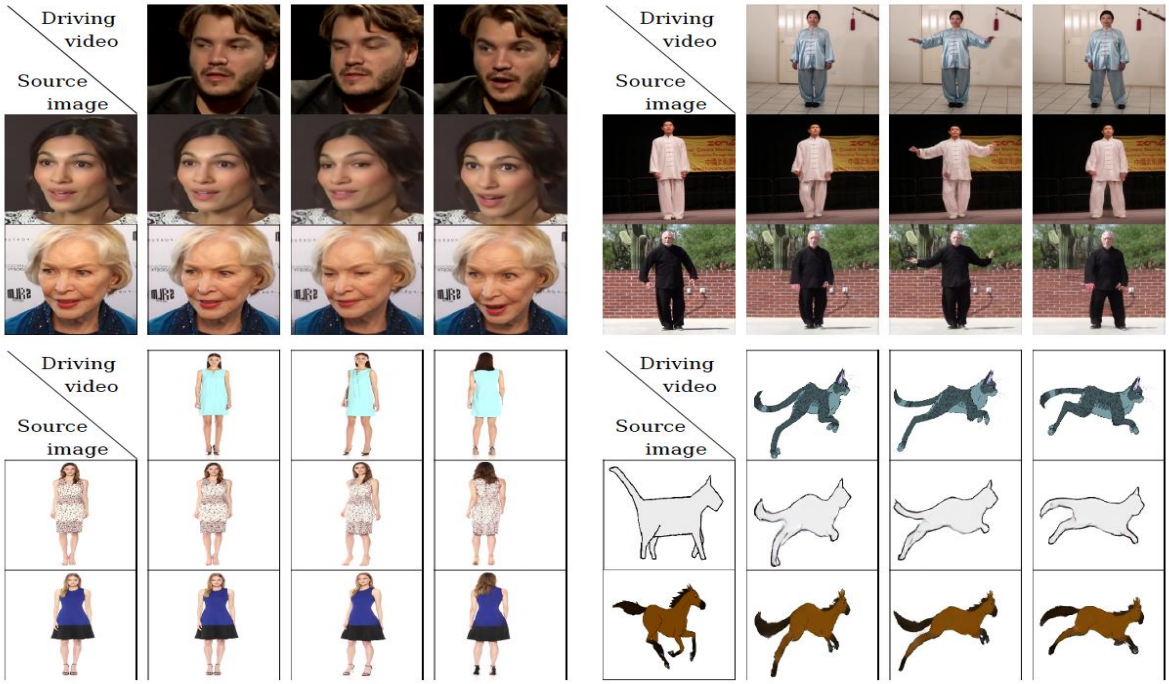


Figure 2 - 12 First order motion model (Aliaksar Siarohin et al 2019).

Motion estimation and image generation are the two fundamental components of the first order motion model, and motion estimation also includes coarse motion estimation and dense motion prediction. While dense motion produces an optical flow and the confidence map for the entire image, coarse motion is described as sparse motions between different object sections. the source and the driving frames taken from the same video are indicated, respectively, by S and D. Figure 2.13 gives the overview of the first order motion model.

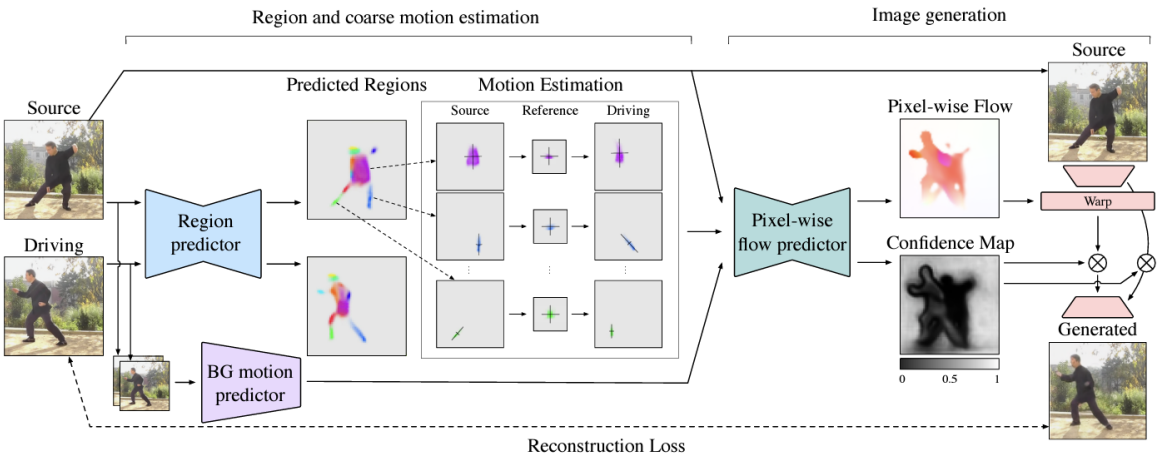


Figure 2 - 13 Overview of the first order motion model (Aliaksar Siarohin et al 2019).

A description of the model. Heatmaps for each component in the source and driving pictures are returned by the region predictor. The regions of each heatmap are then transformed from the source to the driving frame through a whitened reference frame by computing the primary axes of each heatmap. Pixel-wise flow prediction network combines background and region transformations. As shown by the confidence map, the target picture is created by warping the source image in feature space using the pixel-wise flow and painting newly introduced regions.

2.11.1 Setting up the environment

We will initiate our process by setting up the computational environment on a machine tailored to our requirements. This machine is specifically selected to efficiently handle the intense processing needs of our project, thereby ensuring optimal performance. The next step involves the cloning of the necessary repositories. These repositories contain a wealth of valuable resources, including data, libraries, and frameworks that are instrumental to our model.

Once our repositories are set up and ready for use, we move on to designing the user interface for our model. We will deploy HTML for this purpose, an efficient markup language renowned for its flexibility and accessibility. This user interface will play a vital role in our project as it will serve as the gateway between us, and the data processed by our model. This interface is not merely a window for viewing input and output. It is an interactive platform that offers a range of customizations. It allows us to select from different deepfake driving images.
Furthermore, the interface is designed to be adjustable, enabling us to fine-tune the output according to our specific needs. This adaptability allows for continuous refinement and enhancement of the results.

2.11.2 Face detection and bonding box

In this phase, a face detection algorithm will be applied to our driving images, which are essentially the deepfake images. This step involves creating bounding boxes around the faces in the selected dataset. These bounding boxes serve as crucial reference points, isolating the faces in the images and enabling more efficient and accurate application of deepfake technology in real-time.

2.11.3 Output host

In this phase, we will establish a secure tunnel to our localhost utilizing tunnelling technologies such as ngrok or Argo. This pivotal process will allow us to retrieve a public URL to access our localhost server, thereby creating a direct, secure link between the server and the outside network. The aim behind this process is not just to create a bridge between the server and the external world, but to also serve our specific testing needs. One of our primary objectives is to investigate the potential of accessing datasets directly via URLs, instead of relying on locally stored datasets. Accessing datasets directly through URLs provides several advantages, including real-time data updates, and less local storage requirements. By setting up this tunnel and retrieving a public URL, we take a significant step towards a more streamlined and dynamic way of handling data. This setup not only improves our testing capabilities but also paves the way for a more efficient, future-proof data handling methodology.

2.12 Summary

We manage to be able to provide a comprehensive, detailed discussion supported by an extensive review of existing literature. We begin with a focus on facial recognition and detection, examining its various applications, methods of implementation, and the issues that arise from these applications.

Secondly, we delve into the intricacies of the General Data Protection Regulations (GDPR), a thorough exploration that significantly contributes to the novelty of our work. We then turn our attention to de-identification techniques, contrasting them with deepfake technology, thereby demonstrating the superiority of the latter as a method for de-identification. Further analysis of deepfake technology is provided, alongside a review of some existing deepfake technologies available in the market.

Furthermore, we explore potential industries where our proposed model could be implemented, illuminating these areas, and justifying our position on the matter. With the advent of the GDPR and the rise of Industry 4.0, the opportunities are seemingly limitless.

Finally, we draw our discussion to a close with an overview of machine learning and its related techniques. This exploration provides the necessary context to understand the interconnectedness of deepfake technology and the real-time deployment of deepfake.

As we move forward, the next chapter will delve into the various methodologies employed to achieve our research objectives.

CHAPTER 3

METHODOLOGY

3.1 Overview

In this chapter, a comprehensive overview of the methodology to be employed in this thesis is presented. The primary goal of this research is to address the research questions and objectives. The methodology adopted in this study aims to provide a clear understanding of the research process, and the various techniques employed to ensure the validity, reliability, and generalizability of the findings. To achieve the research goals, a structured research framework has been developed that will guide the entire research process. The framework includes the selection of appropriate research models, privacy considerations, algorithm deployed and datasets. This framework will ensure that the research process is well-structured and that all necessary steps are taken to produce reliable and valid results.

3.2 Research Methodology

This study aims to explore the digital landscape that the world is rapidly moving towards, with the Internet of Things (IoT) revolutionizing digital connectivity. The increasing prevalence of connected cameras in homes, offices, streets, public transport, and vehicles presents a significant threat to data privacy. In this research, a model utilizing deepfake technology is presented as a potential solution to limit and reduce the adverse impacts of facial recognition systems on data privacy. The study will also compare other de-identification techniques with the proposed deepfake model. Figure 3.1 shows the recommended research framework that outlines the methodology to be adopted in this study.

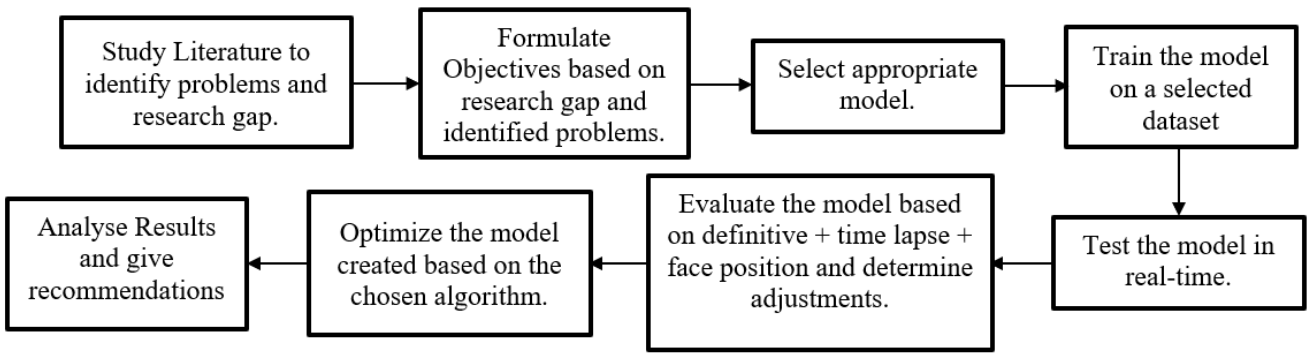


Figure 3 - 1 Developed Research Framework.

3.3 Selection of Appropriate Models

The selection of an appropriate model for the deidentification of deepfakes is a critical step in effectively mitigating the harmful effects of manipulated media. It involves careful consideration of various factors, including the complexity of the deepfake algorithms being targeted, the computational resources available, and the specific goals of the deidentification process. Models for deidentification can range from simple heuristic-based approaches to sophisticated deep learning architectures tailored to the task. Heuristic-based methods often rely on predefined rules and algorithms to detect and remove potential artifacts introduced by deepfake algorithms. On the other hand, deep learning models leverage large datasets to learn complex patterns and features indicative of manipulated content, offering higher accuracy but requiring significant computational resources for training and inference. The selection of the appropriate model thus depends on a balance between accuracy, efficiency, and scalability, considering the specific requirements and constraints of the deidentification application.
To achieve deidentification based on the set objectives, two models are selected: first order motion model and the mediapipe.

3.3.1 First Order Motion Model of Deidentification

In the study of deidentification of deepfake, the first-order motion model (FOMM) plays a crucial role in understanding and mitigating the effects of manipulated videos. By formulating the motion of facial features and expressions as a first-order differential equation, anomalies in the movement patterns introduced by deepfake algorithms can be detected. To achieve this, the velocity, v and displacement, d of key facial landmarks is analysed over time to distinguish between genuine and manipulated videos. optical flow estimation and motion consistency checks are employed to identify inconsistencies and artifacts characteristic of deepfake-generated content. In this research, optical flow estimation is employed. Optical flow refers to the pattern of apparent motion of objects in a sequence of images or between consecutive frames in a sequence. For a pixel located at position (x, y) in frame t, its displacement over time can be described by the optical flow vector (u, v). The optical flow equation typically takes the form of equation (4.1)

$$I_x u + I_v y + I_t = 0$$

where I_x and I_y are the spatial gradients of the image intensity with respect to x and y, respectively; I_t represents the temporal gradient of the image intensity with respect to time;

and u and v are the horizontal and vertical components of the optical flow vector, respectively. Figure 3.2 shows the framework for obtaining the setting up a project using FOMM for image anonymization.

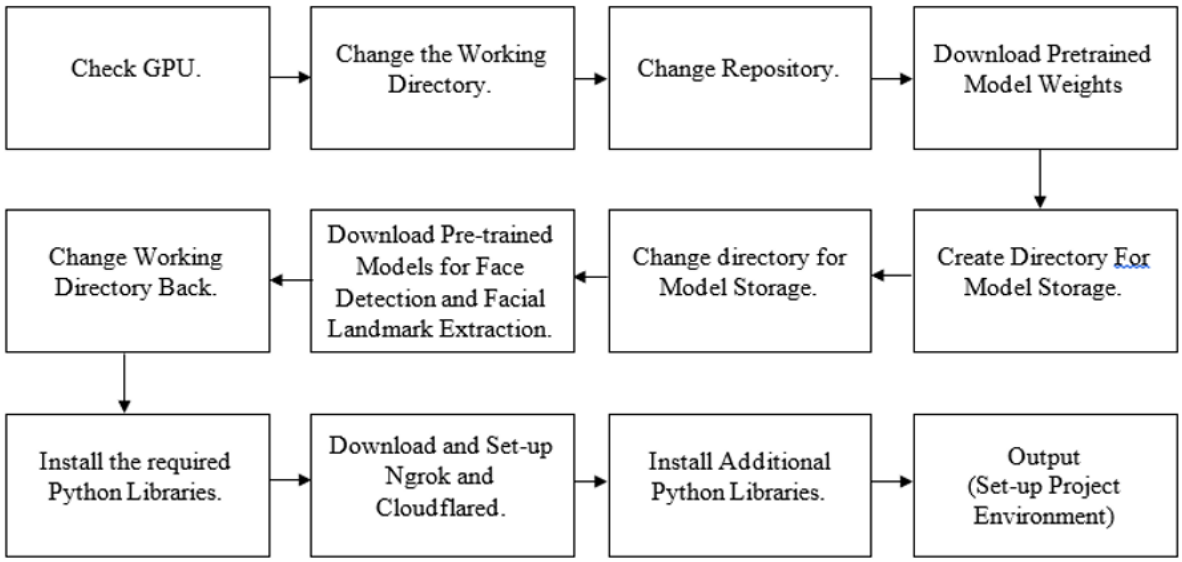


Figure 3 - 2 Setting up using FOMM for image anonymization.

3.3.2 Mediapipe Model of De-identification.

MediaPipe is an open-source cross-platform framework for building real-time computer vision and machine learning pipelines. It provides a flexible and efficient infrastructure for processing multimedia data such as images and videos and can be used for a wide range of applications such as object detection, face detection, hand tracking, and augmented reality. The MediaPipe framework consists of several stages, each of which performs a specific task in the processing pipeline. Mediapipe is an open-source cross-platform framework developed by Google that provides a collection of reusable pipelines and components for building computer vision and multimedia applications. The framework (Figure 3.3) includes pre-built pipelines for face detection and face landmark detection, which use a combination of machine learning and computer vision techniques to detect and locate faces and facial landmarks in images and videos.

The type of machine learning used in the inference stage of MediaPipe can vary depending on the application but commonly involves deep learning models such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs). These models are trained on large datasets of labelled examples and can learn complex patterns and relationships in the data.

The formulas used in deep learning models such as CNNs and RNNs can be quite complex, but they generally involve a series of matrix operations and nonlinear transformations. For a convolutional layer in a CNN, equation (3.2) gives the output.

$$Y = \sigma(WX + B) \tag{3.2}$$

Where:

- Y is the output of the convolutional layer
- X is the input to the layer
- W is the weight matrix for the layer
- B is the bias vector for the layer
- σ is a nonlinear activation function such as ReLU or sigmoid

This formula represents a convolutional operation on the input data X, using a set of learnable weights W and biases B. The output of this operation is passed through a nonlinear activation function σ to introduce nonlinearity into the model (Figure 3.4). This process is repeated for each layer in the CNN until the final output is produced.

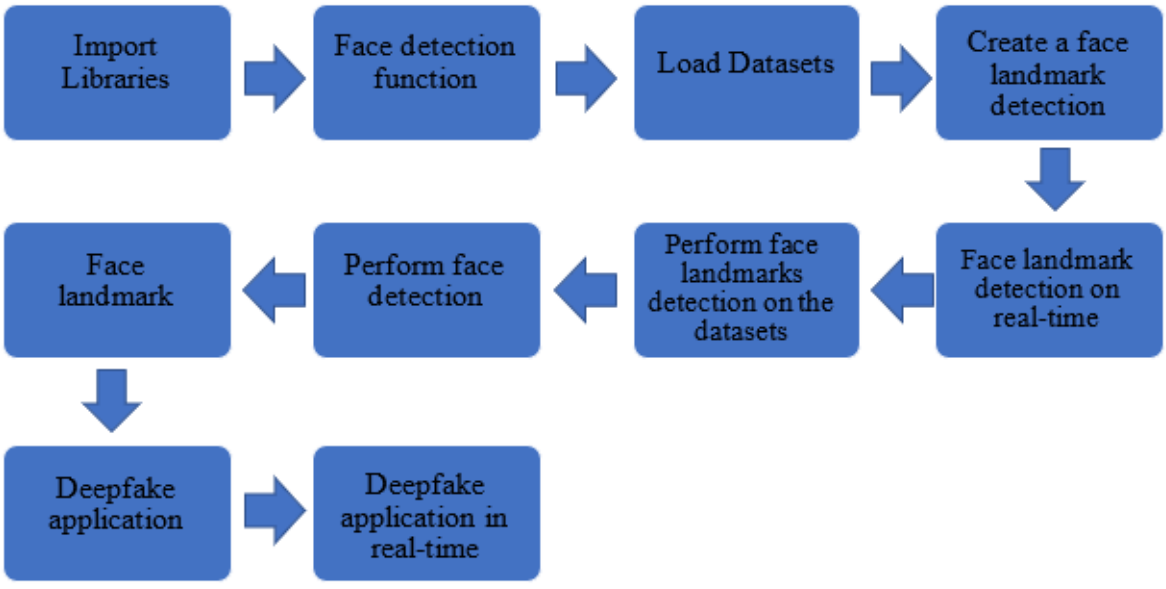


Figure 3 - 3 Mediapipe model framework.

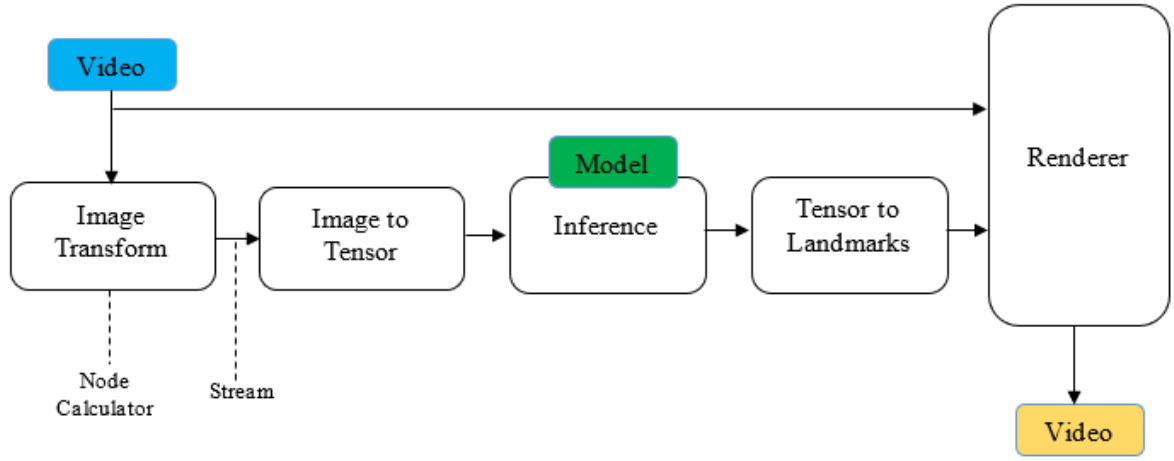


Figure 3 - 4 Mediapipe model

3.3.3 Applying the Deepfake Algorithm

In this phase of the project, the deepfake technology will be put into action. The checkpoint from the trained first-order motion model, a crucial component containing the information acquired during the model's training phase, will be loaded as the first step. Subsequently, the system will be directed to the driving images dataset, comprising a collection of deepfake images, and the source will be set to the camera of choice. Finally, the system will be commanded to read and detect faces from the selected camera source. Once the faces have been identified, the deepfake images from the driving images dataset will be applied. The culmination of this process will result in the successful application of the deepfake technology.

3.3.4 Libraries Utilized

Various python libraries were imported into the program. When the Program is run, it will use these libraries to perform various functions. There are as follows:

- **"Import Mediapipe"** The mediapipe library, is a library used for building a machine learning pipeline for various multimedia applications.
- "**Import cv2"** The OpenCV library, which is a popular computer vision library used for image and video processing.
- **"Import itertools"** is the library, which provides functions for working iterators and combinations of data.

- **“Import Numpy”** The Numpy library, which is used for numerical computing in Python.
- **“From time import Time”** The time function from the time module, which we applied here to measure the time taken to execute certain parts of the program.
- **“Import matplotlib.pyplot”** The pyplot module from the matplotlib library which is a library used to help us in creating visualizations.

3.3.5 Face Detection

The face detection pipeline in Mediapipe uses a machine learning model to identify regions in the input image that contain faces. The model is based on a single-shot multibox detector (SSD) architecture, which is a type of convolutional neural network (CNN) that is optimized for real-time object detection. The SSD model is trained on a large dataset of labelled face images to learn to recognize the distinctive features of human faces. Figure 3.5 gives the facial detection process.

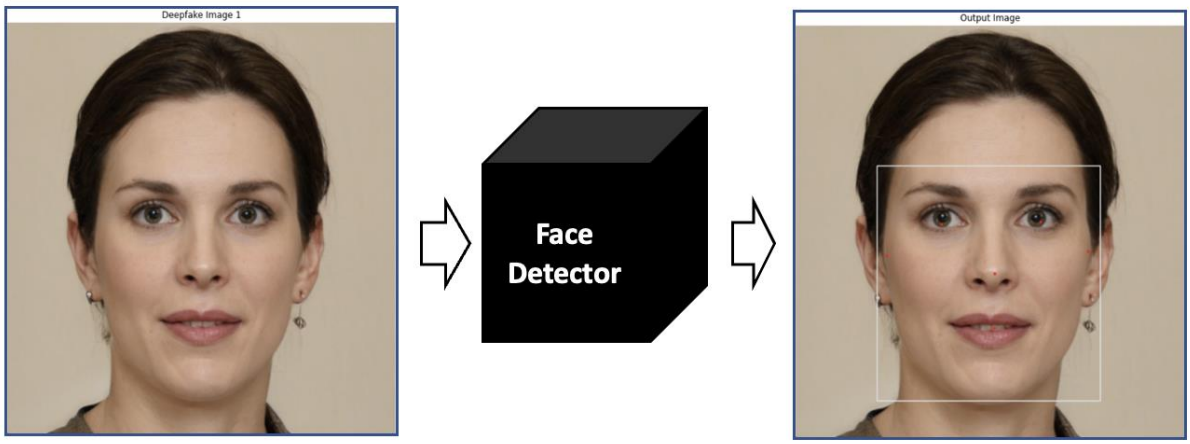


Figure 3 - 5 The facial detection process.

3.3.6 Face Landmark Detection

The face landmark detection pipeline in Mediapipe uses a machine learning model to predict the coordinates of specific facial landmarks such as the eyes, nose, and mouth. The model is based on a type of neural network called a regression tree ensemble, which is trained on a large dataset of labelled facial landmark images to learn to map the input image features to the landmark coordinates. The facial landmark detection is as shown in Figure 3.6

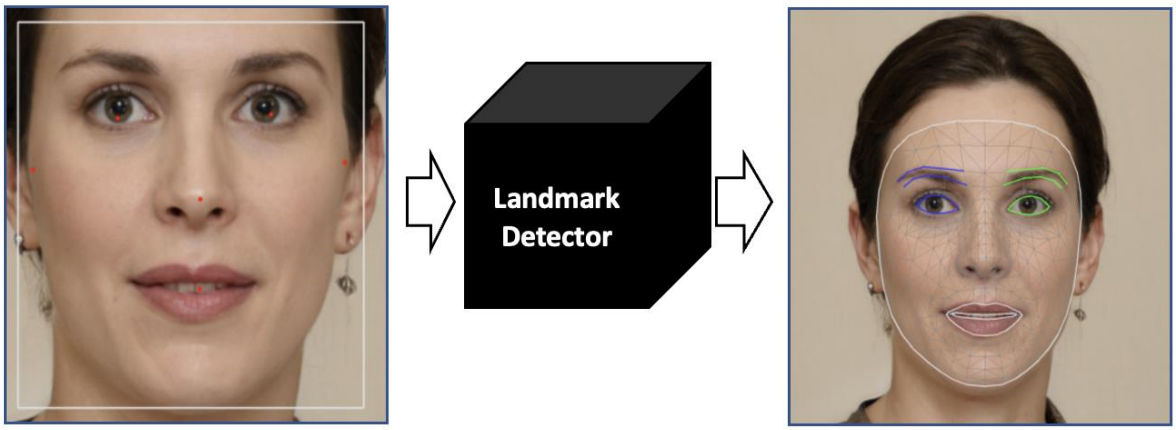


Figure 3 - 6 Facial Landmark detection

When the system first starts processing the video, it takes the first frame and runs it through a face detection model to identify any faces present in the image. The model then maps out a bounding box around each detected face.

After the face detection step, the system runs the frame through a landmark detection model to identify specific facial landmarks, such as the eyes, nose, and mouth. This step provides additional information about the faces that can be used for tasks like emotion recognition or facial expression analysis.

However, this process only needs to be done for the first frame of the video. In subsequent frames, the system can skip the face detection step because it already knows where the faces are located in the image. Instead, the system uses a facial tracking model that takes the coordinates of the bounding box from the previous frame and tracks the face throughout the video.

This process of using facial tracking instead of facial detection for subsequent frames significantly reduces latency because the facial tracking model is much faster than the facial detection model. By combining both facial detection and facial tracking, the system can increase the speed of the entire model while still maintaining accurate tracking of faces throughout the video (Figure 3.7).

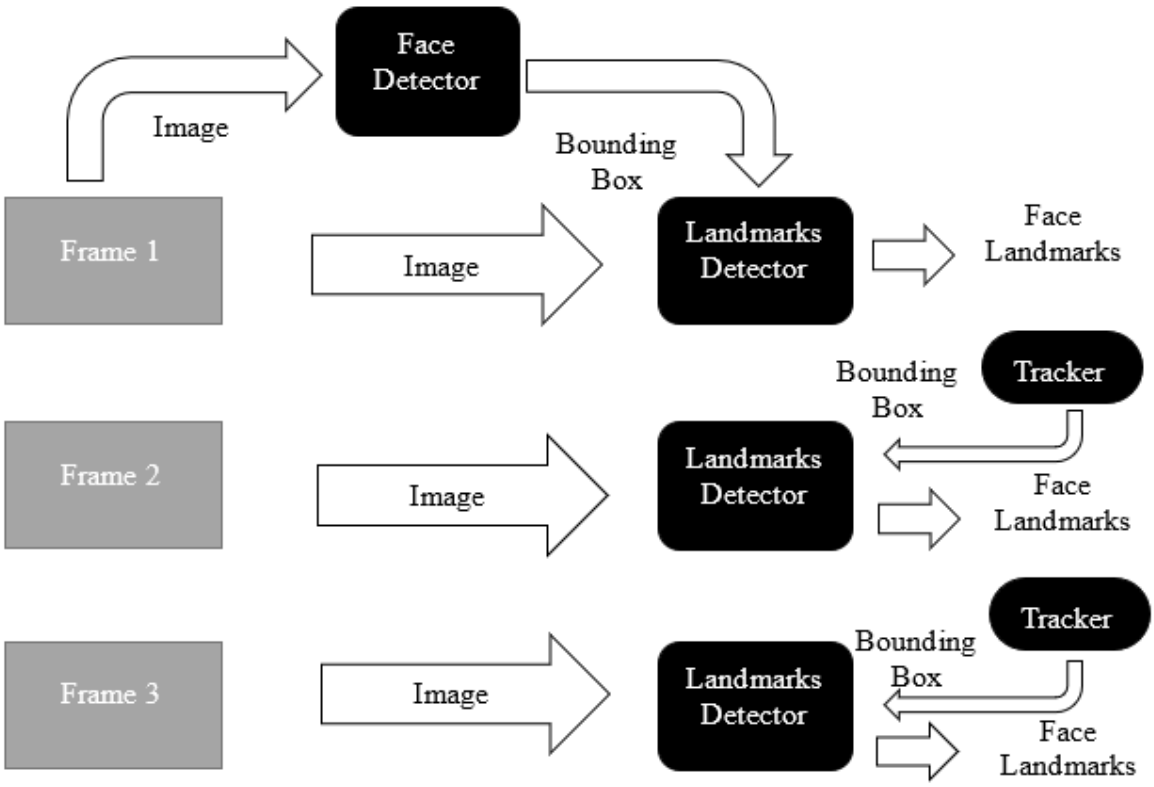


Figure 3 - 7 Process facial landmark detection frame by frame.

3.3.7 Creating the Deepfake

This process involves defining several key parameters that will shape the behaviour and output of the developed model. The first parameter is the source input, which essentially sets the baseline for the deepfake transformation. It is the original image or video stream upon which the deepfake model will apply changes. Next, the 'driving image' or the deepfake image is defined. This is the image that provides the target characteristics for the transformation, essentially dictating the desired output of the deepfake process. The 'face landmark' parameter comes next. These landmarks are key points on the face (like the corners of the eyes or the tip of the nose) that the model uses to recognize and align faces. Defining these landmarks enhances the model's accuracy in manipulating facial features. Then, the 'face part' parameter, which controls the region of the face that the model focuses on, is defined. The ‘face indexex’ is then specified. These indexes refer to the specific faces in the source input that the model that should apply the deepfake. This is especially important when the source input contains multiple faces. After defining these parameters, their names and settings were aligned with the specific objectives. Each of these parameters is adjustable, providing the desired flexibility and

control over the deepfake process. The final step in this stage is to set up a display, which will provide a visual output of the model's results. This procedure not only allows easy monitoring of the deepfake process in action but also helps the assessment of the accuracy and effectiveness of the model, informing future refinements and improvements.

3.3.8 Applying the Deepfake

In the final stage of this research, the deepfake model is brought to life in a real-time environment. This step marks the practical application of all the preceding stages, where the theoretical underpinnings and preparatory groundwork are actualized. Firstly, the source input is fed to the chosen camera. This input could be a live video feed or a static image, depending on specific requirements. By allowing flexible source input, flexibility is ensured for the model to adapt to a wide range of use cases. Next, face detection technology is applied. This process identifies and isolates the faces in the source input, providing a focused region for the deepfake model to work on. By narrowing down the area of interest to just the faces, efficiency and accuracy of the model are increased. The final step in this process, and indeed in the entire project, is to apply the deepfake model. At this stage, the model leverages the parameters set earlier, transforming the faces in the source input based on the characteristics of the driving image. As the deepfake model goes to work, its effects are observed in real-time, witnessing the transformation as it happens. This real-time application showcases the culmination of efforts, bringing together all elements of the project into a single, dynamic process.

3.3.9 Method with DeepFaceLab

To DeepFaceLab for the creation of deepfakes, Figure 3.8 shows the framework that was followed. The model has two sides. The backend where most of the codes are written and the frontend where the user interface is, enable the user to customize their pipeline. The various stages is as depicted in Figure 3.9

- Data Collection: Gather the necessary data for your deepfake project. This includes obtaining a dataset of training images (both source and target faces) and optionally, a set of driving videos for better animation results.

- Preprocessing: Before training the deepfake model, preprocess the collected data. This may involve aligning faces, cropping images, and ensuring consistency in lighting and pose across the dataset.
- Training the Model: Utilize DeepFaceLab to train the deepfake model using the prepared dataset. Training involves iteratively optimizing the model's parameters to generate realistic facial manipulations. This process may take significant computational resources and time.
- Fine-tuning (Optional): Fine-tuning the trained model can further enhance the quality of generated deepfakes, especially when dealing with specific characteristics or features.
- Conversion: After training or fine-tuning, use DeepFaceLab to convert the desired source images or videos into deepfake content. This step involves applying the trained model to the input data to create the manipulated output.
- Post-processing: Refine the generated deepfake content to improve its quality. Post-processing techniques may include smoothing facial animations, adjusting colors, or refining details to make the deepfake appear more realistic.
- Evaluation: Assess the quality of the generated deepfakes. This involves subjective evaluation by human observers as well as objective metrics to measure factors like realism, consistency, and artifacting.
- Deployment (Optional): If applicable, deploy the generated deepfake content for its intended purpose. This could involve sharing the content online, incorporating it into a multimedia project, or using it for research or entertainment purposes.
- Ethical Considerations: Throughout the process, it's essential to consider the ethical implications of creating and sharing deepfake content. This includes obtaining consent from individuals depicted in the source material, avoiding harmful or misleading use cases, and being transparent about the artificial nature of the content.

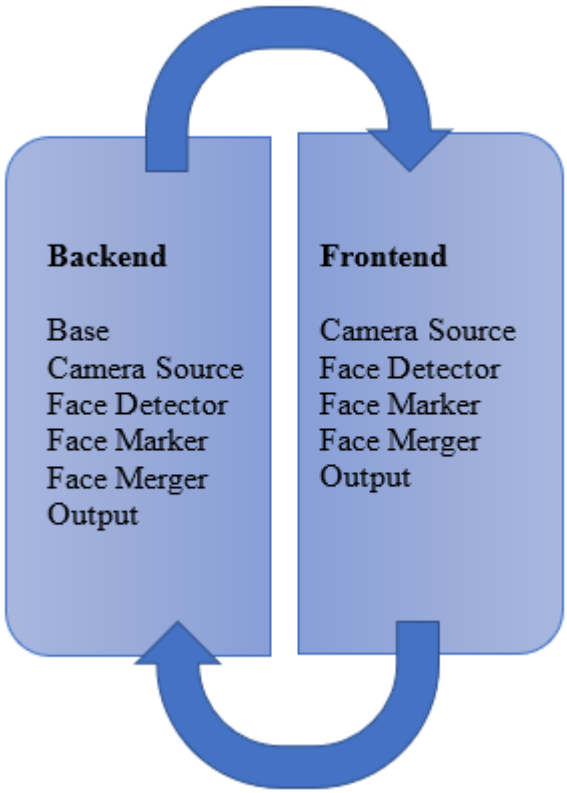


Figure 3 - 8 Shows how the DeepFaceLab system communicates between the backend and the frontend.

The Base serves as the fundamental framework of the system, acting as a canvas or structural layout for the model. Key functions are encapsulated within several classes.

Camera source is to enable the system to handle different camera drivers, including DirectShow, Microsoft Media Foundation, GStreamer, and others. The library can handle different resolution types and offers a wide range of camera configurations like FPS, rotation, horizontal flip, brightness, contrast, saturation, hue, and sharpness among other settings.

The system will employ the YoloV5 face detection method, an advanced and efficient technology for real-time object detection. YOLO, an acronym for 'You Only Look Once,' will use the fifth iteration, YoloV5, as it substantially improves object detection precision and computational efficiency than the previous versions.

The Face Marker will deploy a face marker algorithm that implements a Face Marker class and a FaceMarkerWorker class, which are part of a backend worker system. The face marker algorithm can use three different types of face landmarking methods: OpenCV LBF, Google Face Mesh, and InsightFace2D106

The Face Merger class is a component designed to facilitate the merging of a source image and a target image derived from a dataset. Its operation relies on five key elements: Backend Weak Heap, Backend Signal, two instances of Backend Connection for input and output respectively, and Backend DB. By combining these components, the FaceMerger allows us to seamlessly merge facial data from two distinct images.

The Output function facilitates interaction with the output window, which is integral for managing video frames.

Figure 3 - 9 the stages for the model to be deployed.

3.4 Privacy Consideration through Masking

Privacy in real-time deepfake applications is achieved through the effective implementation of masking and obfuscation methods, which prevent the re-identification of individuals while preserving the utility of the data. One commonly used metric for quantifying privacy protection is the information entropy (H). Information entropy measures the uncertainty or randomness associated with the data, with higher entropy indicating greater anonymity and lower re-identification risk. By applying masking and obfuscation techniques such as pixelization (P), blurring (B), and dynamic masking (D), the information entropy of sensitive regions in images or videos is increased, thereby reducing the likelihood of identifying specific individuals. The overall privacy level (PL) achieved by these methods can be expressed as a function of the entropy increase (ΔH) and the proportion of masked information (M) as shown in the equation below:

$$PL = f(\Delta H, M)$$

where $\Delta H = H_{masked} - H_{original}$, and M represents the fraction of masked pixels relative to the total number of pixels in the image or video. By maximizing the entropy increase and the proportion of masked information, masking and obfuscation methods enhance privacy protection and mitigate the risks associated with re-identification attacks.

```
import cv2
def pixelize(image, block_size):
    # Get dimensions of the image
    height, width = image.shape[:2]
    # Resize the image to the nearest multiple of the block size
    new_width = (width // block_size) * block_size
    new_height = (height // block_size) * block_size
    resized_image = cv2.resize(image, (new_width, new_height))
    # Resize the image back to its original dimensions with nearest-neighbor interpolation
    pixelized_image = cv2.resize(resized_image, (width, height), interpolation=cv2.INTER_NEAREST)
        return pixelized_image
def blur(image, kernel_size):
```

```
    # Apply Gaussian blur to the image
    blurred_image = cv2.GaussianBlur(image, (kernel_size, kernel_size), 0)
        return blurred_image
# Load the input image
input_image = cv2.imread('input_image.jpg')
# Apply pixelization with block size of 20
pixelized_image = pixelize(input_image, block_size=20)
# Apply Gaussian blur with kernel size of 9
blurred_image = blur(input_image, kernel_size=9)
# Display the original, pixelized, and blurred images
cv2.imshow('Original Image', input_image)
cv2.imshow('Pixelized Image', pixelized_image)
cv2.imshow('Blurred Image', blurred_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- The **pixelize** function takes an input image and a block size as parameters. It resizes the image to the nearest multiple of the block size using nearest-neighbor interpolation, effectively pixelizing the image. It then resizes the pixelized image back to its original dimensions.
- The **blur** function takes an input image and a kernel size as parameters. It applies Gaussian blur to the image using the specified kernel size.
- The input image is loaded using **cv2.imread**.
- Pixelization and blurring are applied to the input image using the defined functions.

3.5 Determining the System Specification

The proposed work will be on live videos. Due to the size of files associated with video feeds, a CPU will not be able to carry out the needed task, hence, the need for a Graphics Processing Unit (GPU). Furthermore, the framework of choice is Python running on an Ubuntu workstation powered by i7-6700 with 16 GB RAM. The system includes NVIDIA GPU GEFORCE GTX 1080. Table 3.1 shows the specification of the GPU.

GPU ENGINE SPECS	Values
CUDA Cores	2560
Base Clock (MHz)	1607
Boost Clock (MHz)	1733
MEMORY SPEC	
Memory Speed	10 Gbps
Standard Memory Config	8 GB GDDR5X
Memory Interface Width	256 bit
Memory Bandwidth (GB/Sec)	320

Table 3 - 1 System specifications

3.6 Datasets

Three different datasets were obtained and utilized for the analysis. To create an almost undetected deepfake video, one needs a lot of source material for the designed deepfake. Record a video and apply it through our pipeline and generate a deepfake video. However, we do not have the processing time of a recorded video, as we are applying it in real-time, so we will use a single image. Various datasets have been published and have been used for the implementation of deepfake algorithms and deepfake detection algorithms. Some of which have been stated in the literature. Of all the generated datasets to date, we will take a closer look and identify the best dataset best suited for our work and adopt it as we create our dataset using the generated photos datasets. We picked out some of the photos generated to create our dataset. Figure 3.10 gives the samples of images from the generated photos.

Figure 3 - 10 sample of dataset images from the generated photos.

Figure 3.10 is just a sample of millions of photos available in the datasets, we picked a few and formed our dataset. The above sample of the dataset has the same size and that will make implementation easier unlike the second dataset deployed for our second model using the first order motion model. Furthermore, the advantage these datasets give us is no risk of using unauthorised faces, and no ethical commitments as these photos and autogenerated using machine learning. Whereas Fakeface.rest returns random faces and does not generate the same face together.

Moreover, the second dataset consists of eight (8) personal images. As we will be using this dataset to test and train our second model using Mediapipe. the dataset consists of 8 different images with different characteristics. Such as quality, size, shape, number of people in a picture and background. Furthermore, for the testing, we used these personal images so there is no risk and ethical clearance needed. Finally, we utilized the datasets curated by DeepFaceLab specifically for face swapping. The process begins with an extensive collection of varied samples of the target face. These samples are then meticulously analysed and modelled, utilizing facial markers to create a comprehensive and precise dataframe. This data frame called faceset serves as a foundational asset in the development of convincingly realistic deepfake imagery.

Figure 3 - 11 Sample of a cleaned-up extracted faceset.

The Facesets utilized in the developed system are sourced entirely from the public domain, thereby sidestepping any potential ethical concerns.

3.7 Summary

In this chapter, we've delved into the heart of our research methodology, exploring the integral systems required for effective image and frame processing. We've examined various frameworks, each comprising different models tailored to our project's unique needs.

In addition, we discussed the datasets crucial for training our deepfake models, elucidating their importance and the roles they play within the broader context of our project. These datasets will act as the foundation upon which our models learn and refine their deepfake capabilities.

This chapter serves as a comprehensive guide, encompassing every vital aspect of our methodology. It outlines the tools and techniques we'll employ as we venture further into the realms of deepfake technology, paving the way for subsequent practical applications and experiments.

In the upcoming chapter, we will put this theoretical groundwork to practical use. By employing the models and methodologies discussed herein, we will conduct a series of experiments aligned with our primary research objective - the real-time application of deepfake technology.

CHAPTER 4

EXPERIMENTAL DESIGN

4.1 Overview

In this chapter, we will be providing a detailed account of the experiments that were conducted to test the validity of our research hypothesis. We will be explaining the code that was used in the experiments, the algorithms employed and the parameters that were set. Furthermore, we will also discuss the results of the experiments, highlighting any significant findings, trends or patterns that were observed in the subsequent chapter. The experiments conducted were designed to test the effectiveness of the proposed methodology in achieving the desired objective. We used a variety of techniques and approaches, ranging from simple heuristics to complex machine learning algorithms, to test the robustness and effectiveness of our proposed solution.

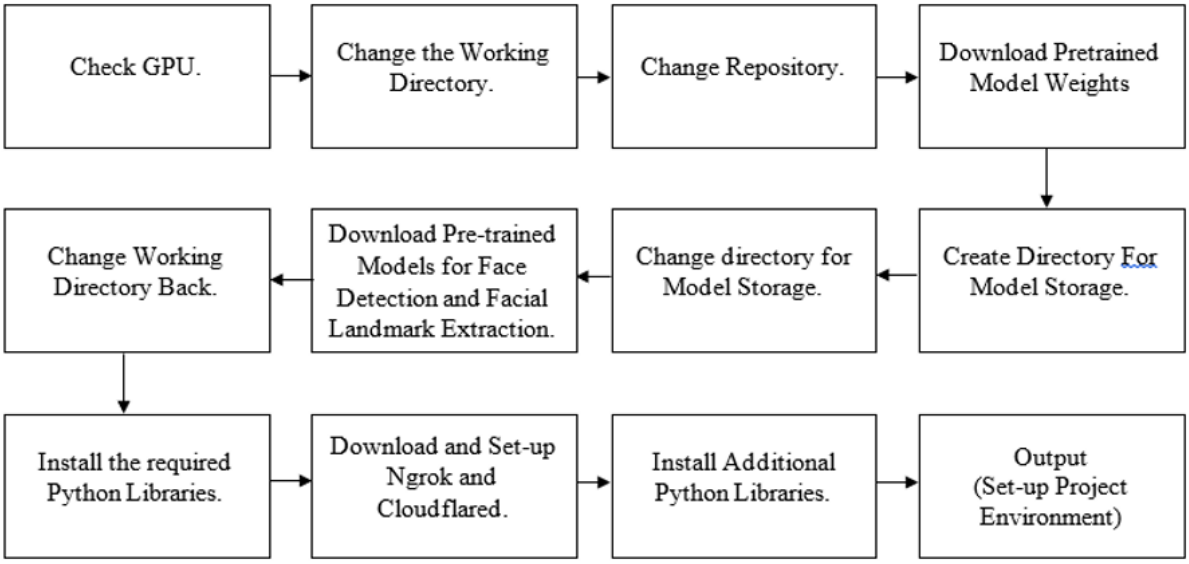
To ensure that our experiments were reliable, we conducted multiple trials under different conditions, varying the parameters and inputs to test the system's resilience to noise and variability. We also used standard metrics to evaluate the performance of the system and compared it against existing state-of-the-art solutions to establish the efficacy of our proposed approach. We will be discussing in detail the algorithms and techniques that were used in the experiments, providing a step-by-step explanation of the code that was used, and highlighting any notable challenges or limitations that we encountered during the process. We will also present the results of the experiments in the form of tables, graphs, and other visual aids, providing a clear and concise interpretation of the data. Overall, this chapter will provide a comprehensive account of the experiments carried out, including the methods, results, and interpretation of findings, providing valuable insights into the feasibility and effectiveness of the proposed methodology.

4.2 Experiment 1 with the First Order Motion Model

4.2.1 Setting up the Environment and Importing Libraries

Setting up the environment and importing libraries is the first step towards achieving the first order motion model. The Mathematical Formulation of the First Order Motion Model (FOMM) serves as a fundamental framework for motion estimation and analysis. This model was used

to represent the displacement of pixels between consecutive frames in the video sequence through a vector field, where each pixel's motion is characterized by a velocity vector. The assumption of constant pixel intensity over time facilitates the formulation of partial differential equations to estimate motion accurately.



The installations and configurations necessary for running the First Order Motion Model for Image Animation with the desired setup was conducted. The model will be used to animate images based on the facial expressions of a driver video, and the installed libraries help support this functionality.

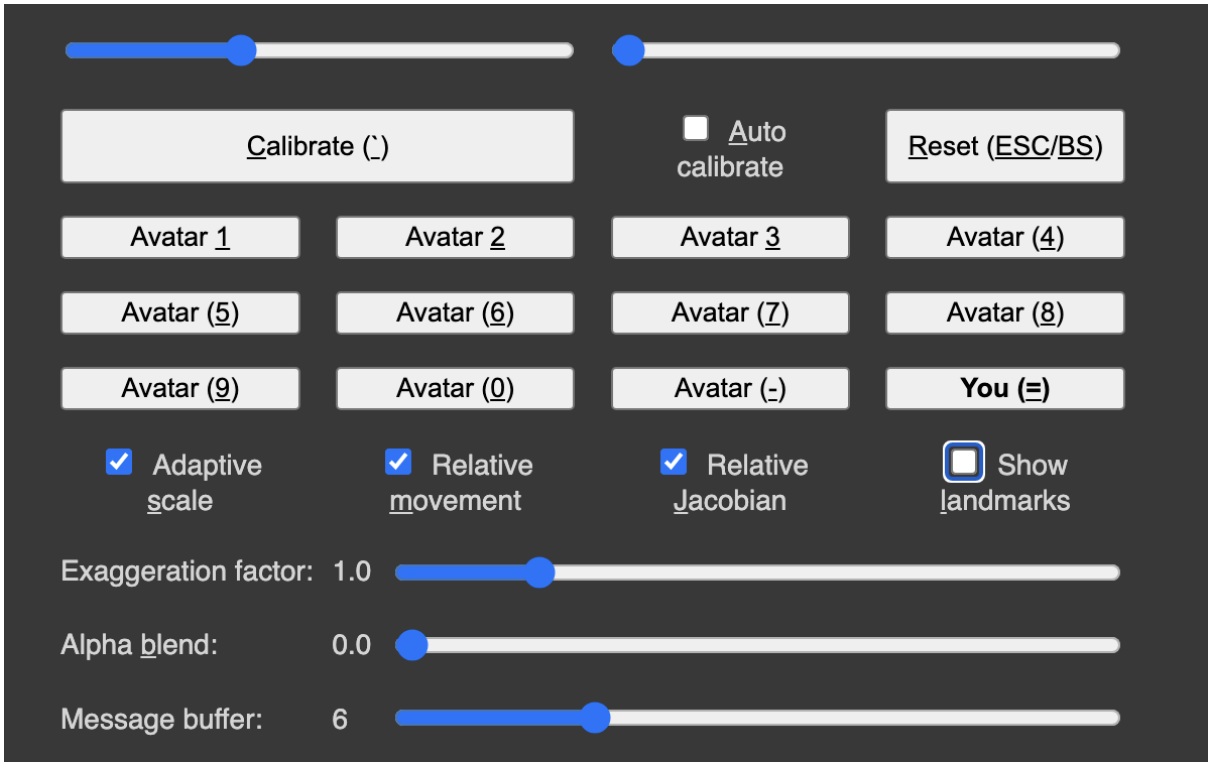


Figure 4 - 1 The interface designed

4.2.2 Importing the Dataset

We used the OpenCV library, which is a popular open-source library used for image processing and computer vision tasks. OpenCV is imported with the alias 'cv2'.

Then, we trying to read the images using the **`imread`** function from the OpenCV library. The **`imread`** function reads an image from a file and stores it in a variable. The file path to the image is passed as an argument to the function. Here, the file paths are **'/content/image1.png'**, **'/content/image2.png', and '/content/image3.png'**. so, the data from each image is preserved.

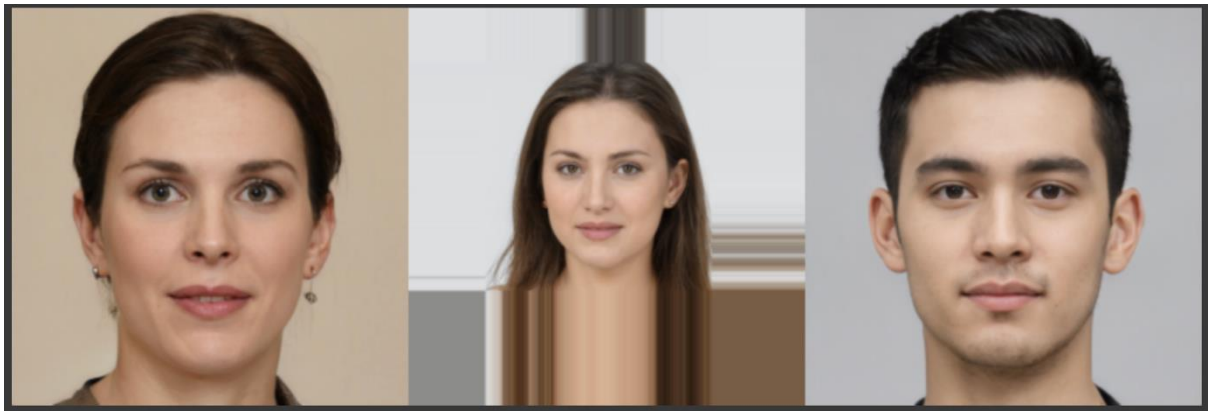


Figure 4 - 2 Test images uploaded

4.2.3 Resizing the images

We performed a series of operations to process and manipulate the set of images. we begin by setting certain parameters for each image, such as whether the image should be centred on the head, whether it should be cropped to the head, and what the expansion factor for cropping should be.

We then import the necessary libraries for image processing, array manipulations, and machine learning. we import `imageio` for reading and writing images, `numpy` for numerical operations, `cv2_imshow` from Google Colab's patches for displaying images, `resize` from skimage's transform for resizing images, and `face_alignment` for detecting facial landmarks.

Next, we initialize the face alignment network `fa`. If the initialization fails (which can happen if the pre-trained model files are not found in the expected directories), the script will delete the cache files and try to initialize the network again.

Then, we define several utility functions such as;

- `create_bounding_box`: This function takes facial landmarks as input and calculates a bounding box around the face. It can also expand the bounding box size by an expansion factor.
- `fix_dims`: This function ensures that an image has the correct number of dimensions and returns a three-channel image.
- `get_crop`: This function takes an image, landmarks, and some additional parameters. It crops the image according to the bounding box around the face and can centre and/or crop the face according to the provided parameters.
- `pad_crop_resize`: This function pads, crops, and resizes an image according to the provided coordinates. If the coordinates fall outside the image boundaries, it pads the image.

Finally, the script reads each image, performs the specified operations (centring, cropping, and resizing), and appends the processed image to the list of source images. The processed images are then displayed side by side.

4.2.4 Applying the Deepfake

We set up and started a secure public URL for their localhost server, specifically for running a face animation model.

1. we start by setting the **`tunnel`** variable **to `'argo'`**, indicating that we're using Cloudflare's Argo Tunnel for creating a public URL for the localhost. Alternatively, we could also use **ngrok**.
2. Then, the script proceeds to stop any previously running **ngrok** or **Argo** tunnel processes. It also attempts to terminate a multiprocessing Pool (if one exists) and close a socket and a server (if they're open).
3. The user then defines the port number as **`6006`**.
4. Depending on the value of **`tunnel`**, the script runs either **ngrok** or **Argo** tunnel in a separate process.
5. The script then checks if the URL of the tunnel has been set up properly, retrying if not.
6. Next, it changes the current directory to where the **"first-order-model"** is located.
7. It then loads pre-trained models for the animation from given paths using a function from the **`demo`** module.
8. A bunch of helper functions are defined, including **`normalize_kp`** to normalize key points, **`norm_source`** to normalize the source image, **`load_stylegan_avatar`** to load and normalize StyleGAN avatars, and **`full_normalize_kp`** and **`make_animation`** for normalization and creation of animations respectively.
9. The script then generates avatars using different APIs and normalizes these images to use them as source images for the face animation.
10. The **`make_animation`** function is the main part of the script. It generates an animation frame given a driving frame (a frame that the source image should imitate). It calculates key points on the driving frame, normalizes them, and then passes the source image, source key points, and normalized driving key points to the generator, which creates and returns a new frame where the source image is animated according to the driving frame.

4.2.5 Applying the Deepfake in Real-time

We created a real-time animation server using Bottle, a lightweight web server microframework for Python.
1. The script first defines several variables: **`avatar`**, **`anti_aliasing`**, and **`save_socket`**. **`avatar`** is initially set to -1, indicating that no avatar is initially selected. **`anti_aliasing`** is set to False, meaning no smoothing will be applied when resizing images. **`save_socket`** will store the WebSocket connection used for real-time data transmission.

2. The script defines a route for the server. The **`@socket.route('/', apply=[websocket])`** decorator creates a route at the server root ('/') where it will apply the WebSocket protocol for real-time communication.
3. In the function **`wsbin(ws)`**, the server continuously waits to receive messages from the client through the WebSocket **`ws`**.
4. We parse messages received from the client and execute commands based on the content of the messages. The commands include:
- Loading a new image to use as an avatar.
- Changing the currently selected avatar.
- Resetting the avatar's animation.
- Adjust various parameters that affect the animation, such as the exaggeration factor, alpha blending factor, and several boolean flags.
5. The command **`make_animation`** is invoked with the given parameters to animate the current frame of the video stream.
6. The resulting image from the animation is converted back to a string in JPEG format and sent back to the client through the WebSocket.
7. The loop inside the **`wsbin`** function continues to run indefinitely until the server stops, or an exception is encountered, making the server continuously ready to receive, process, and respond to messages from the client.
We set up a web server using the Bottle microframework, and it makes use of Gevent's WebSocket for handling real-time communication.
1. A class **`MyGeventWebSocketServer`** is defined, which extends the **`ServerAdapter`** class from the Bottle library. This custom class is used to integrate Gevent's WebSocket server with the Bottle web server.
2. The **`run`** method within **`MyGeventWebSocketServer`** is where the server is started. It takes as arguments the host and port on which to run the server, and a handler which contains the application logic for the server. If the server is not in 'quiet' mode, it sets up a logger to record events.
3. The **`shutdown`** method is used to stop and close the server when it is no longer needed.
4. In the main part of the script, a Pool of processes is set up using the **`multiprocessing`** library. This pool will contain two worker processes.
5. One worker process is launched using the **`apply_async`** function. The **`use_cam`** function (which isn't provided in this code snippet) is passed as the task to be executed by this process. It appears that **`use_cam`** is supposed to connect to a camera, to capture video data, and send

it to an **ngrok** tunnel URL. The second parameter, **`0.8`**, is also passed to **`use_cam`**, but without the function definition, its purpose is unclear.
6. A new **`MyGeventWebSocketServer`** object is instantiated, with the host set to **`'0.0.0.0'`** (meaning the server is reachable at any IPv4 address of the machine) and the port set to a variable **`port`** (which isn't defined in the given code snippet).
7. Finally, the Bottle application, represented by **`socket`**, is run on this server. The **`io.capture_output()`** function from the **`IPython.utils`** module is used as a context manager to suppress any output from the **`socket.run()`** call.

4.3 Experiment 2 with Mediapipe

In the absence of the system and giving that we will not be doing a lot of training on our model as it is going to be applied in real-time. we opted to use Google Colaboratory by Google. Collaboratory or "colab" for short enables us to write and execute Python codes through the browser, and it is well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter Notebook service that requires no setup. It provides access to computing resources, including GPUs, suitable for our work.
As we ran the pipeline in Colab the environment kept on crashing when we were trying to apply the deepfake in real-time using the webcam. We had to test the system using Jupyter Notebook. It was able to run, unfortunately, we were unable to apply the deepfake as the PC in use was so slow. We manage to get an **MSI STEALTH 15M Pc** with **i7 Intel Core, NVIDIA GeForce RTX 3060 6GB GPU, 16GB RAM and 512GB SSD.**

4.3.1 Face Detection Function

We started by importing the **"face_detection"** module from the **"mp.solutions"** package. This module provides a pre-trained machine learning model for detecting human faces in images or videos. The model is trained on a large dataset of faces, and it uses a deep neural network architecture to make the necessary predictions.
Furthermore, we created an instance using the **"FaceDection"** class this is the main class used for performing face detection. The class works with two parameters "model_selection" and **"min_detection_confidence"** that we were able to adjust as follows:

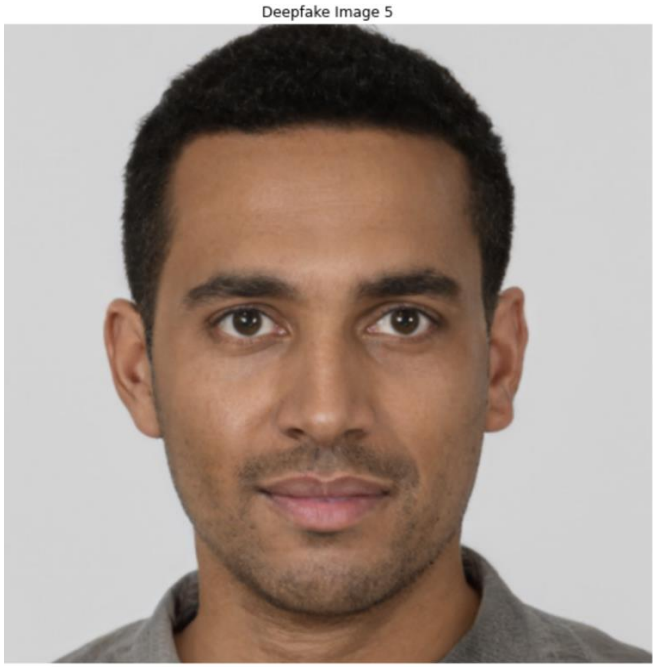
- The **“model_selection”** parameter is used to specify which pre-trained face detection model is to use. In this case it is set to **0**, which corresponds to the general face detection model.
- The **“min_detection_confidence”** parameter specifies the minimum confidence level required for a face detection to be considered valid. The value is set to **0.5,** which means that only the detection with a confidence score of **50%** or higher is returned.

Lastly, the import command was executed to import the “**drawing_utils”** module from the **“mp.solutions”** package. This module provided us with a set of utility functions for drawing the results of the face detection algorithm onto an image or video frame. These functions can be used to visualize the face detection results and to help us with debugging and testing.

4.3.2 Performing Face Detection

The code starts by assigning the output of processing an image named **`test_image_10`** to the variable **`face_detection_results`** using the **`process ()`** method of a face detection object named `face_detection`. The code then checks if any detections were found in the image by verifying that the **`face_detection_results`** object has a non-empty **`detections`** attribute. If detections are present, the code enters a loop that iterates through each detected face and prints some information about it. Inside the loop, the code prints the face number and its confidence score, both of which are attributes of the current face object being processed.

The code then assigns the face location data to a variable named **`face_data`** and prints the relative bounding box of the face, which is a property of the **`face_data`** object. Finally, the code loops through the first two key points of the face using a range of 2 and prints the name of the key point and its relative position, both of which are attributes of the **`face_data`** object. The **`mp_face_detection.FaceKeyPoint`** class is used to access the name and value of each key point.




```
FACE NUMBER: 1
--------------------------------
FACE CONFIDENCE: 0.96

FACE BOUNDING BOX:
xmin: 0.20574143528938293
ymin: 0.3122577965259552
width: 0.5897252559661865
height: 0.5908468961715698

RIGHT_EYE:
x: 0.3848162591457367
y: 0.4768412113189697

LEFT_EYE:
x: 0.6299744844436646
y: 0.48496368527412415
```

Figure 4 - 3 Face Detection

4.3.3 Face Landmark

To get the face landmark, we first created a copy of our target images named **“Test_image 1 to 10” and "deepfake_image 1 to 5"** 1 using NumPy’s **"copy"** method. The copied image is stored in a new variable as **“Test_image__copy”** and **"deepfake_image__copy."** Respectively.

The next line of code checks if there are any face detections in "**face_detection_results_test**" and "**face_detection_results_deepfake**." If there are any detections, the code loops through

each detection using a "for" loop. Inside the loop, the "**mp_drawing.draw_detection**" function is called with three arguments - the "**Test_image__copy**" or "**deepfake_image__copy**" image, the current face detection, and the drawing specification for the key points. This function draws a bounding box around the detected face and adds key points (landmarks) to the face. The key points are drawn with a red colour and have a thickness and circle radius of 2 pixels.

After the loop, the modified image "**Test_image__copy**" or "**deepfake_image__copy**" is displayed using matplotlib's "**imshow**" function. The function also creates a new figure with a size of 10x10 using the "**fig = plt.figure(figsize=[10, 10])**" line of code. Finally, the image is shown on the screen using the "**plt.show()**" function.

Overall, the code is processing an image with face detection results using MediaPipe and then draws bounding boxes and key points around the detected faces in the image. The modified image is then displayed on the screen.

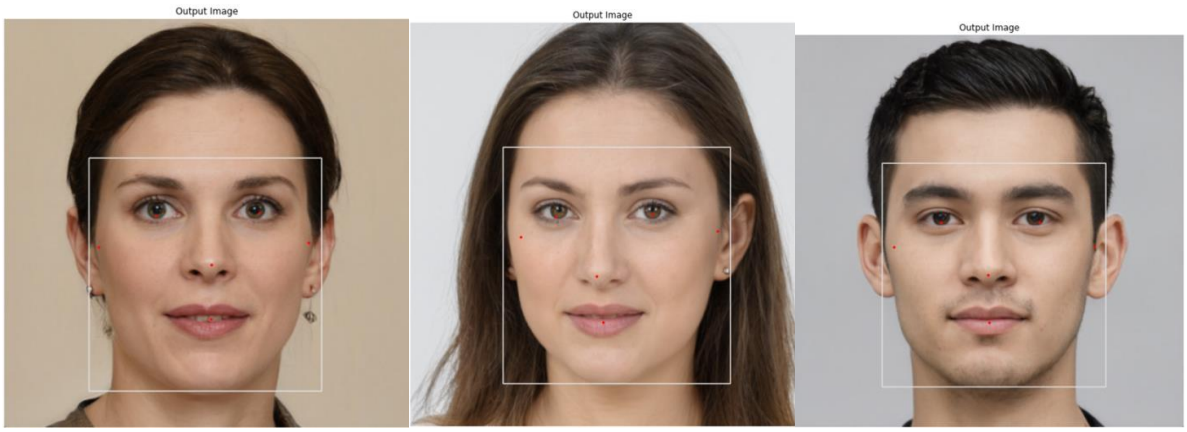


Figure 4 - 4 Face Landmark detection

4.3.4 Creating A Face Landmark Detection

In our attempt to create a face landmark detection we defined a function called **`detectFacialLandmarks`** that takes in three arguments, **`image`**, **`face_mesh`**, and **`display`**.

The **`image`** parameter is an image file that contains a face or faces that need to be detected.

The **`face_mesh`** parameter is an instance of the MediaPipe Face Mesh model that will be used to detect the facial landmarks on the faces in the image.

The **`display`** parameter is an optional Boolean argument that is set to **`True`** by default. If set to **`True`**, the function will display the output image that shows the detected landmarks on the face.

The function first processes the image using the **`face_mesh`** model to detect the facial landmarks on the face(s) in the image. It then creates a copy of the input image in **`output_image`**.

If the **`results`** object returned by the **`face_mesh`** model has **`multi_face_landmarks`**, indicating that there are multiple faces in the image, the function loops through each **`face_landmarks`** and draws the landmarks on the **`output_image`** using **`mp_drawing.draw_landmarks()`** function.

The function first draws the tessellation of the facial mesh using the **`FACEMESH_TESSELATION`** argument, and then the facial contours using the **`FACEMESH_CONTOURS`** argument.

Finally, if **`display`** is set to **`True`**, the function displays a plot with two images side by side. The first is the original input image, and the second is the **`output_image`** with the detected landmarks drawn on it. If **`display`** is set to **`False`**, the function returns the **`output_image`** as a NumPy array and the **`results`** object.

4.3.5 Perform Face Landmark Detection on The Datasets

The above code is a Python script that loads an image from a file, passes it to a function named **"detectFacialLandmarks,"** and sets the display flag to True to show the output.

The first line of code loads an image from the file path **"Datasets/Test/Test1.jpg"** using OpenCV's **"cv2.imread"** function and stores it in the variable "image." This creates a NumPy array that represents the image in memory. This Process was repeated throughout our datasets.

The second line of code calls the **"detectFacialLandmarks"** function and passes in the loaded image as the "image" argument and another argument **"face_mesh_images,"** which is presumably an instance of the Mediapipe Face Mesh solution. The **"display"** flag is set to True, which means that the output will be displayed on the screen.

The **"detectFacialLandmarks"** function processes the input image with MediaPipe Face Mesh and draws landmarks on any detected faces. The output is a modified version of the input image with facial landmarks overlaid on the faces in the image. If the display flag is True, the modified image is displayed on the screen in a new figure.

Overall, the code aims to demonstrate the use of the Mediapipe Face Mesh solution to detect facial landmarks and illustrate the output on an image.

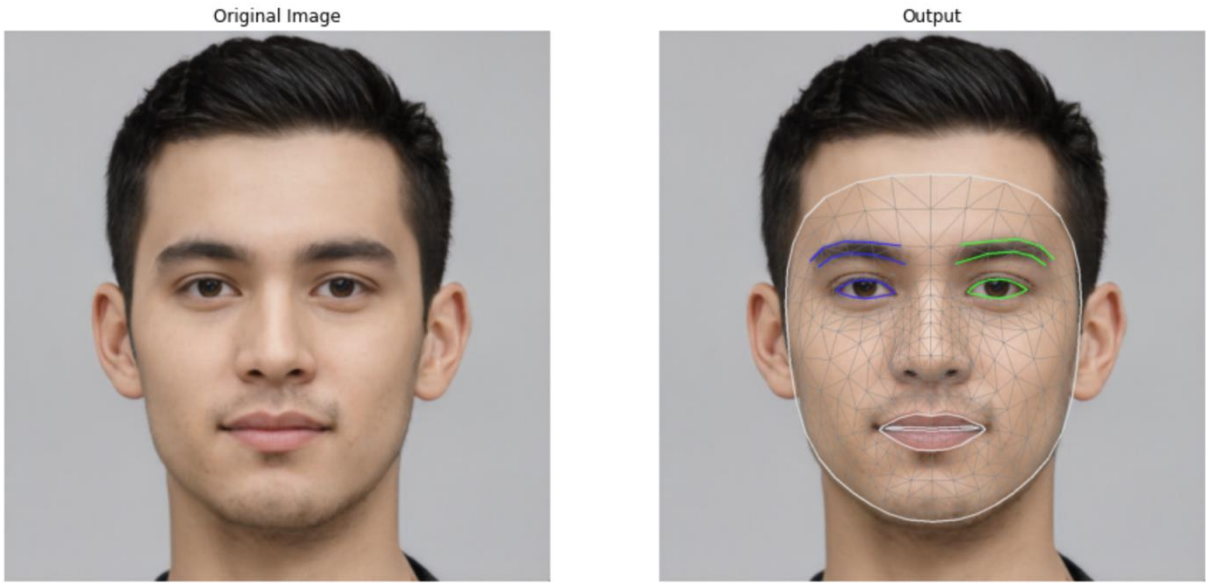


Figure 4 - 5 Face Landmark Detection in Real-time

4.3.6 Face Landmark Detection in Real-Time

We start by initializing a camera video capture object using the OpenCV library. Argument 0 is used to capture video from the default camera of the system.

Followed by setting the width and height of the video capture frame to 1280 and 960, respectively.

The **'cv2.namedWindow()'** function creates a window named '**Face Landmarks Detection**' with a normal window size.

The **'while'** loop iterates continuously until the video capture is open. The **'camera_video.isOpened()'** function checks if the camera is open.

Inside the loop, the **'camera_video.read()'** function reads the current frame from the video capture object. The **'ok'** variable checks if the frame is successfully read. If it's not successfully read, the loop continues.

The **'cv2.flip()'** function is used to flip the frame horizontally. This is done to get a mirror image of the face as the camera usually captures a reversed image of the face.

The **'detectFacialLandmarks()'** function is called with the current frame, **'face_mesh_videos**', and **'display=False'** as arguments. This function detects facial landmarks on the face in the current frame. The returned values are assigned to the **'frame, _'** variables.

The **'time()'** function is called to measure the time taken to detect the facial landmarks. The **'time2'** variable stores the current time.

If the difference between the current time and the previous time is greater than 0, the frames per second (FPS) is calculated as the reciprocal of the time difference. The FPS value is displayed on the frame using the **'cv2.putText()'** function.

The **'time1'** variable is updated with the current time for the next iteration.

The current frame with the facial landmarks detected is displayed on the 'Face Landmarks Detection' window using the **'cv2.imshow()'** function.

The **'cv2.waitKey()'** function waits for a key press event for 1 millisecond. The **'& 0xFF'** operation is used to mask the non-ASCII character values in the keyboard input. The **'27'** key is used to break out of the loop if it is pressed.

Finally, the video capture object is released using the **'camera_video.release()'** function, and all the OpenCV windows are closed using the **'cv2.destroyAllWindows()'** function.

4.3.7 Deepfake Application

A function called `overlay` that takes in the following parameters, is first defined:

1. **`image`**: This parameter is a NumPy array representing the original image on which the deepfake image will be overlaid.
2. **`deepfake_image`**: This parameter is a NumPy array representing the deepfake image that will be overlaid onto the original image.
3. **`face_landmarks`**: This parameter is a dictionary that contains the facial landmarks of the face region to which the deepfake image will be applied.
4. **`face_part`**: This parameter specifies the part of the face to which the deepfake image will be applied.
5. **`INDEXES`**: This parameter is a dictionary containing the indexes that correspond to the different parts of the face, e.g., 'right_eyebrow', 'left_eye', etc.
6. **`display`**: This parameter specifies whether the resultant image should be displayed or not. By default, it is set to True.

The function starts by creating a copy of the original image and assigning it to a variable called **`annotated_image`**. The **`try`** block is used to handle any exceptions that may arise while executing the code.

The **`getSize`** function is called to obtain the height, landmarks, and face part height of the face region specified by the `face_landmarks` and **`face_part`** parameters. The **`required_height`** variable is calculated as 2.5 times the face part height.

The **`deepfake_image`** is resized to match the required height and width of the face region. The threshold is set to 25, and the inverse binary threshold is applied to the resized deepfake image to obtain the **`deepfake_image_mask`**.

The centre of the face part is calculated using the mean of the landmarks. A region of interest (ROI) is extracted from the original image using the location of the face part and the dimensions of the deepfake image. The **`deepfake_image_mask`** is applied to the ROI using the **`cv2.bitwise_and()`** function to obtain the resultant image.

The deepfake image is then added to the resultant image using the **`cv2.add()`** function. Finally, the resultant image is assigned to the corresponding location in the **`annotated_image`** array.

If **`display`** is True, the **`annotated_image`** is displayed using **`matplotlib`**. Otherwise, the **`annotated_image`** is returned.

In summary, the **`overlay`** function applies a deepfake image to a specified face region of an original image by resizing the deepfake image to match the face part's size, creating a mask using the deepfake image, and applying the mask to the corresponding ROI in the original image. The deepfake image is then added to the resultant image, which is assigned to the corresponding location in the **`annotated_image`** array.

4.3.8 Deepfake Application in Real-time

The program reads from the webcam, detects facial landmarks, overlays an image based on these landmarks, and displays the frame. Here is a brief explanation of the important elements in the code:

1**. `camera_video = cv2.VideoCapture(0)`**: Initializes a VideoCapture object to read from the webcam. Argument 0 indicates the default webcam of choice.

2. **`camera_video.set(3,1280)`** and **`camera_video.set(4,960)`**: These lines set the video resolution to 1280x960.

3. `cv2.namedWindow('Face Filter', cv2.WINDOW_NORMAL)`: Creates a named window where the frames from the webcam will be displayed.

4. **`face_filter = cv2.imread('media/face_filter.png')`**: Reads a face filter image that will be overlaid on the detected face.

5. The while loop **`while camera_video.isOpened():`** runs until manually closes the program, capturing and processing one frame at a time.

6. The **`detectFacialLandmarks(frame, face_mesh_videos, display=False)`** function is supposed to detect facial landmarks in the frame.

7. If landmarks are detected, the face filter is overlaid on the frame at the appropriate location using the **`overlay(frame, face_filter, landmarks, display=False)`** function.

8. The frame, now with the face filter, is then displayed on the screen.

9. The program waits for the '**ESC**' key to be pressed. If it's pressed, the while loop breaks and the program ends.

10. **`camera_video.release()`** and **`cv2.destroyAllWindows()`** close the camera feed and the created window, cleaning up the resources.

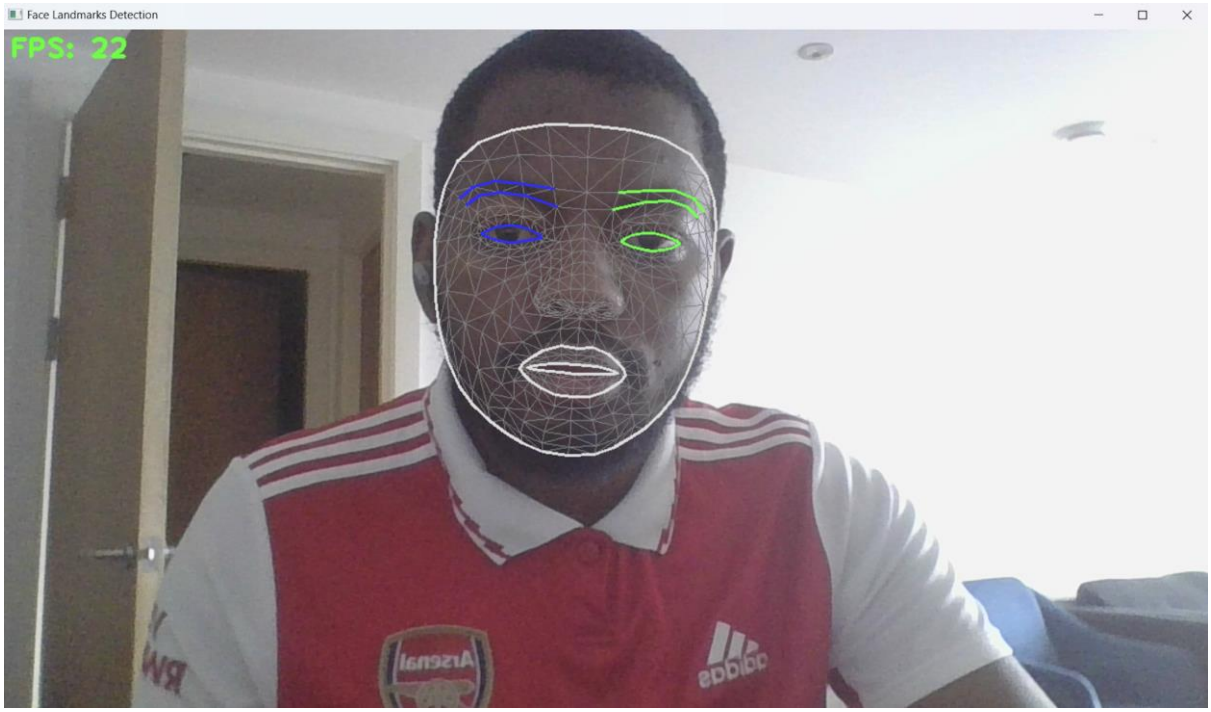


Figure 4 - 6 Application of Mediapipe

4.4 Experiment 3 with the DeepfaceLab

We were able to utilise the same system that we implemented the second model with **MSI STEALTH 15M Pc** with **i7 Intel Core, NVIDIA GeForce RTX 3060 6GB GPU, 16GB RAM and 512GB SSD.** Through this research, we learned the more processing the more system needed.

4.4.1 Camera Source

The `CameraSource` class is a PyQt-based graphical interface facilitating interaction with a back-end camera source Indicating its role as a front-end representation of the back-end CameraSource. This class includes several GUI elements corresponding to the control sheet properties of the CameraSource backend object:

1. **`q_driver`**, **`q_device_idx`**, **`q_resolution`**, **`q_fps`**, **`q_rotation`**, **`q_flip_horizontal`**: These interactive elements enable users to adjust camera parameters such as the driver, device index, resolution, frames per second, rotation, and horizontal flip, respectively.
2. **`q_open_settings`**, **`q_load_settings`**, **`q_save_settings`**: These buttons allow users to open comprehensive settings, load prior settings, and save current settings, respectively.

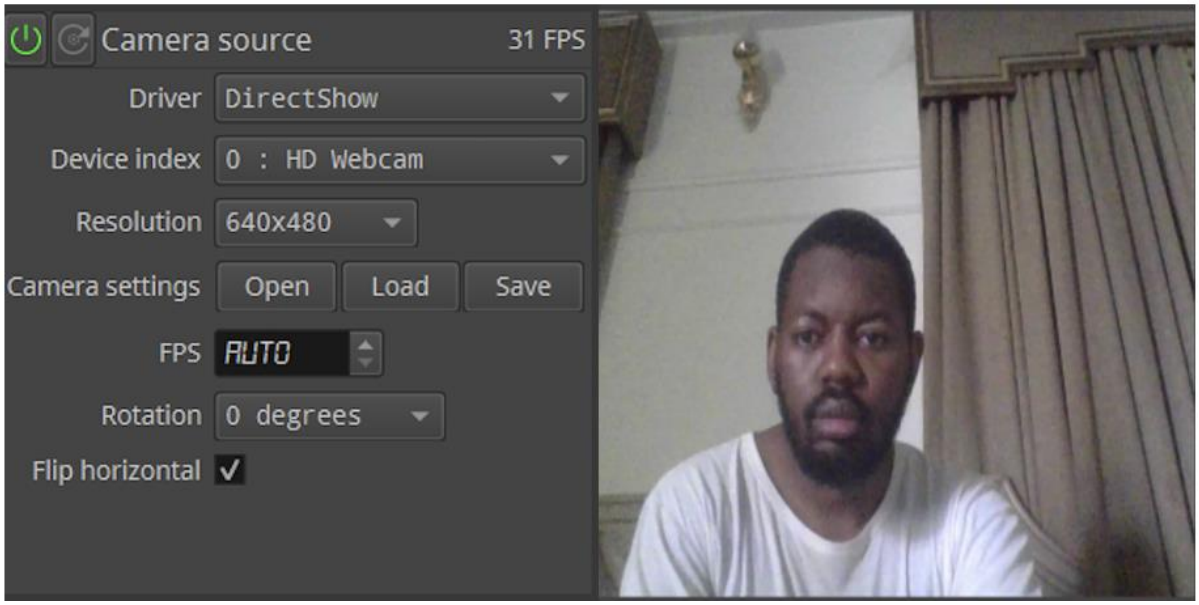


Figure 4 - 7 Camera Source UI and Outcome

Each GUI element has an associated `LabelPopupInfo` for user-friendly labels. The PyQt's **`XGridLayout`** organizes these GUI elements into a grid-like structure. The class methods initialize the UI, set text labels, link the UI with the backend, and arrange the elements into the layout. The parent class is then initialized with **`super().__init__`**. Overall, **`CameraSource`** illustrates the application of PyQt for designing a robust, interactive interface for a complex backend system.

4.4.2 Face Detector

The **`FaceDetector`** offers a graphical user interface (GUI) to interact with the backend **`FaceDetector`**. This GUI is designed to allow users to adjust various parameters pertaining to face detection in an image.

The class contains several attributes:

1. A reference to the backend **`FaceDetector`**, checked for validity in the constructor.
2. A timer that calls the **`_on_timer_10ms`** method every 10 milliseconds.
3. **`LabelPopupInfo`** and associated widgets for each adjustable parameter, including the type of detector, device, window size, detection threshold, maximum number of faces, sorting order, and temporal smoothing.

Upon initialization, a layout grid is constructed and populated with these labels and widgets, forming the user interface. The widget updates are tied to a corresponding control sheet, allowing real-time modification of the backend face detector's settings.

The **`_on_backend_state_change`** method is called when the backend state changes. If the backend is stopped, the coordinates of the detected faces are cleared.

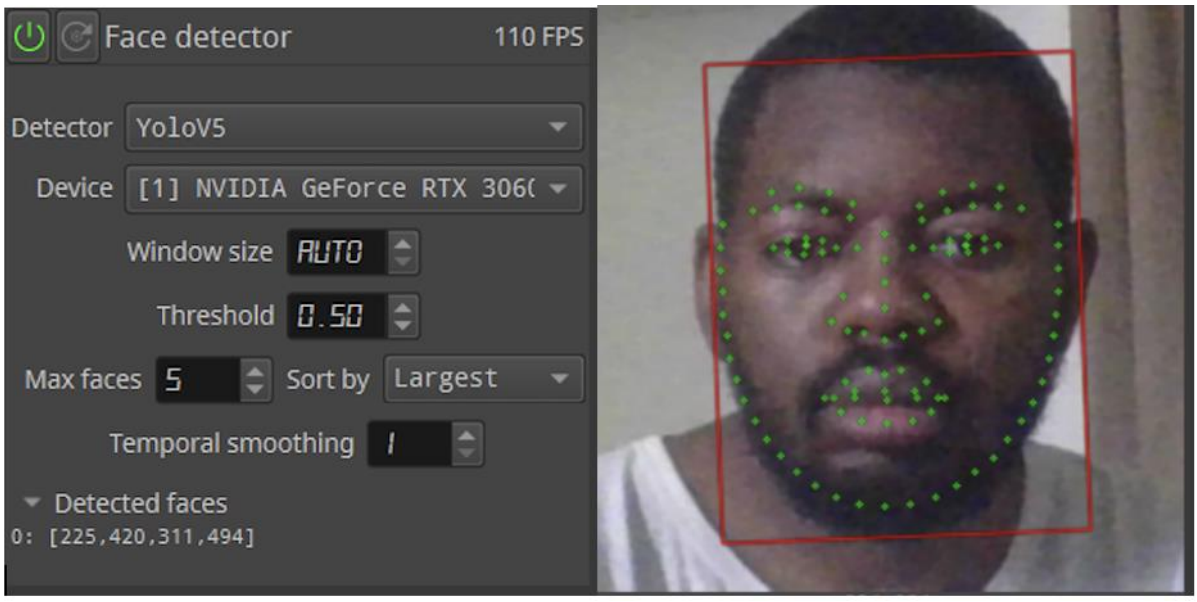


Figure 4 - 8 Face Detector UI and the outcome

The **`_on_timer_10ms`** method, which is triggered every 10 milliseconds, updates the information displayed on the GUI. When the collapsible **`q_detected_faces`** section is opened and the window is not minimized, this method retrieves the current output of the backend and updates the face coordinates label with the latest positions of the detected faces. It accomplishes

this by extracting face swap information for each face from the backend output, converting this information to a string format, and setting this string as the text of the **`q_face_coords_label`**. In sum, the `QFaceDetector` class serves as a user-friendly interface for the **`FaceDetector`** backend, allowing users to easily modify parameters and view the results of face detection.

4.4.3 Face Marker

The **'FaceMarker'** has a similar structure as the **`FaceDetector`**, with the GUI comprising **`LabelPopupInfo`** and corresponding widgets for each adjustable parameter. The adjustable parameters for the `FaceMarker` include the marker type, device, marker coverage, and temporal smoothing. These parameters are organized in a grid layout **(`grid_l`)**, with each parameter and its corresponding widget occupying one row. The marker coverage and temporal smoothing parameters and their widgets are further organized into a sub-grid layout **(`sub_grid_l`)**, which is added to the main grid layout in the third row.
Each widget is connected to a corresponding control sheet, which allows real-time modification of the backend face marker's settings.

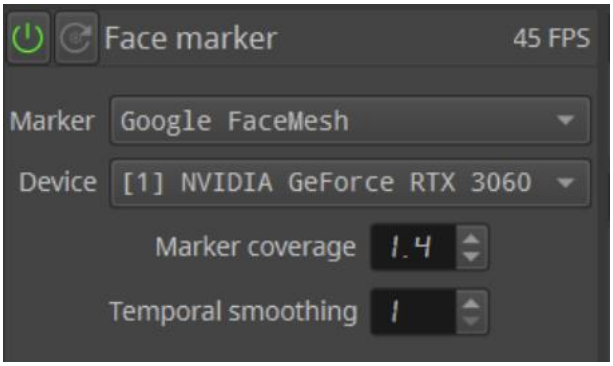


Figure 4 - 9 the Face marker UI

Upon initialization, the **`FaceMarker`** class populates the layout grid with these labels and widgets, forming the user interface. This class does not have any additional methods to handle backend state changes or timed events, unlike **`FaceDetector`**. In summary, `FaceMarker` provides a user-friendly interface to manipulate the face marking process, which typically involves marking key facial landmarks on the detected faces for further processing.

4.4.4 Face Merger

The face merger is typically the last step in a deepfake pipeline, where the processed face from the source image is merged into the target image. The parameters of this process include the face offset in the x and y direction, face scale, face mask type, face mask erode, face mask blur, colour transfer, interpolation, colour compression, and face opacity.

In the GUI, these parameters are represented by labels **(`QLabelPopupInfo`)**, and widgets which can be either a **`SpinBoxCSWNumber`**, **`SliderCSWNumber`**, **`CheckBoxCSWFlag`**, or a **`ComboBoxCSWDynamicSingleSwitch`**. Each widget is connected to a corresponding control sheet, which allows real-time modification of the backend face merger's settings.

The labels and widgets are organized in a grid layout **(`grid_l`)**, with each parameter and its corresponding widget occupying one row. Some parameters such as face x and y offsets, face mask source, celeb, landmarks, face mask erode, and blur are further organized into a horizontal layout **(`QXHBoxLayout`)**, with labels and widgets placed side by side.

Upon initialization, the `FaceMerger` populates the layout grid with these labels and widgets, forming the user interface.

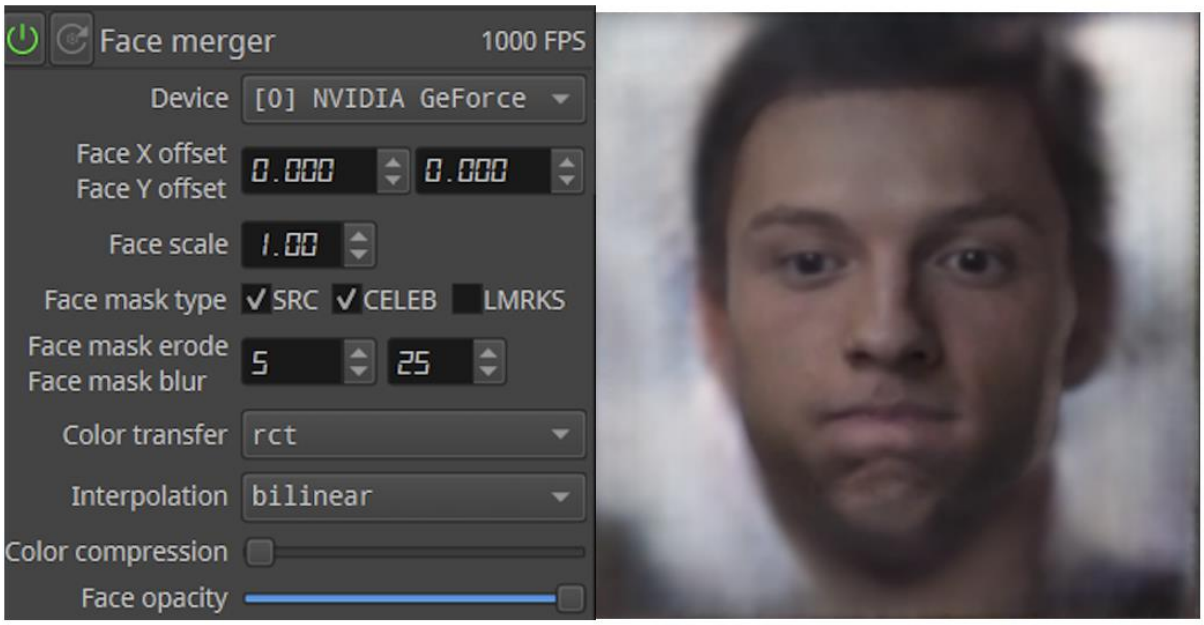


Figure 4 - 10 Face merger UI and outcome

It provided us with the interface to tweak the parameters of the face merging process to achieve the desired output.

4.4.5 Output

`Output' panel for controlling some kind of stream output functionality. Here's a brief overview:

1. The constructor of `Output` takes an instance of the `Output` class as an argument. It initializes various Qt-based UI components using both built-in PyQt5 widgets and some custom ones, according to the backend's control sheet (`cs`).
2. Various components are instantiated, including labels, combo boxes, push buttons, spin boxes, a path editor, an error message display, and a checkbox. The labels use the localization function `L` to handle multi-language support.
3. Most of the components seem to be linked with control sheet parameters. For instance, `q_source_type` is linked with `cs.source_type`, and any change in the UI widget is likely to update this parameter. The widgets have labels and tooltips attached to them.
4. The UI components are then organized and added into a `QXGridLayout`, which is a custom grid layout. The layout handles the positioning of the widgets on the GUI. The grid layout is then passed to the `QBackendPanel` superclass during initialization.
5. The streaming address and port are handled specially at the end of the layout.

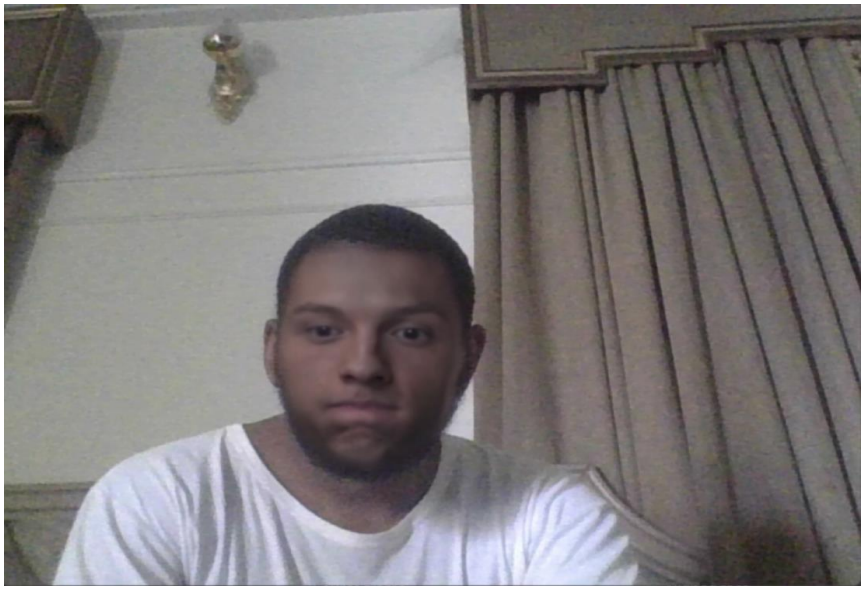


Figure 4 - 11 Output

4.5 Summary

In this chapter, we've provided a detailed step-by-step account of the experiments conducted as part of our research. We began with the initial experiment involving the use of the first-order

motion model, followed by a subsequent experiment utilizing the MediaPipe framework. These carefully designed experiments form the crux of our practical exploration into the world of deepfake technology. The first-order motion model offered insights into the fundamental mechanics of image manipulation, while the MediaPipe framework brought in more advanced capabilities for real-time applications. Looking forward, the next chapter is dedicated to analysing the results obtained from these experiments. We will delve into a thorough discussion of our findings, deciphering what the data tells us about the performance and potential of our chosen methodologies. This examination will not only shed light on the outcomes of our current experiments but will also guide us in refining our future research strategy, ensuring that we continually build upon our understanding and application of deepfake technology. Our ultimate goal is to derive meaningful conclusions from our results that will contribute significantly to this evolving field of study.

CHAPTER 5

RESULT ANALYSIS AND DISCUSSION

5.1 Overview

In this chapter, we turn our focus to the outcomes of our research, examining the results produced by our various models. Our primary objective in this discussion is to unravel the implications of these results and understand what they reveal about the effectiveness and utility of our chosen methodologies. We will meticulously analyse the output from each model, highlighting the key aspects that determine their performance. In doing so, we will gain a deeper understanding of how well our models have performed in the context of our objectives and the broader landscape of deepfake technology in real-time.

Moreover, this chapter will also entail a comparative discussion of our models' outcomes. By juxtaposing these results, we will be able to identify the strengths and weaknesses of each model, giving us a comprehensive perspective on their relative performance. Such a comparison will not only shed light on which model performs best under certain conditions but will also inform our decisions for future model selection and optimization. This analytic and comparative approach will allow us to draw meaningful insights from our models, moving us closer to our ultimate goal: the development and application of effective, efficient, and responsible deepfake technology in real-time.

5.2 Result Analysis

5.2.1 Experiment 1 with the first order motion model

5.2.1.1 Front

In the following analysis, we present the outcome of our deepfake application implemented in real-time. The figure below provides a side-by-side comparison of the source material and the generated output. This comparison allows for a clear visual assessment of the transformation achieved through our deepfake model. In our evaluation, it is apparent that the deepfake aligns exceptionally well with the source, maintaining its integrity without any perceptible disruptions.

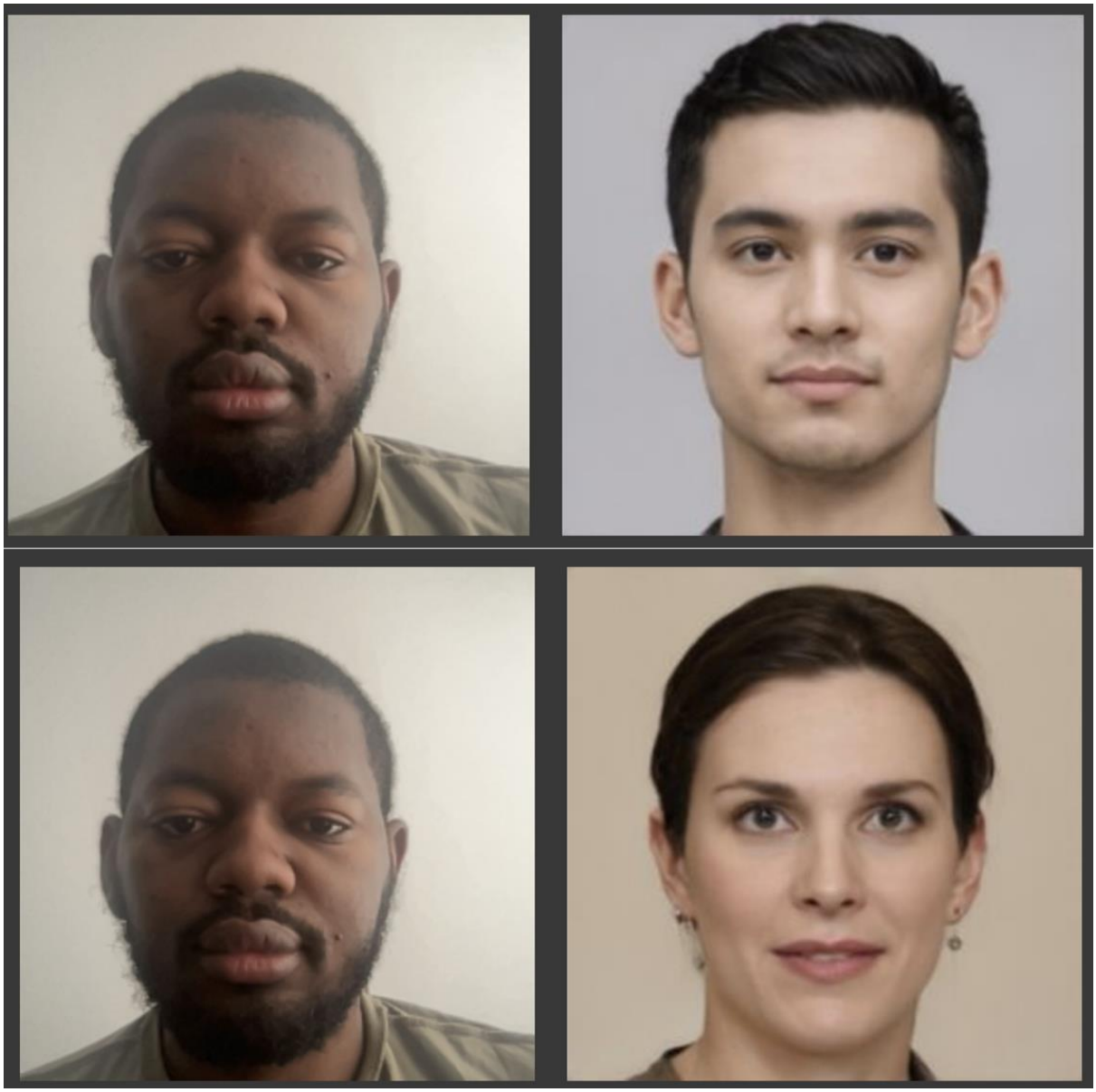


Figure 5 - 1 front comparison of the source and output

5.2.1.2Right-side

Turning our attention to the right-side comparison, we encounter a contrasting result. Unlike the seamlessly integrated deepfake presented earlier, here the deepfake fails to maintain its cohesion and falls apart.

While at first glance this might seem like a shortcoming, it's important to interpret this result in the context of our broader goal - de-identification for privacy preservation and data protection. In terms of achieving anonymity and preserving privacy, the breaking down of the deepfake serves to further obfuscate the original identity, essentially meeting the objective of de-identification for privacy preservation and data protection.

Figure 5 - 2 Right-side comparison of the source input and the output

5.2.1.3 Left-side

Similar to our observations from the right-side comparison, the left-side comparison also yields results that, on the surface, might be perceived as less successful. However, much like in the right-side instance, it's essential to view these outcomes in light of our overarching objective - de-identification for privacy preservation and data protection.

Figure 5 - 3 left-side comparison of the source input and the output.

5.2.1.4 Relative movement

The results of the frontal application incorporating a relative movement function was obtained. This technique is designed to mirror the facial markers of the source input within the output, aiming for a dynamic replication effect.

As shown in the figure below, the use of this function results in the output replicating the long-faced characteristics of the source input. However, it's evident that the implementation of the function doesn't produce a flawless replica, the cleanliness of the final output leaves room for improvement.

The deepfake model's attempt to adjust to the long-faced feature of the source adds an additional layer of complexity to the mapping and replication process. The model is not just

tracing the movements in the source input, but also the particular facial features, in this case, the longer face. This additional challenge seems to have introduced some degree of distortion into the output.

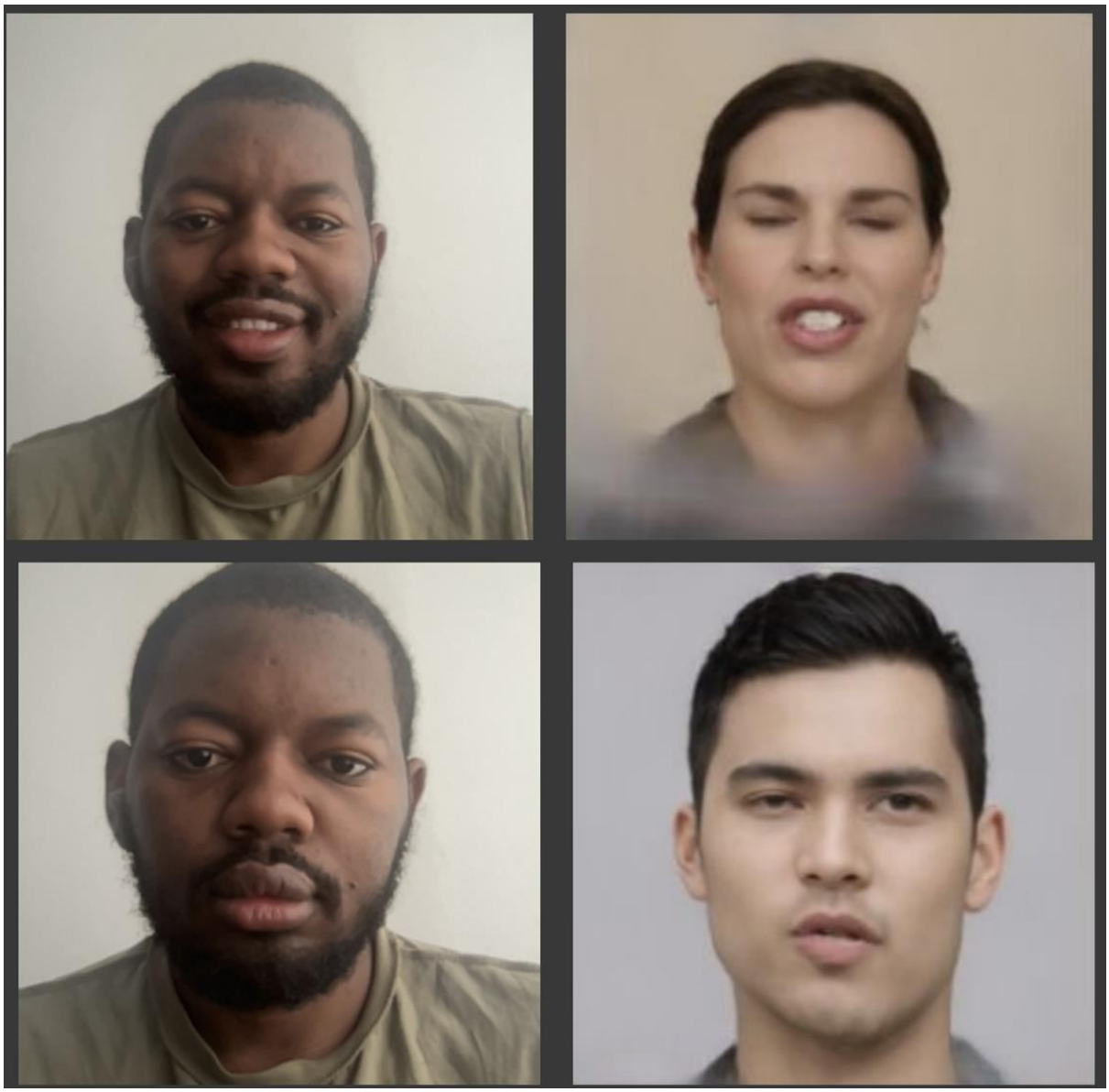


Figure 5 - 4 side-by-side comparison of the source input and the output with relative movement function

5.2.1.5 Overall Model performance

In light of our experimentation, we attribute the disintegration of the deepfake technology, as witnessed in certain applications, to insufficient training of the designed model. The priority given to achieving optimal processing speed and real-time application may have compromised the thoroughness of the model's training phase, leading to the observed flaws in certain outcomes.

	Real-time	Multiple faces	Source Background	De-identification	Face Expressions
Model	√	**X**	**X**	√	√

Table 5 - 1 The performance of the model.

Despite the imperfect results, it is crucial to emphasize the invaluable insights derived from this experimentation process. These insights underscore the intricate complexities that surface when the deepfake model attempts to accurately replicate both dynamic facial movements and distinctive facial features from the source input in real-time. Moreover, they shed light on the nuanced balance between processing speed, real-time application, and the overall integrity of the deepfake outcome. Crucially, these insights pave the way for further enhancements to the model, highlighting areas requiring attention and improvement. This continuous process of learning, adapting, and evolving is intrinsic to the pursuit of excellence in the domain of deepfake technology, pushing the boundaries of what is currently achievable in this fascinating field.

5.2.2 Experiment 2 with Mediapipe

Contrasting with the model discussed earlier, the upcoming model provides a structure that allows us to individually verify each stage of the process. This modular approach offers us the ability to test and validate the functionality of each segment of our model in isolation.

5.2.2.1 The Face Detection

First and foremost, our methodology necessitates the accurate execution of face detection. This critical first step establishes the foundation upon which the entire model is built, and the precision at this stage significantly influences the subsequent processes. It's crucial to ascertain whether the system can perform this task effectively, as it determines the feasibility of the whole model.

As we implemented face detection, we found that the system exhibited a commendable level of accuracy across our entire image database. This successful outcome confirmed the system's capability to effectively identify and capture facial features within images, which is the bedrock

of the deepfake process. The evidence of this successful face detection is clearly illustrated in Figure 5 - 5 below.

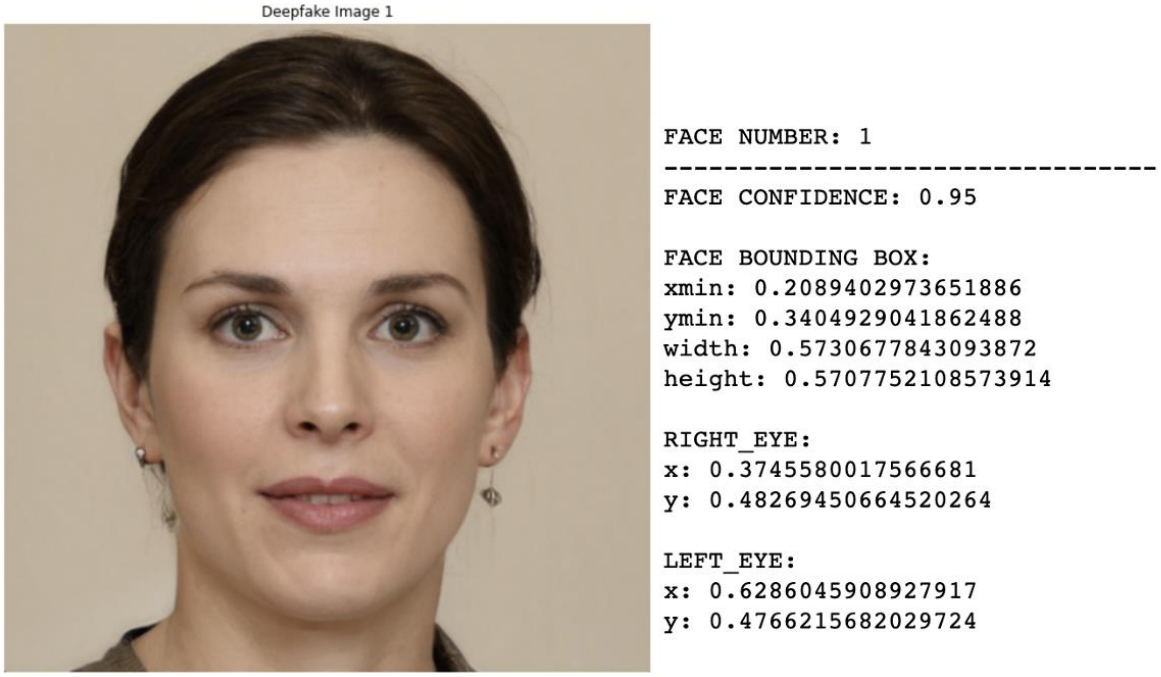


Figure 5 - 5 Single image face detection

Building upon the initial success in face detection, the subsequent aim was to determine whether the system could accurately identify multiple faces within a single image. This capability is pivotal for scenarios involving multiple individuals, and ensuring its functionality is crucial for a comprehensive and versatile deepfake model.

As depicted in Figures 5 and 6, the model underwent testing on an image featuring two distinct faces. The results were highly encouraging; both faces were successfully identified with a high degree of confidence - 98% for the first face and 93% for the second. These outcomes confirm

the model's ability to detect multiple faces within a single image, thereby broadening its range of applicability significantly.

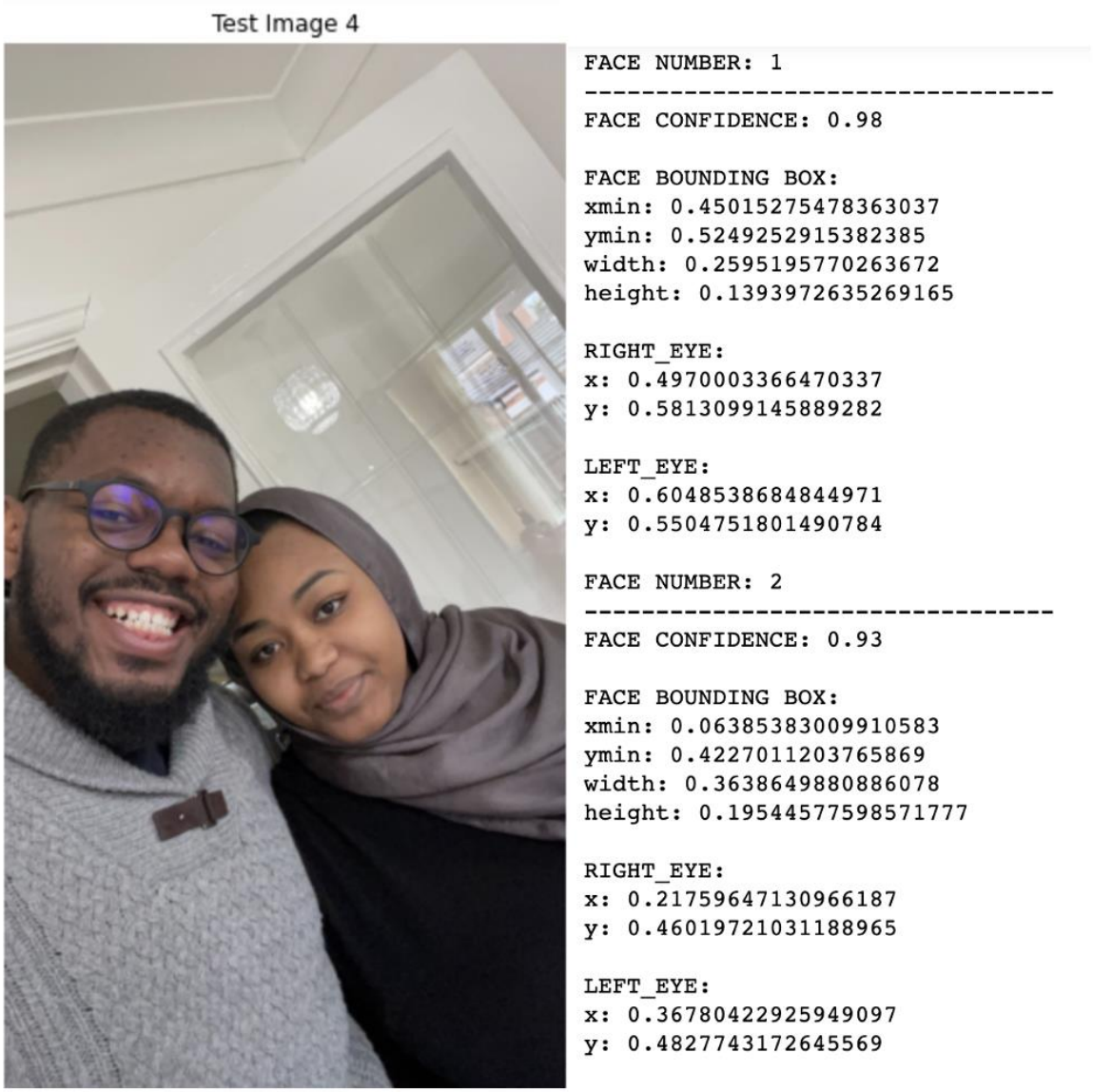


Figure 5 - 6 Two faces face detection.

Following the successful test of detecting two faces within a single image, the decision was made to increase the complexity by introducing an image containing three distinct faces, as shown in Figure 5 - 7. The results, while not as high as in the two-face detection test, still proved promising. Interestingly, these trials revealed a correlation between the distance of faces from the camera and the confidence of the detection. The confidence level decreased as the faces were further away from the camera, with a confidence of 93% for the first face and 85% for the second and third faces respectively. This valuable insight enables a better understanding

of the limitations and variables impacting the model, informing future refinements and considerations in system deployment.

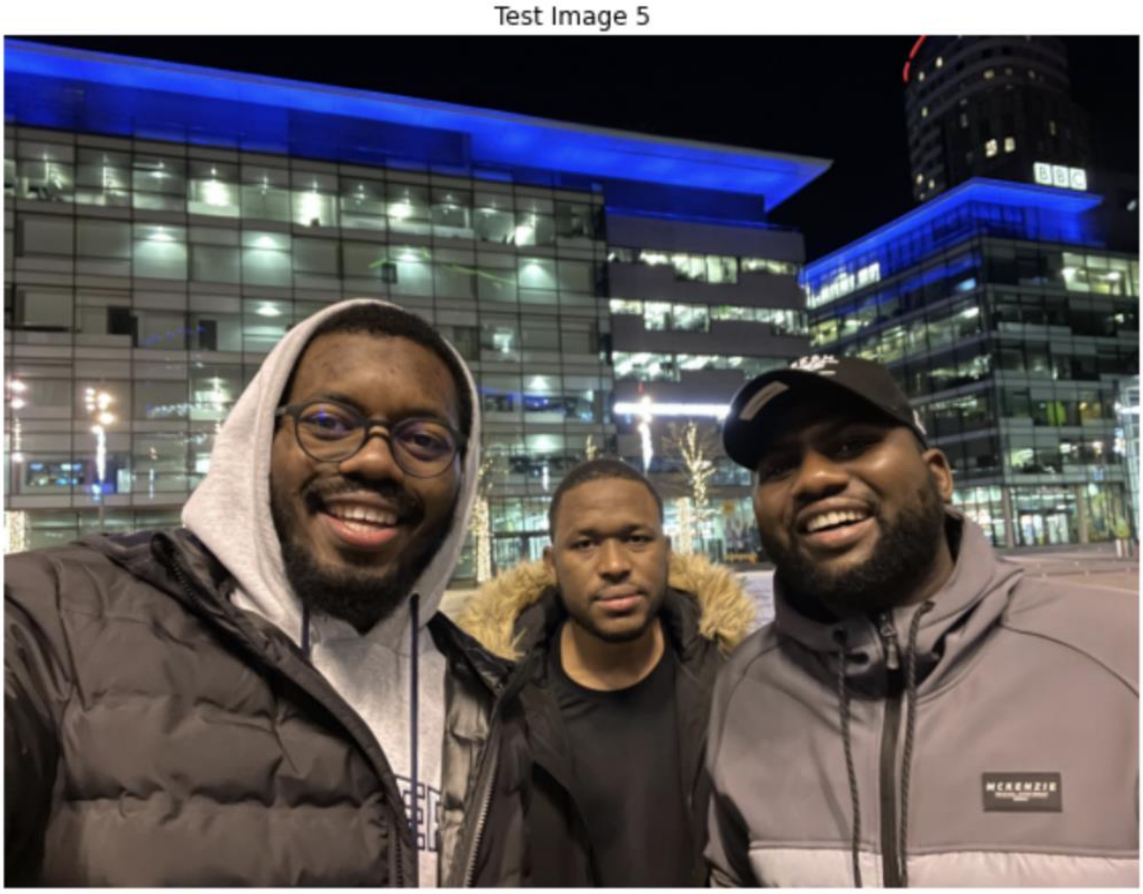


Figure 5 - 7 Three faces face detection

Beyond simply identifying faces within an image, the face detection system initiates advanced computational calculations, establishing the foundations for further processing. For instance, it automatically determines the dimensions of the bounding box for each identified face, an essential step for localized processing of facial features. Moreover, the system begins pinpointing specific facial features, such as the location of the eyes on each identified face.

This detailed analysis not only demonstrates the sophistication of the model but also lays the groundwork for subsequent stages where these data points become essential for tasks such as facial movement tracking and accurate deepfake creation.

5.2.2.2 Bonding Box

In this section, the outcomes of bounding box calculations applied to the referenced images will be demonstrated. These visual representations will illuminate the level of precision the system has achieved in identifying and framing facial features within diverse image contexts.

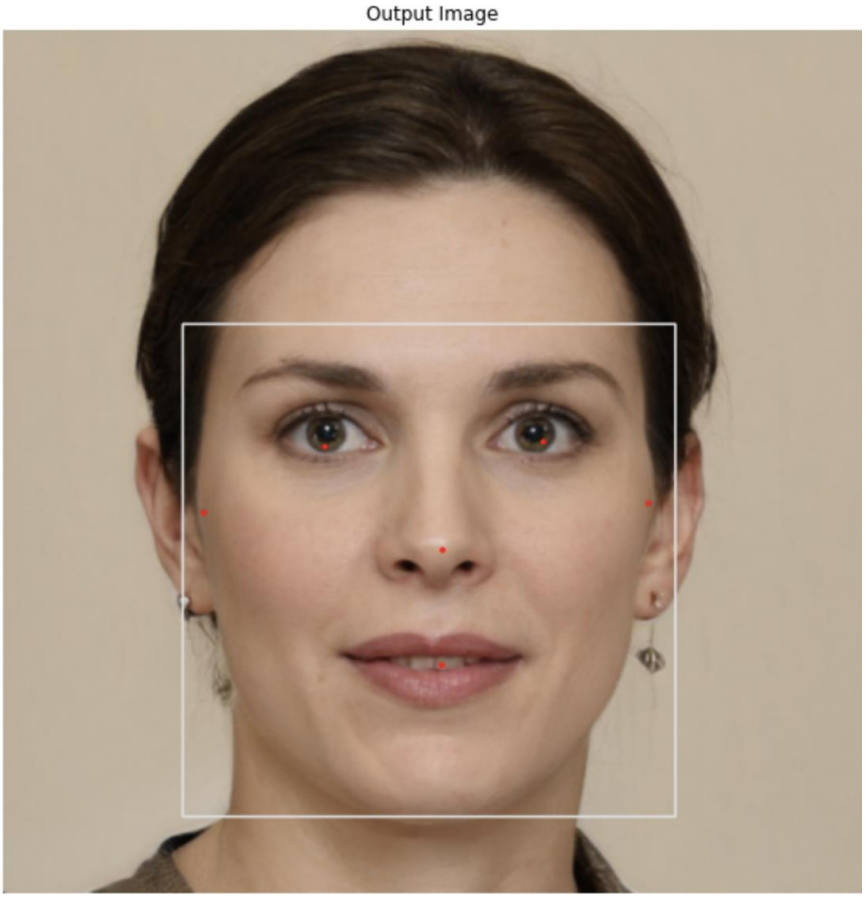


Figure 5 - 8 Single image bonding box

Firstly, Figure 5 - 8 presents the application of a bounding box on an image featuring a single face. This demonstrates the basic function of the developed model in processing an isolated facial image.

Figure 5 - 9 Two faces Bonding box

Subsequently, we'll present Figure 5 - 9, which showcases the system's capacity to identify and create bounding boxes around two distinct faces within a single image. This illustrates the model's ability to process images with two faces.

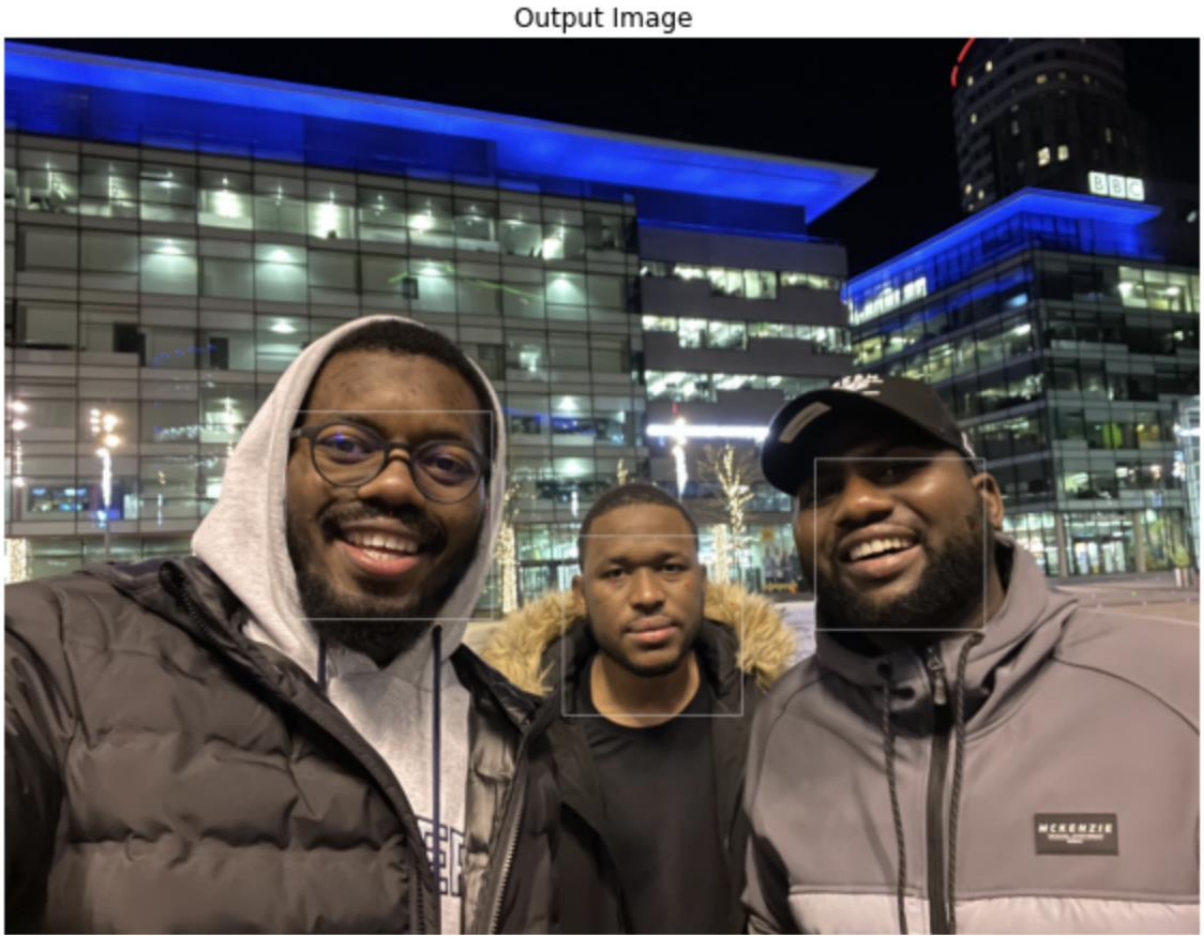


Figure 5 - 10 Three faces Bonding box

Finally, the complexity increases in Figure 5 - 10, which exhibits the result of applying bounding boxes to an image containing three faces. The bounding boxes highlight the accuracy of the model in detecting and differentiating multiple faces, even in more complex scenarios.

5.2.2.3 Face detection

Having conducted facial detection tests with bounding boxes, the evaluation shifted to testing the system's ability to process facial images using Mediapipe, a framework specifically designed for machine learning-based solutions. These tests aimed to provide a side-by-side comparison of the original image and the system's output, enabling the assessment of the accuracy and efficiency of the deepfake model. Each image underwent processing using

Mediapipe to detect and mark facial landmarks, an essential process for creating realistic deepfakes.

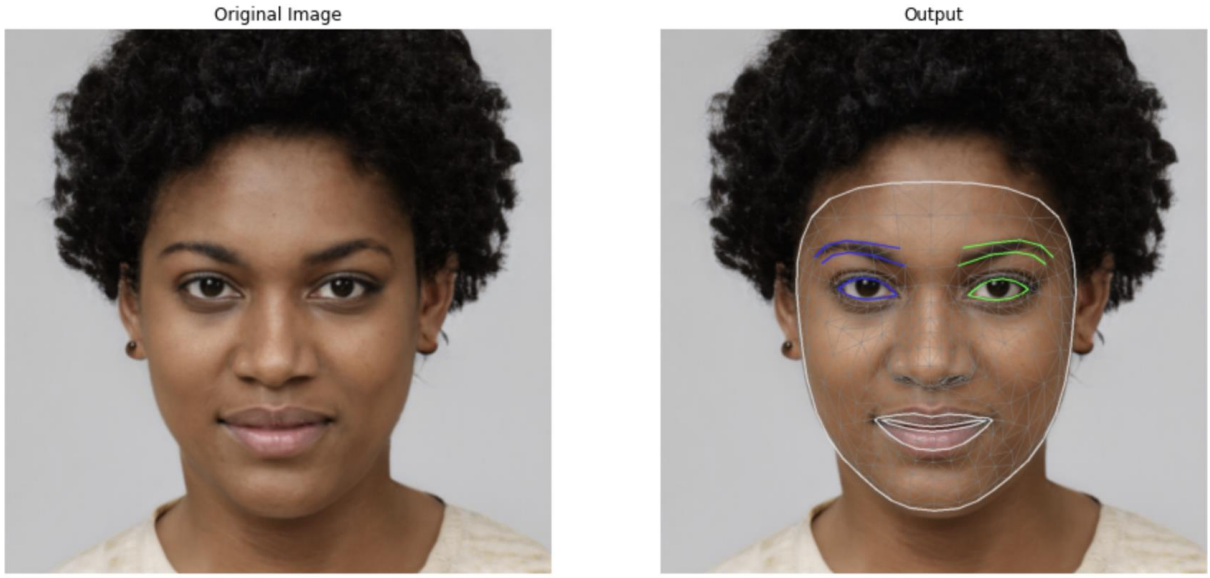


Figure 5 - 11 Single face detection with Mediapipe

Firstly, Figure 5 - 11 depicts the result of applying Mediapipe to a single-face image. This example offers a straightforward illustration of the system's ability to map facial landmarks accurately.

Figure 5 - 12 Face detection of two faces with Mediapipe

Figure 5 - 12 demonstrates the model's capability in handling an image with two faces. The application of Mediapipe to a dual-face image is slightly more complex, challenging the system to identify and differentiate between multiple faces.

Figure 5 - 13 Face detection of three faces with Mediapipe

Lastly, Figure 5 - 13 shows the system's handling of an image featuring three faces. This trial pushes the model further, revealing its capability to accurately map facial landmarks on multiple faces in a more intricate scenario.

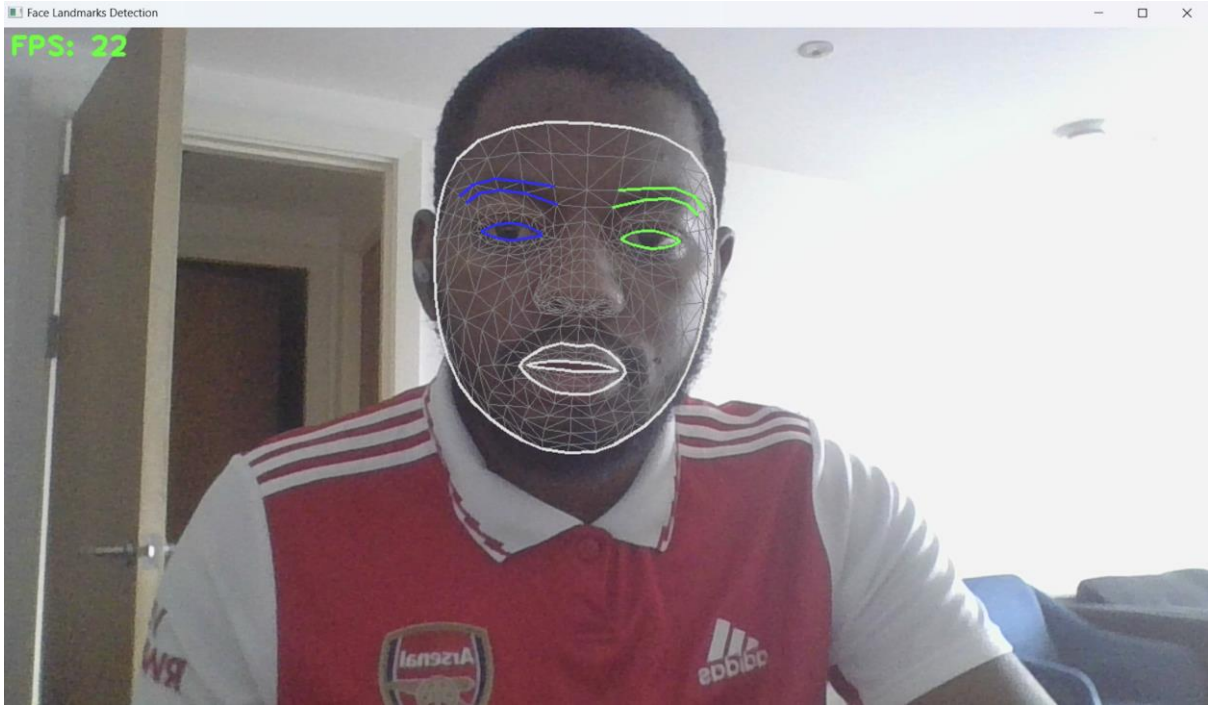


Figure 5 - 14 Face Detection in real-time with Mediapipe

Figure 5 – 14 displays the outcomes of real-time face detection using the Mediapipe framework, emphasizing the model's capacity to process live feeds accurately and efficiently. This image illustrates the system's performance when applied to real-time scenarios, demonstrating its ability to detect and map facial landmarks swiftly and accurately in a dynamic environment. This real-time demonstration not only showcases the proficiency of the model in managing fast-changing conditions but also underscores the potential applications of the model in real-time situations.

5.2.2.4 Overall Model Performance

The overall model performance as shown in Table 5.4 depict that model performed in real time with convincing source background, deidentification and facial landmark expressions but not so well when it comes to multiple faces in a single image.

	Real-time	Multiple faces	Source Background	De-identification	Face Expressions

Model	√	**X**	√	√	**X**

Table 5 - 2 Overall performance of the second model

5.2.3 Experiment 3 with DeepfaceLab

The final model experimented with had a distinct edge over previous models because of its Graphical User Interface (GUI), which provided real-time updates at every stage of the process. This feature significantly enhanced the user experience by offering visual feedback and confirmation of the system's operations, as detailed in the four stages below.

5.2.3.1 Source Input

The model we utilized allowed for flexible source input selection, adding an extra layer of customizability to its operation. In our case, we selected the built-in HD webcam of our device as the input source, as shown in Figure 5 – 15.

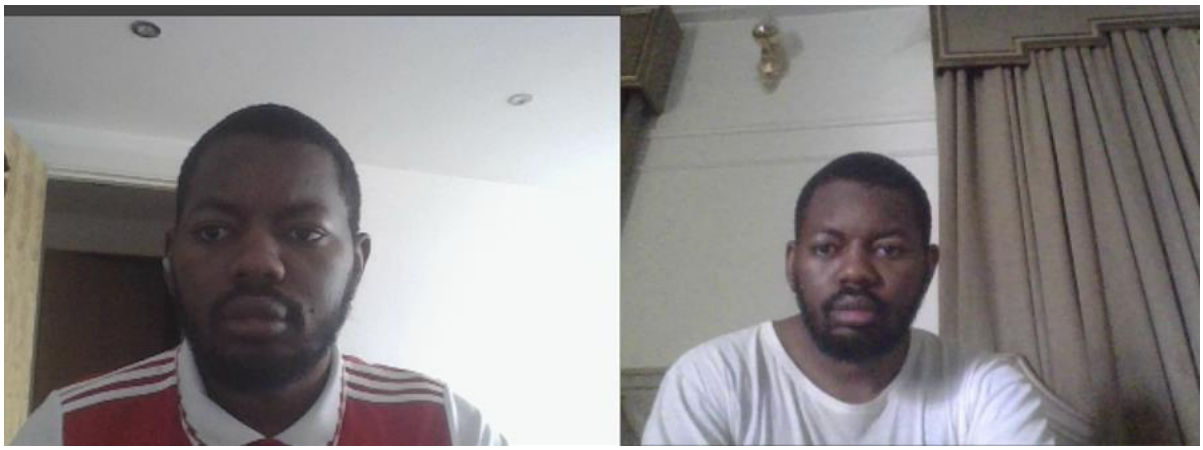


Figure 5 - 15 Source Input

This feature is particularly useful when running the application on a setup with multiple cameras. The model lists all available cameras, giving users the autonomy to select their preferred one. This flexible functionality is essential for ensuring the model can adapt to varying user needs and equipment setups. This step can be seen as a preliminary one before the model begins its operation, effectively linking the chosen camera to the deepfake application. Once the camera source is confirmed, the model can then start the process of face detection, face merging, and eventually generating the deepfake in real-time.

5.2.3.2 Face Detector

The next step in the model involved the implementation of YOLOv5 for face detection from the chosen input source. YOLOv5 is an object detection model, known for its accuracy and

computational efficiency. In our application, it was used to identify and locate faces within the frame captured by the webcam.

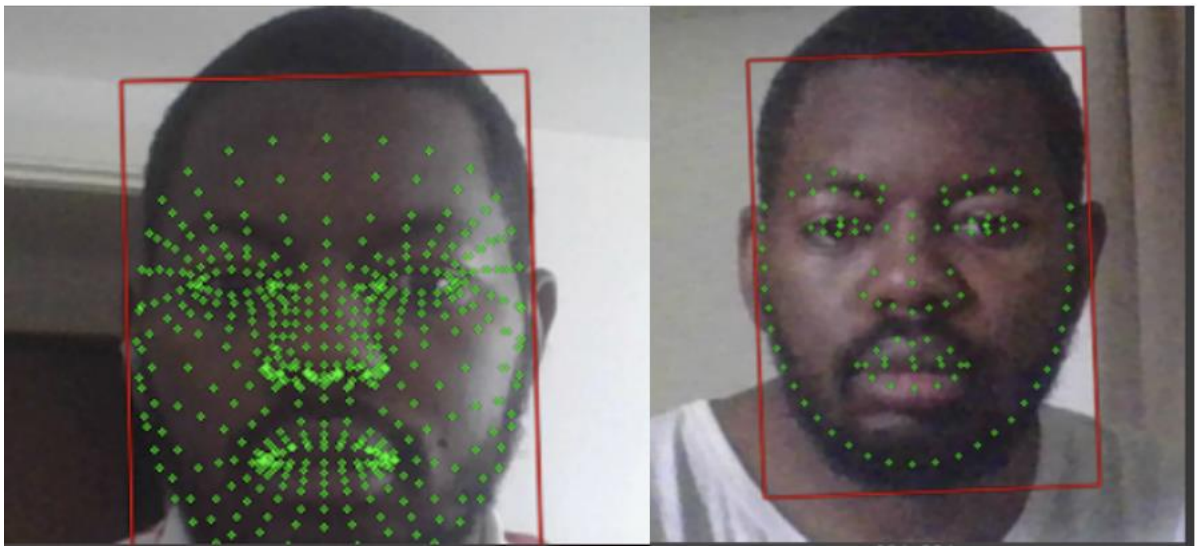


Figure 5 - 16 face Detector

As depicted in Figure 5-16, the model successfully detected a face within the image. However, attempts at multi-face detection were unsuccessful with this particular model. It is crucial to note that while the model demonstrated high proficiency in single-face detection, its current configuration appeared to be less effective in simultaneously identifying multiple faces in the same frame. These findings suggest that while YOLOv5 is a powerful tool for single-face detection in real-time deepfake applications, further refinement or a different approach may be necessary to reliably detect and process multiple faces concurrently. This information will be highly valuable in guiding future work in this area.

5.2.3.3 **Face Merger**

Following the selection of the preferred deepfake from the dataset, the model proceeds to download the corresponding pre-rendered model. Subsequently, the face merger creation process begins, where the deepfake model attempts to integrate and blend seamlessly with the detected face from the input source. During this process, the system strategically targets areas identified through face detection, effectively isolating the face from the rest of the background.

This focus ensures that the deepfake application is strictly limited to the intended area, enhancing the realism of the final result.

Figure 5 - 17 Face Merger

In this stage, the system essentially grafts the deepfake onto the source face, aligning and blending features as closely as possible to achieve a convincing deepfake outcome. Figure 5-17 provides a visual representation of this process, where the deepfake is actively applied to the source face. This operation is a crucial part of the system, as it's responsible for the realistic synthesis of the deepfake with the real face.

5.2.3.4 ***Output***

Once the face merger process, which can be quite time-consuming, is complete, the model proceeds to the final stage - output generation. At this juncture, the system can manifest the deepfake application in real-time. The output window emerges after the processes stated above in the program, enabling immediate witnessing of the deepfake in action. It displays the source input with the successfully applied deepfake, effectively demonstrating real-time de-identification technique. This constitutes the primary achievement of the work, illustrating the feasibility and practicality of real-time deepfake technology for privacy preservation and data protection, as shown in Figure 5 - 18 below.

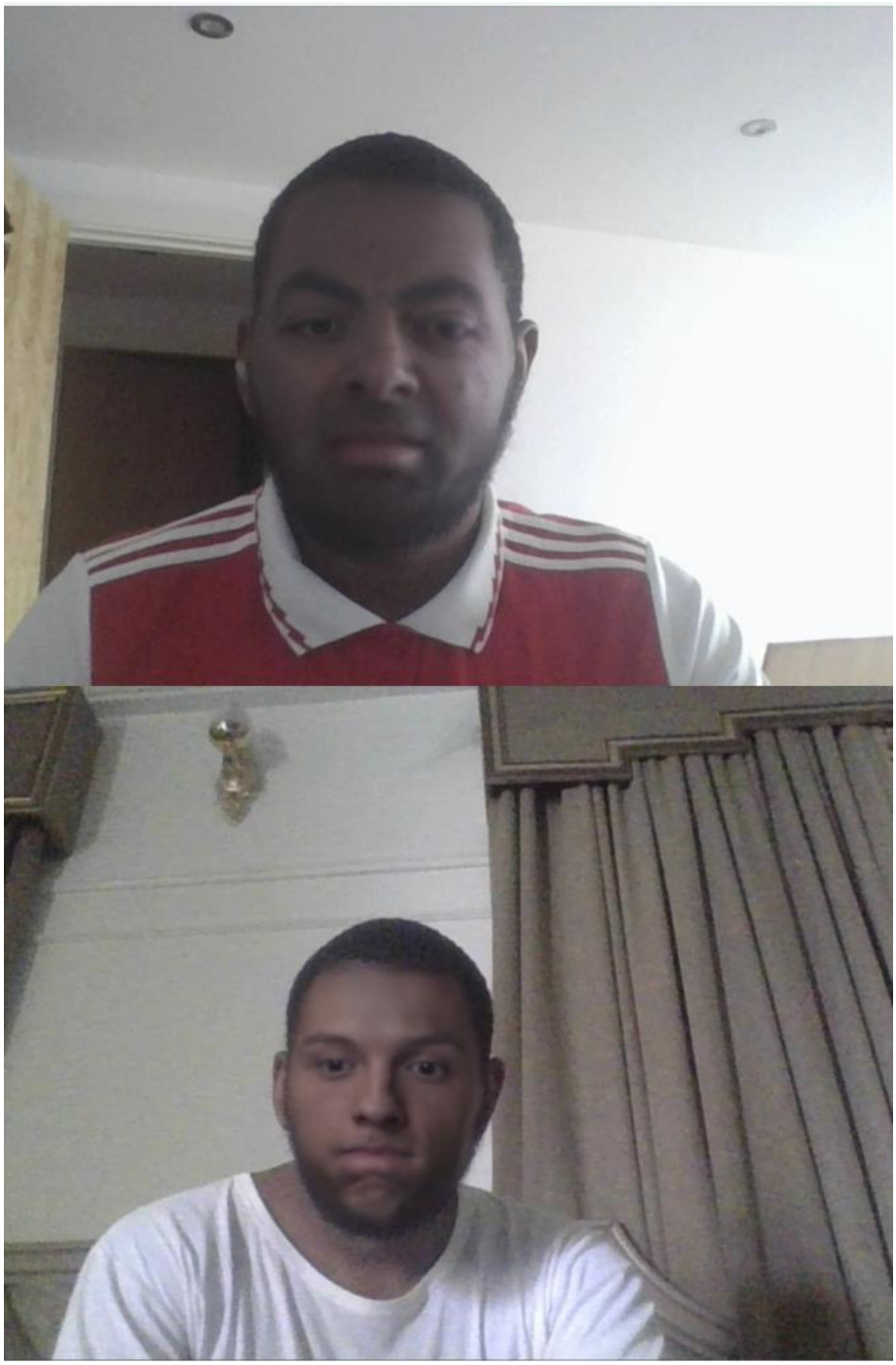


Figure 5 - 18 Final output

5.2.3.5 ***Overall model performance***

The exploration of this model has offered a wealth of insights, highlighting both its strengths and limitations. We found that this model leans heavily towards post-processing rather than real-time processing. The process of downloading the trained face model is a significant aspect of its operations, necessitating ample GPU capacity for smooth functioning.

That being said, the model excels in creating a convincing deepfake overlay. It successfully provides an output, showcasing its capacity for effective facial transformations. However, a key limitation that emerged was its inability to detect and apply the deepfake to more than one face simultaneously.

This limitation points towards an area for future development and improvement. Enhancing the model's multi-face detection capabilities could significantly broaden its potential applications, opening up new avenues for exploration in the field of real-time deepfakes.

Despite its shortcomings, the model's performance underpins the enormous potential that deepfake technology holds for privacy preservation and data protection. Future iterations of the model can continue to build upon these findings, striving towards a robust, efficient, and versatile deepfake solution for real-time applications.

	Real-time	Multiple faces	Source Background	De-identification	Face Expressions
Model	√	**X**	√	√	√

Table 5 - 3 Overall performance of the third model

5.3 Result Comparison

All three models performed real-time tasks successfully as expected. For all cases of single or multi faces image, the three models perfomed excellently while they all failed to perform optimally for multiple faces in a single image. The source background of model 2 and 3 were better obtained compared to model 1. In terms of de-indetification, the three models successfully accomplished the tasks with model 1failing facial expressions. The performance analysis is as summarized in table below.

	Model 1	*Model 2*	*Model 3*
Real-time	√	√	√
Multiple faces	**X**	**X**	**X**
Source Background	**X**	√	√
De-identification	√	√	√
Blended face swap	**X**	**X**	√
Output pop-out	**X**	√	√
Face Expressions	√	**X**	√

Table 5 - 4 the models' comparison

.

5.4 Discussion

Through the comprehensive discussion, it becomes evident that each model tested in the experiments exhibits unique strengths and weaknesses, contributing to a deeper understanding of the challenges and possibilities in the realm of real-time deepfake technology. Here are some key insights gleaned from the analysis:

Experiment with First Order: Its closed-source nature hindered attempts at model improvement. However, it underscored the importance of a large and diverse dataset for the

driving source, facilitating a well-blended model. Despite limitations in GPU capabilities on Google Colab, the model successfully applied deepfake as a de-identification technique.

Experiment with Mediapipe: Initially promising due to its open-source nature, it encountered significant constraints in Google Colab and high-powered systems. Its notable contribution lies in recognizing that facial detection with Mediapipe could serve as a rudimentary de-identification method, although not achieving the desired blended deepfake application.

Experiment with DeepfaceLab: Despite being closed-source, this model, the most recent among the three, featured both front and backend systems. The well-trained driving model yielded the best-blended deepfake application. However, large file sizes and high processing power requirements resulted in system crashes during lengthy deployments.

These findings confirm that while achieving the best-blended deepfake is desirable, it may not always be the fastest or most resource-efficient approach. Future work must balance considerations such as processing power, model openness, training data quality, and real-time application efficiency.

Ultimately, the objective of utilizing deepfake technology as a real-time de-identification method for privacy preservation and data protection was realized to varying extents by all models. The insights and lessons derived from each model are invaluable for guiding future developments and refinements in the domain of real-time deepfake technology.

5.5 Summary

In this chapter, we provided a comprehensive analysis of our experimental results obtained from the three deepfake models under investigation. Each model was critically examined, with a focus on assessing their performances in various aspects of deepfake application. Through our analysis, we highlighted the strengths and weaknesses inherent in each model, providing an in-depth understanding of their operational dynamics.

As we advance to the next chapter, we will present a summary of our thesis, discussing the overall findings and their implications. We will also elaborate on potential areas for future research and address the limitations we encountered during our research process.

CHAPTER 6

CONCLUSION

6.1 Overview

In this final chapter, we will reflect on the work carried out throughout this thesis, highlighting the main findings, and discussing their implications. We will also consider the limitations encountered during our research and discuss potential directions for future research.

6.2 Thesis Summary

The Research embarked on a multifaceted exploration of real-time deepfake technology with a primary focus on privacy preservation and data protection. Through the integration of fist-order motion, mediapipe and DeepfaceLab models, a real-time deepfake system was developed and implemented. By incorporating privacy considerations via masking techniques, potential risks associated with the misuse of deepfake technology were mitigated. The investigation also showed the intricate nuances of various model characteristics and their impact on deepfake performance. The findings revealed that while each model has its strengths, they all faced significant challenges. These include the need for extensive training, the demand for high computation power, and the difficulty in maintaining a balance between blending quality and real-time applicability.

This comprehensive analysis valuable insights into the optimization of deepfake algorithms, contributing to the ongoing discourse on the ethical and technical dimensions of the synthetic media. Furthermore, the thesis assessed the feasibility and practical implications of real-time deepfake technology on both single and multi faces within a single image shed light on its potential applications and limitations. By addressing these critical aspects, a deeper understanding of the opportunities and challenges inherent in the development of deepfake technology in real-world scenarios was explored. Overall, this research advances the responsible development and utilization of deepfake technology, emphasizing the importance of balance innovation with ethical considerations.

6.3 Limitations

As we navigated through our research, we encountered several obstacles that shaped the trajectory and results of our work. One of the most prominent challenges was the proprietary nature of two of the models used in our experiments. Because the inner workings of these models were not publicly accessible, we were limited in our capacity to refine, modify, or tailor these tools to suit our specific objectives. This factor underscored the importance of open-source tools in scientific research, which allow for customization and enhancements based on individual project needsAnother significant limitation was the computational demands of real-time deepfake applications. Given the intensive computational resources required to generate and manipulate high-quality deepfake images in real-time, we were consistently challenged by the need for powerful processing capabilities. Our experiments made it clear that the successful deployment of deepfake technology in real-time applications would necessitate a delicate balance of processing power, speed, and quality. As technology continues to evolve, there's hope for more efficient algorithms and models that can streamline this process without compromising the outcome's quality.

Furthermore, our findings emphasized the critical role that the quality and quantity of the training data play in achieving a convincingly blended model. For a deepfake to be indistinguishable from reality, the driving source must be thoroughly trained with a vast, high-quality dataset. We encountered challenges when the dataset was insufficient or of poor quality, which in turn affected the final output. This points to the need for robust, diverse, and high-quality datasets in training deepfake models.

Despite these limitations, our research offers valuable insights into the complexities of developing and deploying real-time deepfake technology. The lessons learned contribute significantly to our understanding of the challenges to be addressed in future work in this exciting and rapidly evolving field.

6.4 Future Works

Our research offers valuable insights into the current capabilities and limitations of real-time deepfake technology for privacy preservation and data protection. While much work remains to be done, the potential applications of this technology are promising, and we look forward to the advancements that future research will undoubtedly bring.

The future of deepfake technology in real-time applications shows significant promise, and the research contributes meaningfully to the knowledge in this domain. the findings underline the need for continued research and development, focusing on improving blending quality, ensuring real-time applicability, and balancing these with computational efficiency.

Further improvements could be achieved by extending model training, refining algorithms, improving resource efficiency, or incorporating additional data sources.

To improve blending quality and applicability in real-time, the exploration of more sophisticated models or improved algorithms may prove beneficial.

To address computational efficiency, future research could investigate ways to optimize the models for more efficient use of resources. This might involve designing lightweight models or developing methods for distributed computing.

REFERENCES

Akhter, I., & Black, M. J. (2015). *Pose-Conditioned Joint Angle Limits for 3D Human Pose Reconstruction* (pp. 1446–1455). http://mocap.cs.cmu.edu.

Alladi, T., Chamola, V., Sahu, N., Communications, M. G.-V., & 2020, undefined. (n.d.). Applications of blockchain in unmanned aerial vehicles: A review. *Elsevier*. https://doi.org/10.1016/j.vehcom.2020.100249

[Archived content]Opinions and recommendations - European Commission. (n.d.). Retrieved July 7, 2023, from https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/index_en.htm

Art. 3 GDPR – Territorial scope - General Data Protection Regulation (GDPR). (n.d.). Retrieved July 7, 2023, from https://gdpr-info.eu/art-3-gdpr/

Art. 99 GDPR – Entry into force and application - General Data Protection Regulation (GDPR). (n.d.). Retrieved July 7, 2023, from https://gdpr-info.eu/art-99-gdpr/

Article 8 - Protection of personal data | European Union Agency for Fundamental Rights. (n.d.). Retrieved April 14, 2022, from https://fra.europa.eu/en/eu-charter/article/8-protection-personal-data

ARTICLE29 - Guidelines on Data Protection Officers ('DPOs') (wp243rev.01). (n.d.). Retrieved July 8, 2023, from https://ec.europa.eu/newsroom/article29/items/612048/en

Artificial Intelligence in Society. (2019). *Artificial Intelligence in Society*. https://doi.org/10.1787/EEDFEE77-EN

Avidan, S., & Butman, M. (2006). Blind vision. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 3953 LNCS*, 1–13. https://doi.org/10.1007/11744078_1/COVER

BehavDT: A Behavioral Decision Tree Learning to Build User-Centric Context-Aware Predictive Model | Request PDF. (n.d.). Retrieved June 21, 2023, from https://www.researchgate.net/publication/338401551_BehavDT_A_Behavioral_Decision_Tree_Learning_to_Build_User-Centric_Context-Aware_Predictive_Model

Braeken, A., Liyanage, M., Kanhere, S. S., & Dixit, S. (2020). *Blockchain and cyberphysical systems*. https://oulurepo.oulu.fi/handle/10024/30452

Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization | UCLA Law Review. (n.d.). Retrieved June 25, 2023, from https://www.uclalawreview.org/broken-promises-of-privacy-responding-to-the-surprising-failure-of-anonymization-2/

Bzdok, D., Krzywinski, M., & Altman, N. (2018). Machine learning: supervised methods. *Nature Methods*, *15*(1), 5. https://doi.org/10.1038/NMETH.4551

Castelluccia, C., Le Métayer Inria, D., & Le Métayer, D. (n.d.). *Impact Analysis of Facial Recognition*.

Chamola, V., Hassija, V., Gupta, V., access, M. G.-I., & 2020, undefined. (n.d.). A comprehensive review of the COVID-19 pandemic and the role of IoT, drones, AI, blockchain, and 5G in managing its impact. *Ieeexplore.Ieee.Org*. Retrieved July 2, 2023, from https://ieeexplore.ieee.org/abstract/document/9086010/

Chamola, V., Kotesh, P., Agarwal, A., Naren, Gupta, N., & Guizani, M. (2021a). A Comprehensive Review of Unmanned Aerial Vehicle Attacks and Neutralization Techniques. *Ad Hoc Networks*, *111*, 102324. https://doi.org/10.1016/J.ADHOC.2020.102324

Chamola, V., Kotesh, P., Agarwal, A., Naren, Gupta, N., & Guizani, M. (2021b). A Comprehensive Review of Unmanned Aerial Vehicle Attacks and Neutralization Techniques. *Ad Hoc Networks*, *111*, 102324. https://doi.org/10.1016/J.ADHOC.2020.102324

Chang, Y.-P., Liu, C.-N., Pei, Z., Lee, S.-M., Lai, Y.-K., Han, P., Shih, H.-K., & Cheng, W.-H. (2019). New scheme of LiDAR-embedded smart laser headlight for autonomous vehicles. *Optics Express*, *27*(20), A1481–A1489.

Civil Aviation Authority. (n.d.). Retrieved July 2, 2023, from https://www.caa.co.uk/

CONSULTATIVE COMMITTEE OF THE CONVENTION FOR THE PROTECTION OF INDIVIDUALS WITH REGARD TO AUTOMATIC PROCESSING OF PERSONAL DATA CONVENTION 108 Guidelines on Facial Recognition Directorate General of Human Rights and Rule of Law 2 Contents. (n.d.).

Davenport, T., & Kalakota, R. (2019). The potential for artificial intelligence in healthcare. *Future Healthcare Journal*, *6*(2), 94. https://doi.org/10.7861/FUTUREHOSP.6-2-94

DGCA RPAS Guidance Manual, Online. Available:https:public... - Google Scholar. (n.d.). Retrieved July 2, 2023, from https://scholar.google.com/scholar?q=DGCA%20RPAS%20Guidance%20Manual,%20Online.%20Available:https:public-prd-dgca.s3.ap-south-1.amazonaws.comInventoryListheaderblockdronesDGCA20RPAS20Guidance20Manual.pdf.

Drone safety rules | Civil Aviation Safety Authority. (n.d.). Retrieved July 2, 2023, from https://www.casa.gov.au/drones/drone-rules/drone-safety-rules

Drones | Civil Aviation Authority. (n.d.). Retrieved July 2, 2023, from https://www.caa.co.uk/drones/

Ethics guidelines for trustworthy AI | Shaping Europe's digital future. (n.d.). Retrieved April 14, 2022, from https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai

EU-US: A new transatlantic agenda for global change. (n.d.). Retrieved April 14, 2022, from https://ec.europa.eu/commission/presscorner/detail/en/IP_20_2279

FAADroneZone Access - Home. (n.d.). Retrieved July 2, 2023, from https://faadronezone-access.faa.gov/#/

Fagnant, D. J., & Kockelman, K. (2015). Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice*, *77*, 167–181. https://doi.org/10.1016/j.tra.2015.04.003

Fundamental Rights Report 2020 | European Union Agency for Fundamental Rights. (n.d.). Retrieved April 14, 2022, from https://fra.europa.eu/en/publication/2020/fundamental-rights-report-2020

Gapeyenko, M., Petrov, V., Moltchanov, D., Andreev, S., Himayat, N., & Koucheryavy, Y. (n.d.). Flexible and reliable UAV-assisted backhaul operation in 5G mmWave cellular networks. *Ieeexplore.Ieee.Org*. Retrieved July 2, 2023, from https://ieeexplore.ieee.org/abstract/document/8482308/

Gellert, R., & Gutwirth, S. (2013). The legal construction of privacy and data protection. *Computer Law & Security Review*, *29*(5), 522–530. https://doi.org/10.1016/J.CLSR.2013.07.005

Gerstmair, M., Melzer, A., Onic, A., & Huemer, M. (2019). On the safe road toward autonomous driving: Phase noise monitoring in radar sensors for functional safety compliance. *IEEE Signal Processing Magazine*, *36*(5), 60–70.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. https://books.google.com/books?hl=en&lr=&id=omivDQAAQBAJ&oi=fnd&pg=PR5&ots=MNU6cvrzPS&sig=h_-QsuHglJHOfK3cTrsFrPToSnc

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (n.d.). *Generative Adversarial Nets*.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, *27*. http://www.github.com/goodfeli/adversarial

Goy, A., Nishtar, S., Dzau, V., Balatbat, C., & Diabo, R. (2019). *Health and healthcare in the fourth industrial revolution: Global Future Council on the future of health and healthcare 2016-2018*. https://apo.org.au/node/235506

Guarnera, L., Giudice, O., Nastasi, C., & Battiato, S. (2020). Preliminary Forensics Analysis of DeepFake Images. *12th AEIT International Annual Conference, AEIT 2020*. https://doi.org/10.23919/AEIT50178.2020.9241108

Han, J., Pei, J., & Tong, H. (2022). *Data mining: concepts and techniques*. https://books.google.com/books?hl=en&lr=&id=NR1oEAAAQBAJ&oi=fnd&pg=PP1&ots=_M9JPLsip2&sig=mgTSETNk6HJaOp13rbL9M2Fh49c

Hassija, V., Chamola, V., … D. K.-I. T. on, & 2020, undefined. (n.d.). A distributed framework for energy trading between UAVs and charging stations for critical applications. *Ieeexplore.Ieee.Org*. Retrieved July 2, 2023, from https://ieeexplore.ieee.org/abstract/document/9019832/

Hassija, V., Saxena, V., Communications, V. C.-C., & 2020, undefined. (2019). Scheduling drone charging for multi-drone network based on consensus time-stamp and game theory. *Elsevier*. https://doi.org/10.1016/j.comcom.2019.09.021

Human Rights Documents. (n.d.). Retrieved April 14, 2022, from https://ap.ohchr.org/documents/dpage_e.aspx?si=A/HRC/44/L.11

Ivanova, Y. (2020). The Role of the EU Fundamental Right to Data Protection in an Algorithmic and Big Data World. *SSRN Electronic Journal*. https://doi.org/10.2139/SSRN.3697089

Karras, T., Laine, S., & Aila, T. (2019). *A Style-Based Generator Architecture for Generative Adversarial Networks* (pp. 4401–4410).

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and improving the image quality of stylegan. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 8107–8116.

Kim, Y. Bin, Kim, J. G., Kim, W., Im, J. H., Kim, T. H., Kang, S. J., & Kim, C. H. (2016). Predicting fluctuations in cryptocurrency transactions based on user comments and replies. *PLoS ONE*, *11*(8). https://doi.org/10.1371/JOURNAL.PONE.0161197

Kubáňova, J., & Kubasákova, I. (2018). The introduction and impact of tachographs on road freight transport. *Transport Means-Proceedings of the International Conference*, 1085–1090.

Lample, G., Zeghidour, N., Usunier, N., Bordes, A., Denoyer, L., & Ranzato, M. (2017). Fader Networks: Manipulating Images by Sliding Attributes. *Advances in Neural Information Processing Systems*, *2017-December*, 5968–5977.

Langner, O., Dotsch, R., Bijlstra, G., Wigboldus, D. H. J., Hawk, S. T., & van Knippenberg, A. (2010). Presentation and validation of the radboud faces database. *Cognition and Emotion*, *24*(8), 1377–1388. https://doi.org/10.1080/02699930903485076

Li, B., Fei, Z., Zhang, Y., Li, B., Fei, Z., & Zhang, Y. (2019). UAV communications for 5G and beyond: Recent advances and future trends. *Ieeexplore.Ieee.Org*. https://ieeexplore.ieee.org/abstract/document/8579209/

Li, K., Voicu, R., Kanhere, S., … W. N.-I. T. on, & 2019, undefined. (n.d.). Energy efficient legitimate wireless surveillance of UAV communications. *Ieeexplore.Ieee.Org*. Retrieved July 2, 2023, from https://ieeexplore.ieee.org/abstract/document/8601408/

Li, M., Zuo, W., & Zhang, D. (2016). *Deep Identity-aware Transfer of Facial Attributes*.

Li, Y., Dai, W., Ming, Z., & Qiu, M. (2016). Privacy Protection for Preventing Data Over-Collection in Smart City. *IEEE Transactions on Computers*, *65*(5), 1339–1350. https://doi.org/10.1109/TC.2015.2470247

Liu, Y., Dai, H., Wang, Q., Shukla, M., communications, M. I.-C., & 2020, undefined. (n.d.). Unmanned aerial vehicle for internet of everything: Opportunities and challenges. *Elsevier*. Retrieved June 25, 2023, from https://www.sciencedirect.com/science/article/pii/S0140366419318754

Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). *Deep Learning Face Attributes in the Wild* (pp. 3730–3738).

Luca Montag, B., Mcleod, R., De Mets, L., Gauld, M., Rodger, F., Pełka, M., & -european Digital Rights, Edr. (n.d.). *Biometric Technology Providers*.

Lynskey, O. (2015). *The foundations of EU data protection law*. https://books.google.com/books?hl=en&lr=&id=jCXYCgAAQBAJ&oi=fnd&pg=PP1&ots=iXk-a5HDyg&sig=JVSuSnH5TT_Fg1KExtuyX8pRlIU

Madiega, T., & Mildebrath, H. (n.d.). *Regulating facial recognition in the EU*. https://doi.org/10.2861/140928

Mandery, C., Terlemez, Ö., Do, M., Vahrenkamp, N., & Asfour, T. (2015). The KIT whole-body human motion database. *Proceedings of the 17th International Conference on Advanced Robotics, ICAR 2015*, 329–336. https://doi.org/10.1109/ICAR.2015.7251476

McPherson, R., Shokri, R., & Shmatikov, V. (2016). *Defeating Image Obfuscation with Deep Learning*. https://arxiv.org/abs/1609.00408v2

Meden, B., Gonzalez-Hernandez, M., Peer, P., & Štruc, V. (2023). Face deidentification with controllable privacy protection. *Image and Vision Computing*, *134*. https://doi.org/10.1016/j.imavis.2023.104678

Mitchell, T. M. (1997). Does Machine Learning Really Work? *AI Magazine*, *18*(3), 11–11. https://doi.org/10.1609/AIMAG.V18I3.1303

Mohammed, M., Khan, M., & Bashier, E. (2016). *Machine learning: algorithms and applications*. https://books.google.co.uk/books?hl=en&lr=&id=X8LBDAAAQBAJ&oi=fnd&pg=PP1&dq=Machine+learning:+algorithms+and+applications.+CRC+Press%3B+2016.&ots=qQGxAvG8FD&sig=NE1r8YQgTX4NxzGh3EP8lajFBRk

Mondschein, C. F., & Monda, C. (2018). The eu's general data protection regulation (GDPR) in a research context. *Fundamentals of Clinical Data Science*, 55–71. https://doi.org/10.1007/978-3-319-99713-1_5/FIGURES/2

Montague, P. R. (1999). Reinforcement Learning: An Introduction, by Sutton, R.S. and Barto, A.G. *Trends in Cognitive Sciences*, *3*(9), 360. https://doi.org/10.1016/S1364-6613(99)01331-5

Mostert, M., Bredenoord, A., … B. V. D. S. J. of, & 2018, undefined. (n.d.). From privacy to data protection in the EU: Implications for big data health research. *Brill.Com*. Retrieved July 7, 2023, from https://brill.com/view/journals/ejhl/25/1/article-p43_43.xml

Nilsback, M. E., & Zisserman, A. (2008). Automated flower classification over a large number of classes. *Proceedings - 6th Indian Conference on Computer Vision, Graphics and Image Processing, ICVGIP 2008*, 722–729. https://doi.org/10.1109/ICVGIP.2008.47

Ondruš, J., Kolla, E., Vertaľ, P., & Šarić, Ž. (2020). How Do Autonomous Cars Work? *Transportation Research Procedia*, *44*, 226–233.

Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition. *BMVC 2015 - Proceedings of the British Machine Vision Conference 2015*.

Pedregosa FABIANPEDREGOSA, F., Michel, V., Grisel OLIVIERGRISEL, O., Blondel, M., Prettenhofer, P., Weiss, R., Vanderplas, J., Cournapeau, D., Pedregosa, F., Varoquaux, G., Gramfort, A., Thirion, B., Grisel, O., Dubourg, V., Passos, A., Brucher, M., Perrot andÉdouardand, M., Duchesnay, andÉdouard, & Duchesnay EDOUARDDUCHESNAY, Fré. (2011). Scikit-learn: Machine learning in Python. *Jmlr.Org*, *12*, 2825–2830. https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf?ref=https:/

Perarnau, G., van de Weijer, J., Raducanu, B., & Álvarez, J. M. (2016). *Invertible Conditional GANs for image editing*.

Pérez, P., Gangnet, M., & Blake, A. (2003). Poisson image editing. *ACM SIGGRAPH 2003 Papers, SIGGRAPH '03*, 313–318. https://doi.org/10.1145/1201775.882269

Polydoros, A., Systems, L. N.-J. of I. & R., & 2017, undefined. (2017). Survey of model-based reinforcement learning: Applications on robotics. *Springer*, *86*(2), 153–173. https://doi.org/10.1007/s10846-017-0468-y

Raab, C. D. (2020). Information privacy, impact assessment, and the place of ethics ∗. *Computer Law and Security Review*, *37*. https://doi.org/10.1016/j.clsr.2020.105404

Ray, S. (2019). A Quick Review of Machine Learning Algorithms. *Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Prespectives and Prospects, COMITCon 2019*, 35–39. https://doi.org/10.1109/COMITCON.2019.8862451

REGULATING BIOMETRICS Global Approaches and Urgent Questions. (2020).

Result details. (n.d.). Retrieved April 14, 2022, from https://search.coe.int/cm/Pages/result_details.aspx?ObjectID=0900001680a28ddf

Ribaric, S., Ariyaeeinia, A., & Pavesic, N. (2016). De-identification for privacy protection in multimedia content: A survey. *Signal Processing: Image Communication*, *47*, 131–151. https://doi.org/10.1016/J.IMAGE.2016.05.020

Ribaric, S., & Pavesic, N. (2015). An Overview of Face De-identification in Still Images and Videos. *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition, FG 2015*, *2015-January*. https://doi.org/10.1109/FG.2015.7285017

Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Niessner, M. (2019). FaceForensics++: Learning to Detect Manipulated Facial Images. *Proceedings of the IEEE International Conference on Computer Vision*, *2019-October*, 1–11. https://doi.org/10.1109/ICCV.2019.00009

Saravanan, R., & Sujatha, P. (2019). A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification. *Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, ICICCS 2018*, 945–949. https://doi.org/10.1109/ICCONS.2018.8663155

Sarker, I. H. (2021a). Deep Cybersecurity: A Comprehensive Overview from Neural Network and Deep Learning Perspective. *SN Computer Science*, *2*(3). https://doi.org/10.1007/S42979-021-00535-6

Sarker, I. H. (2021b). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, *2*(3), 1–21. https://doi.org/10.1007/S42979-021-00592-X/FIGURES/11

Sarker, I. H., Abushark, Y. B., Alsolami, F., & Khan, A. I. (2020). IntruDTree: A Machine Learning Based Cyber Security Intrusion Detection Model. *Symmetry 2020, Vol. 12, Page 754*, *12*(5), 754. https://doi.org/10.3390/SYM12050754

Sarker, I. H., Furhad, M. H., & Nowrozy, R. (2021). AI-Driven Cybersecurity: An Overview, Security Intelligence Modeling and Research Directions. *SN Computer Science*, *2*(3). https://doi.org/10.1007/S42979-021-00557-0

Sarker, I. H., & Kayes, A. S. M. (2020). ABC-RuleMiner: User behavioral rule-based machine learning method for context-aware intelligent services. *Journal of Network and Computer Applications*, *168*, 102762. https://doi.org/10.1016/J.JNCA.2020.102762

Sathya, R., in, A. A.-I. J. of A. R., & 2013, undefined. (n.d.). Comparison of supervised and unsupervised learning algorithms for pattern classification. *Citeseer*.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A Unified Embedding for Face Recognition and Clustering. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12-June-2015*, 815–823. https://doi.org/10.1109/CVPR.2015.7298682

Schwartz, P. M., & Peifer, K.-N. (2017). *Transatlantic Data Privacy*. https://papers.ssrn.com/abstract=3066971

Singh, A. (2016). *A review of supervised machine learning algorithms; A review of supervised machine learning algorithms*.

Singh, M., Aujla, G., 2020-IEEE, R. B.-I. I., & 2020, undefined. (n.d.). ODOB: One drone one block-based lightweight blockchain architecture for internet of drones. *Ieeexplore.Ieee.Org*. Retrieved July 2, 2023, from https://ieeexplore.ieee.org/abstract/document/9162950/

Skrúcaný, T., Šarkan, B., Figlus, T., Synák, F., & Vrábel, J. (2017). Measuring of noise emitted by moving vehicles. *MATEC Web of Conferences*, *107*, 72.

Srivastava, S., Gupta, S., Dikshit, O., & Nair, S. (2020). A review of UAV regulations and policies in India. *Lecture Notes in Civil Engineering*, *51*, 315–325. https://doi.org/10.1007/978-3-030-37393-1_27

Statement by the Special Rapporteur on the right to adequate housing, Leilani Farha, during the Interactive Dialogue at the Human Rights Council | OHCHR. (n.d.). Retrieved April 14, 2022, from https://www.ohchr.org/en/statements/2017/03/statement-special-rapporteur-right-adequate-housing-leilani-farha-during?LangID=E&NewsID=21264

Study supporting the impact assessment of the AI regulation | Shaping Europe's digital future. (n.d.). Retrieved April 11, 2022, from https://digital-strategy.ec.europa.eu/en/library/study-supporting-impact-assessment-ai-regulation

The EU rights to privacy and personal data protection: 20 years in 10 questions — Vrije Universiteit Brussel. (n.d.). Retrieved April 14, 2022, from

https://researchportal.vub.be/en/publications/the-eu-rights-to-privacy-and-personal-data-protection-20-years-in

The Norwegian Civil Aviation Authority - regjeringen.no. (n.d.). Retrieved July 2, 2023, from https://www-regjeringen-no.translate.goog/no/dep/sd/org/underliggende-etater/statens-luftfartstilsyn/id443418/?_x_tr_sl=no&_x_tr_tl=en&_x_tr_hl=en&_x_tr_pto=sc

The OECD Artificial Intelligence Policy Observatory - OECD.AI. (n.d.). Retrieved April 14, 2022, from https://oecd.ai/en/

Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., & Nießner, M. (2020). Face2Face: Real-time Face Capture and Reenactment of RGB Videos. *Communications of the ACM*, *62*(1), 96–104. https://doi.org/10.1145/3292039

Things, I. S.-I. of, & 2019, undefined. (2019). A machine learning based robust prediction model for real-life mobile phone data. *Elsevier*. https://www.sciencedirect.com/science/article/pii/S254266051830180X

This is how we lost control of our faces | MIT Technology Review. (n.d.). Retrieved April 11, 2022, from https://www.technologyreview.com/2021/02/05/1017388/ai-deep-learning-facial-recognition-data-history/

Tolosana, R., Vera-Rodriguez, R., Fierrez, J., Morales, A., & Ortega-Garcia, J. (2020). Deepfakes and beyond: A Survey of face manipulation and fake detection. *Information Fusion*, *64*, 131–148. https://doi.org/10.1016/J.INFFUS.2020.06.014

Use of facial recognition technology for migrant disembarkation in Italy. (n.d.). Retrieved April 14, 2022, from https://www.europarl.europa.eu/doceo/document/E-9-2021-002182_EN.html#def2

Vashisht, S., Jain, S., Communications, G. A.-C., & 2020, undefined. (n.d.). MAC protocols for unmanned aerial vehicle ecosystems: Review and challenges. *Elsevier*. Retrieved July 2, 2023, from https://www.sciencedirect.com/science/article/pii/S0140366420310781

Verdoliva, L. (2020). Media Forensics and DeepFakes: An Overview. *IEEE Journal on Selected Topics in Signal Processing*, *14*(5), 910–932. https://doi.org/10.1109/JSTSP.2020.3002101

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *1*. https://doi.org/10.1109/CVPR.2001.990517

Wehde, M. (2019). Healthcare 4.0. *IEEE Engineering Management Review*, *47*(3), 24–28. https://doi.org/10.1109/EMR.2019.2930702

Winter, G. (2019). Machine learning in healthcare. *Https://Doi.Org/10.12968/Bjhc.2019.25.2.100*, *25*(2), 100–101. https://doi.org/10.12968/BJHC.2019.25.2.100

Wojewidka, J. (2020). The deepfake threat to face biometrics. *Biometric Technology Today*, *2020*(2), 5–7. https://doi.org/10.1016/S0969-4765(20)30023-0

Xiong, X., & De La Torre, F. (2013). Supervised descent method and its applications to face alignment. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 532–539. https://doi.org/10.1109/CVPR.2013.75

Yun, H.-S., Kim, T.-H., & Park, T.-H. (2019). Speed-Bump Detection for Autonomous Vehicles by Lidar and Camera. *Journal of Electrical Engineering & Technology*, *14*(5), 2155–2162.

Zalnieriute, M. (2018). Developing a European standard for international data transfers after Snowden: Opinion 1/15 on the EU-Canada PNR agreement. *Modern Law Review*, *81*(6), 1046–1063. https://doi.org/10.1111/1468-2230.12378

Zhu, B., Fang, H., Sui, Y., & Li, L. (2020). Deepfakes for medical video de-identification: Privacy protection and diagnostic information preservation. *AIES 2020 - Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 414–420. https://doi.org/10.1145/3375627.3375849

Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). *Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks* (pp. 2223–2232).

Dolhansky, B., Howes, R., Pflaum, B., Baram, N., & Ferrer, C. C. (2019). The deepfake detection challenge (dfdc) preview dataset. *arXiv preprint arXiv:1910.08854*.

Jafri, R., & Arabnia, H. R. (2009). A survey of face recognition techniques. *Journal of information processing systems*, *5*(2), 41-68.

Khoo, B., Phan, R. C. W., & Lim, C. H. (2022). Deepfake attribution: On the source identification of artificially generated images. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *12*(3), e1438.

Kirchengast, T. (2020). Deepfakes and image manipulation: criminalisation and control. *Information & Communications Technology Law*, *29*(3), 308-323.

Kwok, A. O., & Koh, S. G. (2021). Deepfake: a social construction of technology perspective. *Current Issues in Tourism*, *24*(13), 1798-1802.

Mahmud, B. U., & Sharmin, A. (2021). Deep insights of deepfake technology: A review. *arXiv preprint arXiv:2105.00192*.

Masood, M., Nawaz, M., Malik, K. M., Javed, A., Irtaza, A., & Malik, H. (2023). Deepfakes generation and detection: State-of-the-art, open challenges, countermeasures, and way forward. *Applied intelligence*, *53*(4), 3974-4026.

Westerlund, M. (2019). The emergence of deepfake technology: A review. *Technology innovation management review*, *9*(11).