SAPIENZA
UNIVERSITÀ DI ROMA

Protecting Users from Face Recognition Systems using Adversarial Machine Learning

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica

Corso di Laurea Magistrale in Laurea Magistrale in Computer Science

Candidate

Zeeshan Hussain Khand

ID number 1880109

Thesis Advisor

Prof. Iacopo Masi

Academic Year 2021/2022

Thesis defended on 29 March 2023
in front of a Board of Examiners composed by:

Prof. Emanuelle Pannizzi (chairman)

Prof. Fabio Galasso

Prof. Claudio Di Ciccio

Prof. Daniele Pannone

Prof. Ivano Salvo

Prof. Zuliani Alberto

Protecting Users from Face Recognition Systems using Adversarial Machine Learning

Master's thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: khand.1880109@studenti.uniroma1.it

Abstract

Face recognition systems have become increasingly important in various fields, including security, law enforcement, and social media. Recently, much has been written on the privacy and ethical concerns raised with unauthorized face recognition. However, such systems are vulnerable to adversarial attacks.

In this thesis, we conducted a thorough literature review of advances in face recognition technology and adversarial attacks that have been proposed to prevent unauthorized face recognition.

We also re-design an attack proposed by Lowkey [9] based on their research paper. Lowkey [9] is state of art in adversarial machine learning for protecting the identity of a user from face recognition. Furthermore, we evaluate the effectiveness of the Lowkey attack against various blur transformations from Kornia during inference. Specifically, we evaluate the robustness of the lowkey attack against the blur transformations such as Box-blur, Pool-blur, Max-blur, Median-blur, Gaussian-blur, and Motion-blur. Our evaluation showed that the Lowkey attack was not effective in protecting against recognition when Median-blur and Box-blur were applied on protected images with larger kernel sizes.

To address this limitation, we developed a new attack that utilized Gaussian-blur and Median-blur with high kernel sizes as part of our attack pipeline to create a more robust attack against these transformations. Our attack gives the same protection as Lowkey and also ensures the transferability of the attack even when blur transformations are applied by Face Recognition systems.

Our attack pre-processes user images to ensure that they cannot be used by a third party for facial recognition purposes, even if the images are publicly available on social media platforms. Our proposed attack improves over the lowkey attack pipeline and takes into account denoising transformations such as median-blur and gaussian-blur, to generate attacks and further protect user images.

To generate attacks and extract features, we used an ensemble of models of IR-152 and ResNet-152 backbones with Cosface and Arcface heads [50]. We evaluated our attack using top-1 and top-5 accuracy. We design the attack to maintain perceptual similarity while ensuring that the probed image of a user is far from their gallery images in the feature space.

Finally, we conclude that more research is needed on the susceptibility of adversarial attacks proposed for face recognition against blur transformations. Our research and proposed attack are a step forward in this direction. A future direction could be to generate attacks that run faster and ensure protection against all geometric transformations.

Acknowledgements

I would like to dedicate this thesis to my paternal aunt who was like a second mother to me. She passed away due to cancer in 2022 but she would be really happy in the heaven to see me graduate and achieve my dreams.

I also dedicate my thesis to my parents who have given me unconditional and unwavering support during the whole Master's degree and especially during the thesis. My father for always encouraging me and being a role model in my life.

I would also like to thank my younger brother who always encouraged me and my friends for their unconditional support during my hard times.

Finally, I dedicate this thesis to my supervisor Prof. Dr. Iacopo Masi for supporting me throughout the thesis and being a great mentor. Honestly, I could not have asked for a better topic and better supervisor to do a thesis with for this degree.

I would also like to thank the Sapienza University of Rome for all the great memories, lessons, and above all support for my studies during the hard and testing times of Covid 19.

Contents

List of Figures

Chapter 1

Introduction

Machine Learning systems since their introduction into our worlds have disrupted a lot of industries and opened many doors of opportunities for humans in the field of science, medicine, and technology.

Machine Learning is a branch of Artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop a conventional algorithm for effectively performing the task. Some popular examples of machine learning include natural language processing, image and speech recognition, and bio-informatics.

Tom Mitechell(1998) [33] defines Machine Learning as Well-posed Learning Problem. A computer program is set to learn from Experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Deep learning is a sub-field of machine learning that involves the use of artificial neural networks with multiple layers to learn complex representations of data. The term "deep" refers to the fact that these neural networks typically have many hidden layers, allowing them to model highly nonlinear relationships between inputs and outputs.

The success of deep learning can be attributed to its ability to learn high-level representations of data automatically, without the need for manual feature engineering. Traditional machine learning techniques often require experts to handcraft features that are relevant to the task at hand, which can be a laborious and time-consuming process. In contrast, deep learning models learn features from raw data, enabling them to automatically learn complex patterns and relationships that might be difficult or impossible to detect using traditional techniques.

One of the key applications of deep learning in computer vision is Face Recognition systems. Deep learning has led to significant improvements in the accuracy and robustness of face recognition models. Deep learning-based face recognition models typically use convolutional neural networks (CNNs) to extract features from face images and then use a classification model to match these features to identities.

However, the advancements in Face recognition have also led to several privacy and ethical concerns. The rise of powerful face recognition models has led to concerns regarding privacy of an individual identity in the digital world and ethical concerns regarding control over their digital images. FR systems are being widely deployed by Government agencies around the world [9], whilst private social media companies are hoarding a large amount of facial data to create personalized experience for users and collect more information about them in order to sell relevant advertisements.

Private companies such as Clearview.AI raise the alarm bells over this concern and caused significant outrage. Many private companies are scrapping social media and other websites that link an individual to an identity and have built a large database with labeled face images for face verification and identification [18].

On the other hand, anyone with a good knowledge of programming and access to a good dataset for training can design a face recognition system. The potential for misuse of facial recognition technology is vast and could lead to disastrous consequences. With the widespread use of street cameras, video doorbells, security cameras, and personal cellphones, individuals can be identified at any given moment. This can give stalkers access to their victims' social media profiles with just a single snapshot [41]. Stores can track customers' precise in-store shopping behavior and connect it to their online ads and browsing profiles. Additionally, identity thieves can use facial recognition technology to identify and potentially gain access to personal accounts [40].

Our aim around this thesis is to design an adversarial attack that protects a user's images from recognition by unauthorized Face Recognition systems. We aim to design images protected by adversarial attacks that are robust to various transformations during inference. We aim to corrupt the databases of FR systems through adversarial gallery images protected by our attack so that when a probe image is provided, the FR systems cannot identify and thus cannot associate an identity to the image. We also provide a comparative review with Lowkey which is the current state of the art data-poisoning alternative, and perform a comprehensive study of this tool, its results, and vulnerabilities.

We hope through this thesis, we are able to take control back against unauthorized face recognition and protect our identity from being manipulated as we navigate the recent advances in deep learning and Artificial intelligence in digital space.

1.1 Introduction to Deep Learning

Deep learning is a sub-field of machine learning that utilizes artificial neural networks, which are modeled after the structure and function of the human brain, to analyze and interpret complex data. The key distinction of deep learning from traditional machine learning is its ability to learn from large amounts of unstructured data, such as images, audio, and text.

A deep learning model is composed of layers of interconnected "neurons" which process and transmit information. The layers at the front of the network called the input layer, receive the raw data, which is then processed and transformed by the layers in the middle of the network, called the hidden layers. The final layer, called the output layer, produces the model's prediction or decision.

Deep learning models are able to learn and improve over time, through the process of training and fine-tuning, and it can be considered as a key technology for achieving artificial intelligence.

Deep neural networks (DNN's) and Convolutional Neural Networks(CNN's) that are type of deep learning models, have been used to achieve state-of-the-art performance in a wide range of applications such as image recognition,speech recognition,face recognition, natural language processing, robotics,healthcare and self-driving cars.

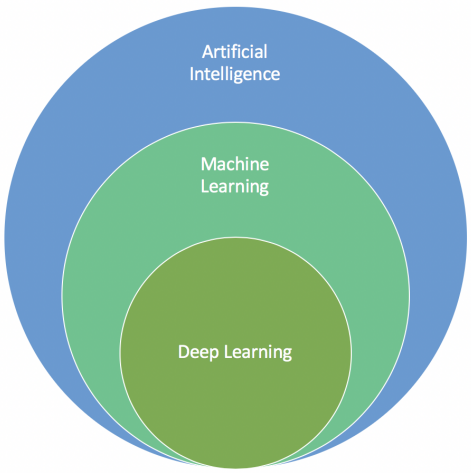


Figure 1.1. Deep Learning diagram

1.2 Face Recognition Systems

Face recognition systems are computer-based systems that are designed to automatically identify or verify a person from a digital image or a video frame from a video source. These systems use advanced algorithms and machine learning techniques to compare facial features from an image or video of a person to a database of known

faces to identify the person or to confirm their identity. Deep face recognition has become increasingly popular in recent years due to its ability to achieve high levels of accuracy, even in challenging scenarios such as low-quality images, large variations in facial expression, and changes in pose and lighting.

Deep face recognition typically involves training deep neural network models on large datasets of facial images. The models learn to extract high-level features from the images, such as the location of the eyes, nose, and mouth, and to use these features to identify individuals. The features extracted by deep face recognition models are typically much more abstract and robust than the features used by traditional face recognition techniques, which often rely on handcrafted features such as eigenfaces or local binary patterns.

The technology collects a set of unique biometric data of each person associated with their face and facial expression to identify, verify and/or authenticate a person.

There are two main modalities of face recognition systems:

1. **Facial Verification:** Face verification is 1:1 matching face recognition. It involves comparing two images of faces to determine if they belong to the same person.
2. **Facial Identification:** The goal of face identification is to answer the question "Who is this person?" by matching the given face image to a set of possible identities in a database. Face identification entails 1:N face recognition where you compare a given face image against images in a database to find potential matches.

Face recognition systems can be used in a variety of applications, including security and surveillance, personal identification, and mobile devices for unlocking the device or for payment authorization. Recently, advancements in deep learning and computer vision have significantly improved the performance of face recognition systems. These systems are now able to recognize faces even in poor lighting or when the person is wearing glasses or has a different facial expression[57].

Several state-of-the-art face recognition models such as VGGFace2, Residual Networks (ResNet) and ArcFace have been extensively studied [4] [19] [12]. The deeply learnt models have focused on improving the bio-metric performance in the presence of severe bio-metric sample quality degradation (i.e. face image) such as pose, illumination, expression, ageing, and heterogeneity [48]. With improved performance, the deep models can be used for identification where a subject is probed within the learned models in a closed enrolment setting or for verification where the model is used to extract the features from two images and thereupon compare them to make a decision based on a pre-computed threshold [55].

1.3 Adversarial Machine Learning

Despite their large-scale usage and high accuracy, [44] first demonstrated that neural networks are vulnerable to adversarial attacks. Adversarial Machine Learning is a sub-field of machine learning that focuses on the study of methods for training models that are robust to adversarial examples[16]. Adversarial examples are inputs to a machine learning model that have been specifically crafted to cause the model to make a mistake. These inputs are often very similar to normal examples, but they have been modified in a way that is intended to fool the model.

There are two main trends in research related to Adversarial machine learning:

1. **Defending against adversarial examples:** This approach involves modifying the machine learning model or the training process to make it more robust to adversarial examples.
2. **Crafting adversarial examples**: This approach involves generating adversarial examples that can fool a machine learning model. This is typically done by finding the smallest perturbation that can be made to an input to cause the model to make a mistake.

Adversarial Machine Learning is important because it highlights the potential vulnerabilities of machine learning models, which can be exploited by malicious actors to cause harm. The nature of this weakness of Deep Neural Networks(DNN) has raised many security concerns regarding the general deployability of neural networks in sensitive domains such as finance, healthcare, and self-driving cars.

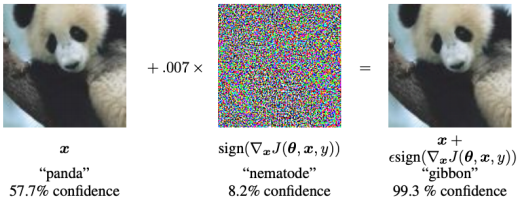


Figure 1.2. Adversarial Attack Example taken from [15]

Chapter 2

Related Work

An adversarial attack on a face recognition system is a type of attack in which an attacker generates specific input samples, called "adversarial examples," to fool or mislead the face recognition model into making a wrong prediction or decision.

In the context of face recognition systems, an attacker could create an adversarial example by slightly altering an image of a person's face in a way that is not noticeable to the human eye but causes the face recognition system to misidentify the person or to identify them as someone else.

There are different methods to create adversarial examples, such as adding noise to the image, changing the color or brightness of the image, or using techniques like "transfer learning" to create adversarial examples that are effective across different models or even across different tasks [56]. With the rise in Deepfake technologies [52] that use a myriad of face data to train, it has become more important than ever to protect our facial images and identity in cyberspace.

This chapter will discuss the advances in Deep Face Recognition and more importantly in learning the right face features for face verification and identification. Then we shall discuss the ethical and security implications of Face Recognition systems and how Adversarial machine learning can be used to protect against invading Face Recognition systems.

The final part would be dedicated to related work in this regard in the context of recent advances in adversarial attacks aimed at face recognition systems.

2.1 Deep Face Recognition

Face recognition is a sub-field of computer vision that involves the identification or verification of individuals based on their facial features. This computer vision task involves the capture, analyze, and comparison patterns based on the person's facial details.

Due to its utility, Face Recognition systems have been widely used as a prominent biometric technique for identity authentication in many areas such as the Military, finance, public security, and daily life. Face Recognition garners great interest due to its applications in entertainment, personal shopping, video surveillance, and photo tagging as well as face verification in mobiles.

However, advances in face recognition involve the models to overcome the PIE(Posture, Illumination, and Expression) [31]. In order to obtain the ideal recognition results, a Face Recognition system needs to fully co-relate the identity signals without being distracted by pose variability, illumination changes, and several dynamic facial expressions. Recently, a big leap forward in resolving the PIE problems in face recognition came with the introduction of deep neural networks and massive labeled datasets for Deep Convolutional Neural Networks(DCNNs) for training [27].

The advancements in face recognition are in tandem with improvements in face processing and detection, availability of massive datasets for training along-with improvements in network architectures and loss functions along with proper evaluation metrics for inference. In 2014, DeepFace [45] and DeepID [39] obtained the state of art verification accuracy in the LFW dataset, surpassing human-level performance and got an accuracy of 99%.

The process of Face Recognition is tied to two phases: Training a deep convolutional neural network model for face recognition and then using the trained DCNN as a feature extractor for face matching. A figure that depicts this pipeline and phases is Figure 2.1 provided by [32] in their paper.

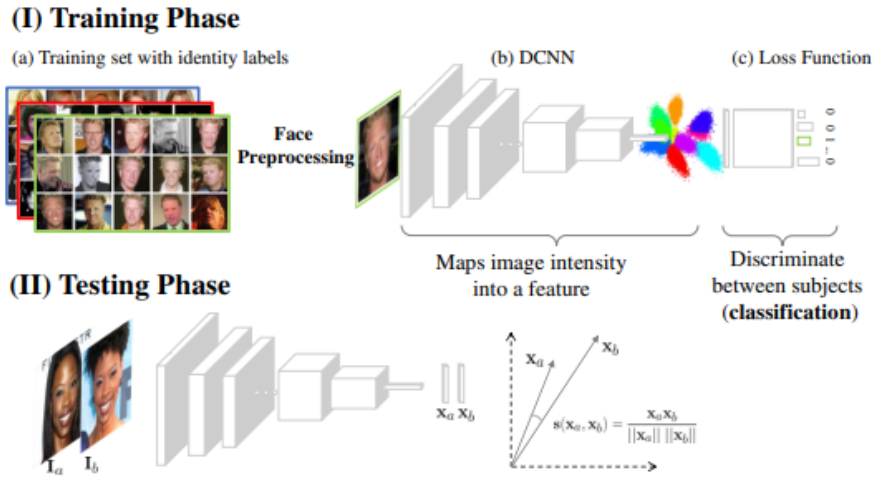


Figure 2.1. Face Recognition phases and pipeline

The advances in Face Recognition technology can be described in terms of a combination of advances in face-detection, data collection, face alignment, and processing, advances in network architectures, and evaluation metrics that we will cover in the

following pages.

2.1.1 Data Processing

A key advancement in the performance of deep neural networks on face recognition can be attributed to the availability of large public datasets in the computer vision community that are used in training routines. The availability of large public datasets makes it a possibility to learn face representations at scale for deep neural networks. Currently, the main datasets to the computer vision community are:

Labeled Faces in the Wild

Labeled Faces in the Wild (LFW) dataset is a widely used benchmark dataset for face recognition research. It was created by researchers at the University of Massachusetts, Amherst, and contains over 13,000 images of faces from the internet, with each image labeled with the name of the person depicted.

The LFW dataset [22] is intended to reflect the challenges of real-world face recognition scenarios, with a wide range of variations in pose, illumination, expression, and occlusion. It includes images of both celebrities and ordinary people, and the faces in the dataset span a wide range of ages, ethnicities, and genders.

The dataset is split into two parts: a development set, which contains 4,000 images, and a test set, which contains 9,000 images. The test set is further divided into a "restricted" subset of 6,000 images and an "unrestricted" subset of 3,000 images, with the latter subset containing more challenging variations in pose and lighting.

The LFW dataset has been widely used as a benchmark for evaluating the performance of face recognition algorithms, and many state-of-the-art deep learning models have been trained and tested on this dataset. The dataset has also been used to study a wide range of topics related to face recognition, such as the impact of different factors on recognition accuracy, the effect of training data size, and the limitations of existing algorithms.

FaceScrub Dataset

FaceScrub is developed by researchers at NUS Singapore. The FaceScrub dataset [35] is a dataset of facial images that was created as a benchmark for face recognition research. It contains over 100,000 images of more than 530 different people, including celebrities, politicians, and other public figures.

The images in the dataset are labeled with the name of the person depicted and also include metadata such as gender, ethnicity, and occupation. The dataset is designed to be representative of the diversity of faces encountered in real-world scenarios, with a wide range of variations in pose, expression, and lighting.

One of the unique features of the FaceScrub dataset is that it includes multiple images of each person, taken from different angles and under different lighting conditions. This makes the dataset well-suited for evaluating the performance of face recognition algorithms under varying conditions and for studying the impact of different factors on recognition accuracy.

The FaceScrub dataset has been used in a wide range of research studies, including the development and evaluation of deep learning models for face recognition, the study of facial expression recognition, and the analysis of facial features for demographic and personality prediction.

This is the dataset we use in our thesis for controlled experiments, comparative analysis with Lowkey, and our results.

Casia WebFace

The CASIA WebFace dataset is a large-scale dataset of facial images created by the Chinese Academy of Sciences' Institute of Automation [57]. The dataset was designed to be used as a benchmark for face recognition research, and it has been widely used in the development and evaluation of deep learning models for face recognition.

The CASIA WebFace dataset contains over 500,000 facial images of over 10,000 subjects. The images are of varying quality and were collected from the internet using a combination of automatic and manual methods. The images include a wide range of facial poses, expressions, and lighting conditions, and the dataset is designed to be representative of the diversity of faces encountered in real-world scenarios.

The images in the dataset are aligned and cropped to remove non-facial regions and are labeled with the identity of the person depicted. The dataset is divided into a training set and a testing set, with the training set containing 494,414 images of 10,575 subjects and the testing set containing 10,000 images of 500 subjects.

The CASIA WebFace dataset has been used in a wide range of research studies, including the development and evaluation of deep learning models for face recognition, the study of facial expression recognition, and the analysis of facial features for demographic and personality prediction.

MS-Celeb 1M Dataset

The MS-Celeb-1M dataset, also known as MS-1M, is a large-scale face recognition dataset that was released by Microsoft in 2016. It contains over 10 million images of approximately 1 million celebrities and public figures and is one of the largest publicly available face datasets [17].

The images in MS-1M were collected from various sources, including search engines and social media platforms, and are labeled with the corresponding celebrity's

name. The dataset is intended for use in face recognition research, including the development of deep learning models for face recognition.

This is a popular benchmark dataset used to train dense deep residual networks that are used in Face Recognition [50].

VGGFace2 dataset

The VGGFace2 dataset is a large-scale face recognition dataset that was released by the Visual Geometry Group (VGG) at the University of Oxford in 2018. It is designed for training and evaluating deep learning models for face recognition.

The VGGFace2 dataset contains over 3.3 million images of more than 9,000 identities, making it one of the largest publicly available face datasets. The images were collected from various sources, including the internet and video frames, and are labeled with the corresponding identity's name. The dataset includes a wide range of variations in pose, lighting, and expression, which makes it a challenging dataset for face recognition. The researchers at the University of Oxford released this dataset to overcome the problem of label noise [32]. It contains over 362 images per subject on average.

2.1.2 Face Processing and Alignment

Face processing and alignment refer to the techniques used to preprocess facial images to prepare them for use in face recognition algorithms. The goal of face processing and alignment is to remove unwanted variations in the images, such as variations in lighting, pose, and facial expression, and to align the images so that the facial features are in consistent locations and orientations.

There has been a significant amount of research in the field of face processing and alignment, with many different techniques proposed for preprocessing facial images.

One common approach is to use face detection algorithms to locate the face in the image and then apply geometric transformations to align the face with a standard reference frame. This can involve scaling, rotation, and translation of the face, as well as warping of the facial features to correct for perspective distortions.

Another approach is to use deep learning models to perform face alignment and normalization. These models can be trained on large datasets of facial images to learn the transformations that are necessary to bring the facial features into a consistent orientation and size. One more techniques that have been used for face processing and alignment include illumination normalization, which adjusts the image to remove variations in lighting, and expression normalization, which removes variations in facial expression.

Overall, the goal of face processing and alignment is to standardize facial images to reduce the impact of unwanted variation and make them more suitable for use in

face recognition algorithms. The specific techniques used for face processing and alignment depending on the application and the characteristics of the input images.

Once a face has been detected in an image, it is typically localized using a bounding box, but this does not always guarantee that the face will be aligned. In addition, the bounding box may be subject to jitter, making it difficult to achieve accurate alignment [32]. To overcome these challenges and improve the robustness of deep convolutional neural networks (DCNNs) to spatial changes, a common approach is to apply data augmentation techniques.

There are several data augmentation techniques used in the computer vision community. The more common approaches used by [8] [38] include 2D data augmentation which is essentially perturbing the sample image to create multiple variable copies of it and then perform the average pooling in the feature space.

A 2D similarity is a transformation that is defined up to 4 Dof (scale, rotation, angle, and 2D translation) and therefore it has very limited deformation capability. 2D similarity transformations for face alignment using detected facial landmarks have been reported to yield better face recognition at inference time [27]. Some standard face recognition libraries also report using affine transformations based on open cv for facial landmark detection and alignment [50].

However, their deformation powers can be limited. This led to the development of complex face alignment by taking advantage of 3D face shapes using 3D generic models. The term "face frontalization" refers to the process of artificially transforming an image to make it appear as if the face is facing directly forward, using a 3D model to create the transformation [32].

However, with the availability of benchmarks such as IJB-A [25] and IJB-B [53], that provide a wide range of pose distributions for an image, it has been possible to perform augmentation in unseen poses [32]. [32] proposed a way to pre-compute the offline rendered multiple views so that a face can be viewed from any arbitrary frontal, half-profile, and full profile view. Moreover, computer vision is now beginning to develop robust deep based face estimation methods [47] [61].

This all advances in Face-specific augmentation have been shown to advance the improvement on several benchmarks such as IJB datasets and LFW. The advantage of face-specific augmentation is that it can be used to enlarge the intra-class variance of pose-deficient training sets.

More recently, GANs [62] have been widely used in face-specific augmentation to generate realistic and diverse synthetic face images that can be used for various face recognition tasks.

In GAN-based face-specific augmentation, the generator model is trained to synthesize realistic and diverse face images, while the discriminator model is trained to distinguish between real and fake images. During training, the generator tries

to generate images that can fool the discriminator, while the discriminator tries to correctly identify the fake images generated by the generator. This process continues until the generator can produce realistic and diverse images that can deceive the discriminator.

GAN-based face-specific augmentation has shown significant progress in various face recognition tasks, such as face verification, face identification, and facial expression recognition. This technique has also been used to generate synthetic face images for training deep learning models, reducing the need for large face image datasets.

2.1.3 Network Architectures

The network architectures used for Face Recognition can be broadly categorized into the following categories:

1. **Convolutional Neural Networks:** CNN's are the most common type of network used for face recognition. These networks consist of multiple convolutional layers that extract features from the input images, followed by fully connected layers that perform the classification. Examples of CNN-based face recognition networks include VGGFace, FaceNet, and DeepFace.

2. **Siamese Networks:** Siamese networks are used for one-shot or few-shot face recognition, where the network is trained to learn a similarity metric between a pair of face images. The Siamese network consists of two identical sub-networks that share the same weights. The output of the two sub-networks is then compared to compute the similarity score between the two faces.

3. **Capsule Networks:** Capsule networks are a relatively new type of network architecture that has shown promise in face recognition tasks. Capsule networks use capsules instead of neurons as basic building blocks, where each capsule represents a specific type of feature in the input image. Capsule networks have been shown to be more robust to variations in face poses and lighting conditions.

4. **Attention Networks:** Attention networks are used to selectively focus on specific regions of the face that are important for recognition. These networks learn to weigh the importance of different regions of the input image based on their relevance to the task. The attention mechanism has been used in conjunction with other network architectures such as CNNs to improve the accuracy of face recognition systems.

Several state-of-the-art models have been proposed in this regard. Some of them are described below :

AlexNet

AlexNet [26] is a deep convolutional neural network architecture that was proposed by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton in 2012. It was one of the first models to demonstrate the effectiveness of deep learning on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset.

The AlexNet model consists of eight layers, including five convolutional layers and three fully connected layers. It uses the rectified linear unit (ReLU) activation function and employs dropout regularization to prevent overfitting. The model also uses local response normalization (LRN) to improve its generalization performance.

AlexNet achieved state-of-the-art performance on the ILSVRC dataset in 2012, significantly reducing the top-5 error rate from the previous best model. Its success paved the way for the development of deeper and more complex neural network architectures, including VGG, ResNet, and Inception.

The AlexNet architecture consists of eight layers: five convolutional layers followed by three fully connected layers. The network takes in an input image of size 227 x 227 x 3 and produces a vector of class probabilities. The network has approximately 60 million parameters, making it a relatively deep architecture at the time of its proposal.

The AlexNet architecture also includes the use of Rectified Linear Units (ReLU) as activation functions, which helped to alleviate the vanishing gradient problem that had plagued deep neural networks for many years.

Overall, the AlexNet architecture was a major breakthrough in the field of deep learning and helped to establish CNNs as the leading architecture for image recognition tasks.

Deepface

Deepface [46] is used by Facebook for tagging images. It was proposed by researchers at Facebook AI Research (FAIR) at the IEEE CVPR in 2014. The model uses a 9-layer deep neural network that analyzes an input image and produces a 128-dimensional vector representing the facial features of the input image.

The architecture of DeepFace includes several stages, such as alignment, representation, and classification. In the alignment stage, the model aligns the input image to a canonical pose to reduce variations caused by pose and orientation. In the representation stage, the model computes a 128-dimensional vector representation of the face using a nine-layer deep neural network. his deep network involves more than 120 million parameters using several locally connected layers without weight sharing, rather than the standard convolutional layers. Finally, in the classification stage, the model uses a linear classifier to classify the input image as either matching or not matching a given reference image.THey use weighted x^2 and siaseme networks

as their evaluation metrics for results. They reported accuracy of 97.35 on Labeled Faces in the Wild (LFW) dataset.

VGG 16

The VGG16 architecture is a deep convolutional neural network (CNN) architecture that was proposed by the Visual Geometry Group (VGG) at the University of Oxford [42]. It is a widely used architecture for image classification and feature extraction tasks.

The architecture consists of 16 layers, hence the name VGG16. In VGG16 there are thirteen convolutional layers, five Max Pooling layers, and three Dense layers which sum up to 21 layers but it has only sixteen weight layers i.e., the learnable parameters layer.

It has a total of 138 million parameters, which makes it a very deep network. VGG16 takes input tensor size as 224, 244 with 3 RGB channels.

The notable feature of VGG16 is its emphasis on utilizing convolution layers with 3x3 filters and a stride of 1, as well as consistently implementing the same padding and a maxpool layer with a 2x2 filter and a stride of 2. This approach resulted in a reduced number of hyper-parameters compared to other architectures.

The first 13 layers of the network are convolutional layers, and the last 3 layers are fully connected layers.

After the convolutional layers, the network has 3 fully connected layers, each followed by a ReLU activation function. The last fully connected layer outputs a probability distribution over the classes in the dataset, which is used for classification.The VGG16 architecture is trained using the cross-entropy loss function, which is a standard loss function for classification tasks. The network is typically trained using backpropagation and stochastic gradient descent (SGD) with momentum.

Overall, the VGG16 architecture is known for its simplicity and effectiveness and has been shown to achieve high accuracy on a variety of image classification tasks.

RESNET Architecture

With the introduction of Alexnet which won the ImageNet 2012 competition, much of the effort in the subsequent network architectures was to design networks that make the network deep by adding more layers into the architecture in order to reduce the error rate.

But as we increase the number of layers, we face the vanishing gradient problem. This causes the gradient to become 0 or too large. Thus, as we increase the layers, the training error and test error also increases.

Residual Network, or ResNet, is a deep learning model designed to address the problem of vanishing gradients in very deep neural networks [20]. The vanishing gradient problem occurs when gradients become too small and are unable to update the weights of early layers in the network, making the network difficult to train. ResNet uses residual connections to allow for easier training of very deep neural networks.

In a ResNet model, residual blocks are used to connect input and output layers. A residual block contains a skip connection that bypasses one or more layers, allowing the network to learn more efficiently. The skip connection adds the input to the output of the residual block, so that the output is the sum of the input and the output of the residual block. This ensures that the gradient can flow directly from the output to the input, avoiding the problem of vanishing gradients.

The ResNet architecture is organized into several stages, each consisting of multiple residual blocks. The first stage of the network performs a convolution on the input image, followed by a series of residual blocks. Each stage doubles the number of filters and reduces the resolution of the feature maps by a factor of two. The final stage of the network contains a global average pooling layer, followed by a fully connected layer that produces the output classification.

ResNet has set the state-of-the-art in several computer vision tasks, including image classification, object detection, and face recognition. Its deep architecture and residual connections enable it to learn richer and more complex representations of images, leading to better performance on these tasks.

GoogleNet

The Inception Network has been a significant advancement in the domain of Neural Networks, particularly for Convolutional Neural Networks. There are currently three versions of the Inception Network, referred to as Inception Version 1, 2, and 3. The first version of the network was introduced in 2014 [43] and was developed by a team at Google, hence the name "GoogleNet". This network was instrumental in establishing a new benchmark for classification and detection in the ILSVRC. The initial version of the Inception Network is commonly known as GoogleNet.

Architecture was developed by a team of researchers at Google in 2014. It was designed to be very deep and yet computationally efficient, with the aim of improving classification accuracy on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset.

The key innovation of GoogleNet is the Inception module, which allows the network to efficiently use a combination of convolutional filters with different sizes (1x1, 3x3, 5x5) and pooling operations in parallel. This allows the network to capture both fine-grained and coarse-grained features at multiple scales, which is important for achieving high accuracy on the ILSVRC dataset.

GoogleNet has 22 layers, making it much deeper than previous state-of-the-art CNN architectures like AlexNet and VGG-16, while still being computationally efficient due to the use of the Inception module. It also includes other architectural innovations like the use of global average pooling instead of fully connected layers at the end of the network, which helps to reduce overfitting.

In the ILSVRC 2014 competition, GoogleNet achieved a top-5 error rate of 6.67, which was significantly lower than the previous state-of-the-art and helped to establish the effectiveness of the Inception module and deep CNN architectures in general. Since then, several other versions of the Inception architecture have been developed, including Inception v2 and Inception v3, which have continued to improve classification accuracy while maintaining computational efficiency.

2.1.4 Loss Function

The loss function is probably one of the most critical choice that would dictate the performance of the model. For a facial recognition system to perform well , it needs to learn how to increase the inter-class distance in feature space for images of different faces and reduce the intra-class variance of faces belonging to the same subject. Several loss functions have been proposed in the computer vision community. We briefly describe some of them.

Softmax Loss

The Softmax loss function [29] is a commonly used loss function in classification tasks, including face recognition. The goal of the Softmax function is to predict the probability distribution over a set of mutually exclusive classes.

Given an input image, a neural network produces a set of outputs corresponding to each class. These outputs are usually unnormalized and are referred to as logits. The Softmax function takes these logits and normalizes them to produce a probability distribution over the classes.

One of the drawbacks of using this loss function for transfer learning is that while it can effectively distinguish between different classes during training, it doesn't directly address the issue of minimizing intra-class variation within each subject.

Center Loss

Center loss [51] is a loss function used in face recognition that aims to learn a center for each identity in the feature space. It aims to mitigate the limitations of softmax loss. It's main utility was to reduce the intraclass varation of each subject. The updated loss function aims to minimize the distance within a class between a given sample x and its corresponding centroid c_i, which is continuously updated during the learning process.

In this approach, each identity is represented by a center vector, which is learned during the training process. The distance between a feature vector and its corresponding center vector is used to compute the loss. The center loss is combined with a softmax loss, which is used to classify the identities.

The center loss is defined as the Euclidean distance between the feature vector and its corresponding center vector. The objective is to minimize the sum of the distances between the feature vectors and their corresponding center vectors. The center loss is added to the softmax loss to train the deep neural network for face recognition. The softmax loss is used to classify the identities, while the center loss is used to learn the centers for each identity. The combination of both losses helps to improve the accuracy of the face recognition system.

Overall, the center loss is an effective loss function for face recognition, especially when dealing with large-scale datasets with a high number of identities.

SphereLoss

SphereFace loss [28] is a type of loss function used in face recognition tasks to learn discriminative features. It was proposed to address the limitation of the softmax loss function in dealing with the large intra-class variations and inter-class similarities in face recognition.

In the SphereFace loss, the angular margin between the features of different classes is increased to increase the discriminative power of the learned features. The angular margin is defined as the angle between the feature vector and the corresponding weight vector in the last fully connected layer.

Compared to the softmax loss, SphereFace loss has several advantages in face recognition tasks. Firstly, SphereFace loss can achieve higher accuracy by increasing the inter-class distance and reducing the intra-class variation. Secondly, SphereFace loss is more robust to changes in illumination, pose, and expression. Lastly, SphereFace loss is easy to implement and can be trained end-to-end using stochastic gradient descent.

However, SphereFace loss also has some limitations. It requires a large amount of training data to achieve high accuracy, and it may be sensitive to hyperparameter tuning. Additionally, the angular margin in SphereFace loss may not be optimal for all face recognition tasks, and other loss functions such as ArcFace and CosFace may be more suitable in some cases.

Triplet Loss

The triplet loss function [21] is a popular loss function used in face recognition tasks, where the goal is to learn a feature representation space that can accurately discriminate between different faces. The triplet loss function aims to learn embeddings in

which the distance between faces of the same identity is smaller than the distance between faces of different identities.

The loss is computed using triplets of images, consisting of an anchor image, a positive image (an image of the same identity as the anchor), and a negative image (an image of a different identity than the anchor). The triplet loss function encourages the distance between the anchor and positive image to be smaller than the distance between the anchor and negative image by a margin.

The advantage of the triplet loss function over other loss functions such as softmax is that it explicitly optimizes the embedding space to improve the discriminability between faces, leading to better face recognition performance.

ArcFace Loss

ArcFace loss is a popular loss function used for face recognition tasks. It was introduced in 2018 by Jiankang Deng, Jia Guo, and Stefanos Zafeiriou in their paper "ArcFace: Additive Angular Margin Loss for Deep Face Recognition".

The goal of ArcFace loss is to learn a discriminative embedding space for face recognition. The embedding space is a lower-dimensional space where each face image is mapped to a point. The distance between two points in this space should be small if the images belong to the same person, and large if they belong to different people [13].

The ArcFace loss is an extension of the softmax loss function. In the softmax loss function, the input to the network is transformed into a probability distribution over the classes, and the cross-entropy loss is used to measure the difference between the predicted and actual class labels.

In ArcFace loss, a new term called "angular margin" is added to the softmax loss function. The angular margin is a function of the angle between the embedding vector of the input image and the weight vector of the ground truth class. By adding this term, the network is encouraged to learn more discriminative features by increasing the angular separation between different classes.

The advantages of ArcFace loss over softmax loss include better discrimination between classes, improved generalization to unseen data, and better robustness to variations in lighting, pose, and expression.

CosFace Loss

Cosface loss function [49] is a variant of the ArcFace loss function used in face recognition. It is also known as the MarginCosine loss or LargeMarginCosine loss.

The Cosface loss function works by adding a cosine margin to the softmax loss function. The margin helps to increase the intra-class distance and decrease the

inter-class distance, making it easier for the model to differentiate between faces.

In ArcFace, the angular margin is added to the cosine similarity between the features and the weights of the corresponding class, while in CosFace, the angular margin is directly added to the dot product between the features and the weights of the corresponding class.

The angular margin is a hyperparameter that controls the degree of separation between classes, and is designed to make the learned features more discriminative. Both ArcFace and CosFace are designed to address the limitations of the traditional Softmax loss, which does not optimize for inter-class variance.

One advantage of CosFace over ArcFace is that it is computationally less expensive, as it involves a simple dot product instead of a more complex calculation of the cosine similarity. Another advantage is that of Cosface loss function over the ArcFace loss function is that it has a larger margin. This means that it is better at separating the classes and can lead to better accuracy in face recognition [61].

2.1.5 Similarity metric (Matching)

In this section , we describe the most common evaluation metrics and matching metrics that are using in the computer vision community in tasks of face recognition.

Cosine Similarity

In the context of face recognition, cosine similarity is a similarity metric that measures the cosine of the angle between two face feature vectors.

When a face image is fed into a deep learning model for feature extraction, it is transformed into a high-dimensional feature vector that represents the distinctive facial features of the image. The cosine similarity metric then calculates the cosine of the angle between two of these feature vectors to determine their similarity.

A cosine similarity score of 1 indicates that the two feature vectors are identical, while a score of 0 means that they are completely dissimilar.

Cosine similarity is widely used in face recognition [36] because it is computationally efficient and can handle high-dimensional feature vectors. It is often used with classification algorithms to determine the identity of a face image by comparing its feature vector with those of known identities in a database.

Euclidean Distance

In face recognition, Euclidean distance is a metric used to measure the distance between two face feature vectors in a multidimensional space. Each face is represented by a feature vector that captures important information about the facial attributes, such as shape, texture, and color. The Euclidean distance between two face feature

vectors is computed by taking the square root of the sum of the squared differences between each corresponding element of the vectors.

The distance value indicates how similar or dissimilar two face feature vectors are. If the distance is small, it means that the two faces are similar, and if the distance is large, it means that the faces are dissimilar[56].

2.1.6 Evaluation metrics

ROC (Receiver Operating Characteristic)

For face verification , ROC curve is used that reports the the recall (True Acceptance Rate - TAR) at multiple cutoff points of the precision (False Alarm Rate - FAR). n face verification, the Receiver Operating Characteristic (ROC) curve is used to evaluate the performance of a binary classification system, which determines whether two face images belong to the same person (positive class) or not (negative class). The ROC curve is a plot of the True Positive Rate (TPR) against the False Positive Rate (FPR) for different threshold values.

In face verification, the TPR is the proportion of positive face pairs that are correctly identified as positive, while the FPR is the proportion of negative face pairs that are incorrectly classified as positive. The threshold value represents the decision boundary for the classification system.

By plotting the TPR against the FPR for different threshold values, the ROC curve provides a visualization of the trade-off between the TPR and FPR for the classification system. A better performing system would have a higher TPR and lower FPR, resulting in a curve that is closer to the top-left corner of the plot.

CMC (Cumulative Matching Characteristic)

The Cumulative Matching Characteristic (CMC) curve is a performance evaluation metric in face recognition that measures the accuracy of the identification process. The CMC curve shows the probability of correctly identifying a person by ranking the similarity scores of their probe face image against a gallery of face images.

The CMC curve is plotted by ranking the similarity scores of the probe image against the gallery images and calculating the identification rate for each rank. The identification rate is the percentage of probe images correctly identified at or below a given rank.

For example, at Rank 1, the CMC curve shows the percentage of probe images that were correctly identified as the same person in the top-ranked image in the gallery. At Rank 5, the CMC curve shows the percentage of probe images that were correctly identified in the top five images in the gallery.

The CMC curve is useful in evaluating the performance of face recognition systems,

especially in situations where identification accuracy is critical, such as in security and surveillance applications. A good face recognition system should have a high identification rate at low ranks, indicating that it can accurately identify individuals even when there are many faces in the gallery.

The amalgamation of our progress in all these aspects has made drastic improvements in how face recognition systems are used, deployed and their overall utility. However, the problem of recognizing faces is not resolved completely. For example, video-based face recognition is still an open problem [81] and clustering of unlabeled faces are still an open problem and next frontiers for face recognition [32].

There is also a discussion over how to reduce the bias in the datasets. Hence, there is still a challenge in developing machines that can make unbiased predictions without being influenced by inherent factors such as ethnicity, gender, age, and others that may be present in the training data.

2.2 Ethical and Privacy Concerns on Face Recognition Systems

Although Face Recognition technology has many potential advantages for security and protection of citizens, its usage comes with many privacy concerns and ethical questions. As rightly pointed by [58], the discussion regarding the concerns on large scale use of Face Recognition systems mostly revolves around the topics of privacy, security, accuracy, bias, and freedom.

The issue of privacy is frequently brought up in discussions about facial recognition technology In early 2020, it was revealed that Clearview.AI [14], a facial recognition startup, had scraped billions of photos from social media platforms and other websites without users' consent. The company then built a powerful facial recognition tool that was sold to law enforcement agencies. This data leakage resulted in a major privacy violation, as millions of people's photos were used without their consent.

In February 2019, security experts uncovered a serious data leak at SenseNets [37], a security software company in Shenzhen that specialized in facial recognition. The leak, which came from an unprotected database, contained over 2.5 million records of citizens with personal information. In 2018, it was discovered that Cambridge Analytica [23] had harvested the data of millions of Facebook users without their consent. While this data leak was not specifically related to facial recognition technology, the incident did highlight the potential risks associated with the collection and use of personal data by technology companies. In 2019, the U.S. Department of Homeland Security experienced a data breach that resulted in the loss of millions of facial recognition photos and other personal data. This breach highlighted the potential dangers of storing sensitive biometric data in centralized databases [24] .

These data breaches can have serious and long-lasting consequences for victims, particularly given the almost-permanent nature of biometric information. In many cases, individuals are not aware that their biometric data is being collected or used. This is particularly concerning when it comes to law enforcement use of facial recognition systems, where individuals may be targeted or tracked without their knowledge or consent [40].

Although typically considered as a means of security identification, facial recognition should not be considered safe enough. Research has shown that GAN-generated Deepfakes[10] videos are challenging for facial recognition systems, and such a challenge will be even greater with the usage of deepfake technology.

Another impact of Face Recognition could be it could misidentify people belonging to certain ethnic groups and skin colours and lead to a bias. According to a study done by researchers at Harvard, the face recognition algorithms give divergent error rates across various demographic groups. They give poorest accuracy against subjects who are female, Black and 18-30 year old [34]. According to a study in 2018 with the name of "Gender Shades Project" by MIT University,the gender classification algorithms even developed by Microsoft and IBM reported high error rates on darker-skinned females [3].

The use of face recognition by law enforcement has also raised concerns that they can be used to target undocumented immigrants and and marginalized community citizens as rightly shown in this article by ICLU [11], Muslims after 9/11 were made subject to strict surveillance and checking for people with beards at airports around the globe [7].

Also , with rise and availability of large scale datasets such MS-Celeb-1M and LFW as well as availability of highly powerful feature extractors, there are potential for misue of this technology by private individuals , cyber hackers and stalkers [40].

Thus, it has become increasingly important to make tools and technologies that allow us to protect our faces from being identified by unauthorized face recognition systems. This can help people share their pictures online without an attack on their privacy and avoid unwanted profiling by companies and intruders.

2.3 Adversarial Attacks against Face Recognition

Adversarial images can be used to protect against some types of face recognition systems. Adversarial images are images that have been modified to include subtle perturbations that are imperceptible to the human eye but can cause the face recognition algorithm to misidentify the subject.

By creating adversarial images that are specifically designed to trick a particular facial recognition system, researchers have been able to show that it is possible to confuse the algorithm and cause it to identify the wrong person.

An adversarial attack on a face recognition system is a type of attack in which an attacker generates specific input samples, called "adversarial examples," to fool or mislead the face recognition model into making a wrong prediction or decision. In the context of face recognition systems, an attacker could create an adversarial example by slightly altering an image of a person's face in a way that is not noticeable to the human eye, but causes the face recognition system to misidentify the person or to identify them as someone else. There are different methods to create adversarial examples, such as adding noise to the image, changing the color or brightness of the image, or using techniques like "transfer learning" to create adversarial examples that are effective across different models or even across different tasks[

There are several adversarial attack techniques in digital domain that have been proposed for face recognition systems to prevent them from scrapping social media profile of user's for developing the face recognition system for mass surveillance. Additionally, adversarial attacks can be used to protect the privacy of individuals by generating adversarial examples that obscure or distort their facial features, making it more difficult for face recognition systems to correctly identify them.

Overall, adversarial attacks can be an effective way to protect against potential misuses and abuses of face recognition systems. A detailed discussion on Taxonomy of Adversarial attacks and their detailed discussion is provided in Chapter 3. Here we discuss the latest work related to protection against unwanted face recognition and a brief overview of the tools and research available in this regard that we studied during this thesis.

2.3.1 EqualAIs by MIT

EqualAIs is a project that was developed out of the 2018 Assembly Program at the Berkman Klein Center for Internet and Society at Harvard University and the MIT Media Lab. Equal AI is a tool that implements the FGSM attack on facial images and evaluates those attacks on a trained subsitute model as well as commercial API's such as Google Vision and Azure Face. They base their FGSM attack on cleaverhans library. The algorithm image attack is based on three steps :

1. An adversarial perturbation of the image to prevent face detection systems from identifying a face
2. A stenographic message encoding that explicitly states that a user does not consent to face detection
3. A visible watermark communicating that the user does not consent to face detection.

A visual depiction of attack is shown in Figure 2.2. [1].

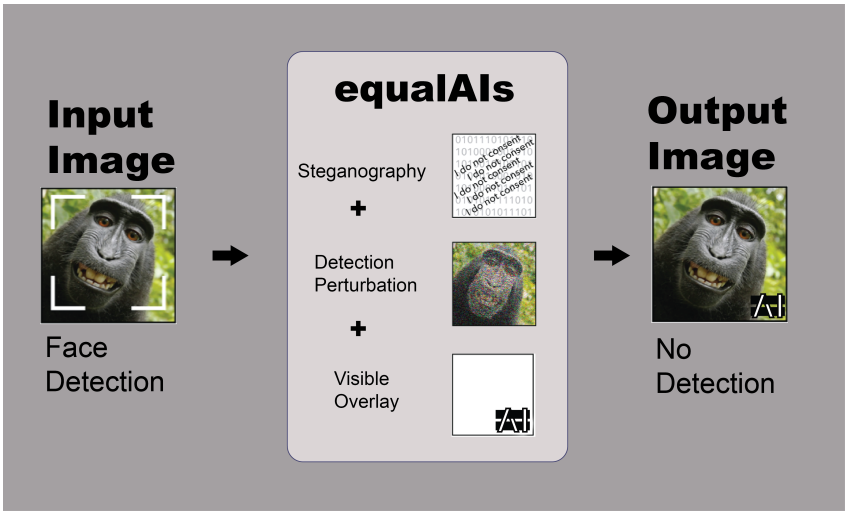


Figure 2.2. Equalai's attack

2.3.2 Fawkes

Fawkes[40] is a data-poisoning tool and algorithm designed by Sandbox to protect user's privacy against unauthorized deep learning models. The main goal of Fawkes is to protect privacy of a user from being against unauthorized deep learning models trained by a third party tracker on user's images [40].The main idea behind the technique, called Fawkes, is to create a modified version of the original data, called a "cloaked" version, that preserves the utility of the data for authorized models, but is not useful for unauthorized models. The facial recognition models trained on cloaked images would have distortion in feature space and cannot recognize probe images of a user and would likely misclassify them.[40].

Fawkes achieves this by helping users add imperceptible pixel-level changes (i.e clock) to their own photos before releasing them. When used to train facial recognition models, these "cloaked" images produce functional models that consistently cause normal images of the user to be misidentified. They design the perturbations/clocks in such a way that features learned from clocked photos of a user are highly dissimilar to the ones learned from uncloaked photos of the same user.

The mathematics behind the Fawkes tool, as presented in the paper "Fawkes: Protecting Privacy against Unauthorized Deep Learning Models," involve the following notations:

- x :represents the image of a user.
- x_T is the target image which is from another user T that is used to generate clocks for user x.
- $\delta(x, x_T)$ is clocks computed for user x based on image x_T from another label/user T

- $x \oplus \delta(x, x_T)$: clocked version of user image
- Φ: Feature extractor of facial recognition model.
- Φ(x) : feature vector extracted from input x

The clocking maximizes the feature distance by modifying x by adding a perturbation (clock) $\delta(x, x_T)$ to x to maximize the changes in x's feature space.

$$\begin{aligned} &\min_{\delta} Dist\left(\Phi(x_T), \Phi(x \oplus \delta(x, x_T))\right), \\ &\text{subject to } \ |\delta(x, x_T)| < \rho. \end{aligned} \tag{2.1}$$

where $Dist(.)$ computes the distance of two feature vectors, $|\delta|$ measures the perceptual perturbation caused by cloaking, and ρ is the perceptual perturbation budget.

A visual depiction of Fawkes attack pipeline is shown in Figure 2.3.

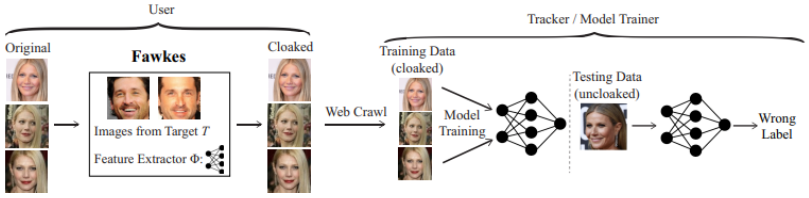


Figure 2.3. Fawkes Attack pipeline [40]

Fawkes claims that it's clocked images ensure protection against VGG2-Incept, VGG2-Dense,Web-Incept,Web-Dense feature extractor models in whitebox setting.While they claim that their model protects against commercial scale API's of Microsoft Azure, Amazon Rekognition and Face ++ and provide a protection success rates > 95% in black-box setting.

2.3.3 Lowkey

Lowkey [9] is the state of the art adversarial attack generation technique. The authors develop their own adversarial filter that accounts for the entire image processing pipeline and is demonstrably effective against industrial-grade pipelines that include face detection and large-scale databases.

They design a black-box adversarial attack on facial recognition models. Lowkey algorithm moves the feature space representations of gallery faces so that they do not match corresponding probe images while preserving image quality[9]. Lowkey also releases an easy-to-use web tool that significantly degrades the accuracy of Amazon Rekognition and the Microsoft Azure Face Recognition API, reducing the accuracy of each to below 1% .

LowKey applies a filter to user images which may end up in an organization's database of gallery images. The result is to corrupt the gallery feature vectors so that they will not match feature vectors corresponding to the user's probe images. LowKey does this by generating a perturbed image whose feature vector lies far away from the original image, while simultaneously minimizing a perceptual similarity loss between the original and perturbed image.Maximizing the distance in feature space prevents the image from matching other images of the individual, while the perceptual similarity loss prevents the image quality from degrading[9]. A visual representation of the lowkey attack pipeline is shown in the following figure which has been taken from their paper [9].

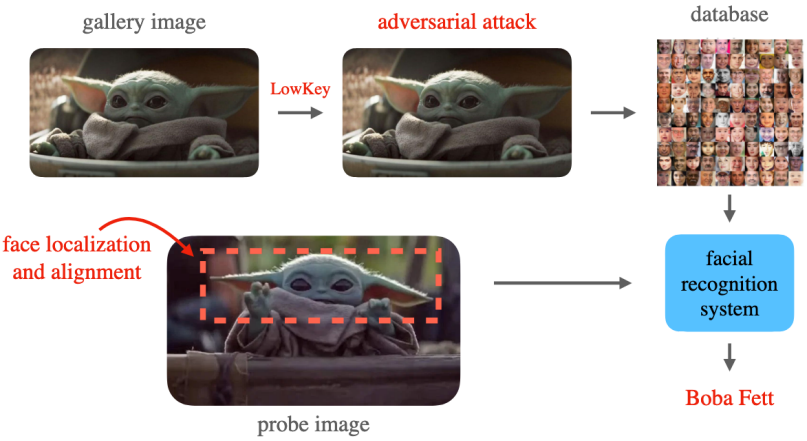


Figure 2.4. Lowkey Attack Pipeline [9]

LowKey is designed to avoid detection by private facial recognition (FR) systems that have undisclosed pre-processing procedures and neural network frameworks. To enhance the tool's ability to work on unknown FR systems, LowKey targets multiple models simultaneously, each with distinct backbone architectures and produced using various training methods.The big difference between Lowkey and Fawkes is that Fawkes attacks the training on face rec. while Lowkey attacks the inference phase itself.

The attack's objective function considers the attacked image's feature vector positions with and without a Gaussian blur for every model in the collection. This technique leads to an improvement in the attacked images' appearance and transferability. Figure 2.5 visually demonstrates the lowkey attack on faces.

2.3.4 FaceOff: Adversarial Face Obfuscation

Face-Off [6] employs black-box adversarial attacks that are transferable, utilizing the Carlini-Wagner [5] and Projected Gradient Descent [30] attacks to safeguard individuals' faces from facial recognition models. The Face-Off pipeline is illustrated in Figure 2.6.

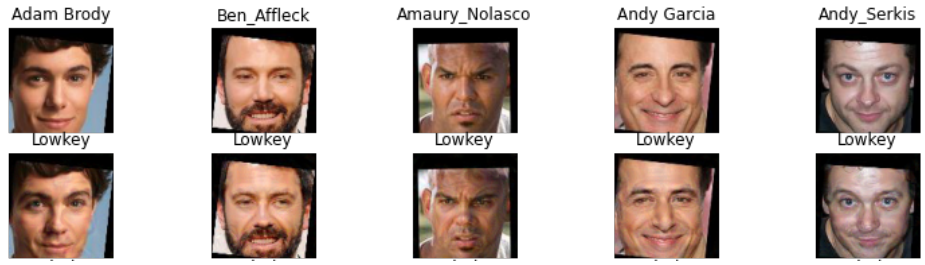


Figure 2.5. Original Faces vs Protected by Lowkey

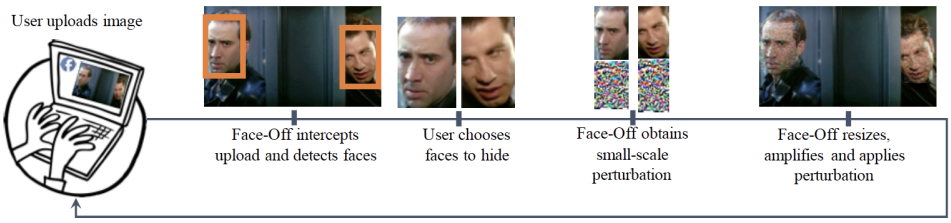


Figure 2.6. Face Off Attack Pipeline

The ultimate goal of Face-Off is to add the adversarial perturbations to the probe images so that the images have adversarial embedding that is far from the source label. They also discuss a wide range of innovative loss metrics to generate adversarial examples against face classifiers in their paper. Face-Off successfully achieves a significant relative decrease in accuracy (approximately -50%) on three commercial face recognition APIs, but it comes with a downside of visually conspicuous noise distortions and extended processing times.

Chapter 3

Background

This chapter will highlight the key terminologies and concepts that a user must be familiar with in order to understand the concepts and work that is described in Chapter 4.

3.1 Face Recognition Terminology

Face recognition (FR) has been a prevalent biometric technique for identity authentication and is broadly used in several areas, such as finance, military, public security, and daily life. A typical FR system's ultimate goal is to identify or verify a person from a digital image or a video frame taken from a video source. Researchers describe FR as a biometric artificial intelligence-based application that can exclusively identify a person by analyzing patterns of the person's facial features. The difference between these two realms of face recognition is described below:

3.1.1 Difference between Face Verification and Face Identification

1. **Facial Verification:** Face verification is 1:1 matching face recognition. It involves comparing two images of faces to determine if they belong to the same person. The goal of face verification is to answer the question "Is this the same person?" by comparing two images, and the result is a binary "yes" or "no" answer. This task is often used in security systems and more recently in mobile phones for access control, where a person's face is compared to an authorized image to verify their identity.

2. **Facial Identification:** Face identification involves comparing a given face image to a database of known faces to determine the identity of the person in the image. The goal of face identification is to answer the question "Who is this person?" by matching the given face image to a set of possible identities in a database. Face identification entails the 1:N face recognition where you compare a given face image against images in a database to find potential

matches. This task is often used in applications such as law enforcement, where a suspect's face is compared to a database of criminal records to identify potential matches.

3.1.2 Gallery

Gallery images refer to images in a database that have known identities, usually sourced from passport photos or social media profiles. The gallery serves as a point of reference for comparing newly added images.

3.1.3 Probe

Probe images are newly captured photographs that the user of the facial recognition (FR) system intends to identify. These images are often obtained from video surveillance footage and are fed into the FR system for comparison with gallery images with known identities. The FR system then determines if there are any matches. Our work is related to Face Identification.In this setting, the face recognition system tries to associate a person with an identity from a set of identities in gallery images stored in database. Given a with a probe image,the system compares the features of the probe image with the gallery images to find the closest matching neighbour in the gallery and determine the identity of the probe image.

3.1.4 Cosine Similarity

Cosine similarity is a metric that determines the similarity between features of faces. It mathematically calculates the cosine of the angle between two features that are projected in a multi-dimensional space. Cosine similarity between two vectors x and y is defined as:

$$C.S(x, y) = \frac{x^T y}{||x||||y||} \tag{3.1}$$

3.1.5 Learned Perceptual Image Patch Similarity (LPIPS)

LPIPS is a metric used to quantify the similarity between two images. It is a learned distance metric that measures the perceptual differences between images [60] .

LPIPS is based on deep neural networks that are trained to predict human judgments of image similarity. It operates by dividing the images into small patches and then computing the perceptual similarity between corresponding patches in the two images. The similarity scores are then averaged across all patches to obtain an overall similarity score. LPIPS essentially computes the similarity between the activation's of two image patches for some pre-defined network in PyTorch. This measure has been shown to match human perception well. A low LPIPS score means that image tensors are perceptual similar.

LPIPS has shown to be a more reliable and robust metric than traditional metrics like mean squared error (MSE) and peak signal-to-noise ratio (PSNR). It has been used in various applications, including image restoration, image compression, and image generation [9] .

3.1.6 Facial Landmarks

Facial landmarks are specific points on the human face that are used in face processing to identify and analyze facial features. In face processing, facial landmarks refer to key points or locations on a human face that can be used to perform tasks such as facial recognition, facial expression analysis, and facial feature manipulation. Facial landmarks are specific points on the face that are commonly used in affine transformations to align and transform images of faces. In this context, facial landmarks are typically used as reference points to compute the affine transformation matrix that maps one face to another.

In affine transformations, the facial landmarks are used to compute the scaling, rotation, and translation parameters necessary to align the faces. For example, to align two faces using affine transformations, the algorithm would first detect and locate the facial landmarks in both images. It would then compute the transformation matrix that maps the landmarks in one image to the corresponding landmarks in the other image. Finally, the transformation matrix would be used to apply the necessary scaling, rotation, and translation to one image to align it with the other.

3.2 Adversarial Attack Taxonomy

Before diving into the details of how adversarial attacks are generated it is important for us to be familiar with the key concepts used in Adversarial Machine Learning.

3.2.1 General Definition

- **Adversarial example**:An adversarial example x' is a manipulated version of a clean image x intended to deceive the deep learning model by introducing slight modifications (e.g., noise). The alterations are typically not noticeable, although in some instances, they may be perceptible.
- **Adversarial perturbation**: It represents the noise that is inserted into a clean image to create an adversarial example.
- **Adversarial Training**:Adversarial training is a machine learning technique used to improve the robustness and security of deep learning models against adversarial attacks.
- **Threat model**:A threat model formalizes assumptions and provides a description of the adversary's capabilities, knowledge, and goals, as well as the

potential attack scenarios and the resources available to the attacker.

3.2.2 Adversarial attack Scenario

In their study, Szegedy et al [44] demonstrated the effect of small perturbations on images in the image classification problem. This led to the development of adversarial examples that can cause high misclassification rates, even in state-of-the-art Deep Neural Networks (DNNs). Since then, many methods have been proposed for creating adversarial attacks, and they can be classified based on the level of knowledge the attacker has over the model.The adversarial capacity is determined by the amount of knowledge the attackers could gain about the model. Thus adversarial attacks in deep FR systems are classified into the following types according to the attack's capacity:

- **Whitebox-attack**: In white-box setting, the attacker has full knowledge of the target model's internal structure, parameters, and architecture. This information can be obtained by either reverse-engineering the model or by having legitimate access to the model's source code. White-box attacks are often the most effective because the attacker has complete information about the model and can easily craft adversarial examples.
- **Blackbox-attack**:In black-box attacks, the attacker has no access to the target model's internal structure or parameters. Instead, the attacker can only observe the inputs and outputs of the model. Black-box attacks are often more challenging to execute since the attacker has limited knowledge about the model's inner workings.
- **Greybox-attack**: In grey-box attacks, the attacker has some partial knowledge of the target model's internal structure, parameters, or architecture. This can be obtained by querying the model multiple times with different inputs and observing its outputs

3.2.3 Adversarial Specificity

Adversarial Specificity refers to the concept of ability of the adversarial attack to allow a specific intrusion or create general problems[48]. The attacks can be categorized into the following types based on their specificity.

- **Targeted Attack**:When conducting a targeted attack, the goal of the adversary is to create an adversarial example that causes the machine learning model to misclassify it as a particular target class. For instance, if the original input image is classified as belonging to class A, the attacker might try to modify the image so that the model misclassifies it as class B. Targeted attacks are typically more challenging to create than non-targeted attacks because the adversary must carefully optimize the perturbations in order to manipulate the model's prediction toward the intended target class.

- **Non-targeted attack**: In a non-targeted attack, the adversary aims to generate adversarial examples that are misclassified by the model, but without any specific target class in mind. In other words, the goal is simply to cause the model to make a wrong prediction. Non-targeted attacks are typically easier to craft than targeted attacks since the adversary does not need to optimize the perturbations for a specific target class.

Both targeted and non-targeted attacks can be used to evaluate the robustness of machine learning models and to improve their resistance to adversarial attacks. Defenses against adversarial attacks may be developed based on the type of attack expected to be encountered in the threat model.

3.2.4 Adversarial Transferability

Adversarial transferability refers to the property of adversarial examples where they can fool multiple machine learning models, even those trained on different datasets or architectures. In other words, an adversarial example generated to deceive one machine learning model may also be effective against other models, including those that are designed to perform different tasks. This property is especially important in black-box attacks, where an adversary may not have access to the target model's training data or other learning parameters. In such cases, the adversary may train a substitute neural network model and generate adversarial examples against it. Because of the transferability of adversarial examples, the target model will also be susceptible to these attacks, even though it was not the original model used to create the adversarial examples.

3.2.5 Adversarial Perturbations

An adversarial perturbation is a type of disruption that can fool a given model on a particular image with high probability. Smaller perturbations are what we aim for to create adversarial examples. In adversarial attack, the goal is to find the minimum norm of an adversarial perturbation that can cause a target model to misclassify an input instance. Specifically, when given an input image x, the perturbation vector h is crafted to change the label of x while keeping the perturbation as small as possible, such that the distance from x to the decision boundary of the classifier is minimized. where d is the dimension of the input image and the corresponding perturbation vector. The perturbations can be further categorized into the following types.

Image Specific Perturbations

[48] defines the image specific perturbations as the ones that can be explicitly generated according to the given input image.

Universal Perturbations

Universal perturbations can be created without requiring any prior knowledge of the specific details of the input images. The term "universal" refers to the ability of a perturbation to be applied uniformly to all input data and have good transferability. While universal perturbations facilitate the creation of adversaries in practical scenarios, current attacks mainly generate perturbations that are specific to each image. Therefore, it is an ongoing research in the field to is to develop universal perturbations that can be applied without the need for reformulation when the input samples are changed..This would in principle improve the efficiency and effectiveness of universal perturbation-based attacks in real-world applications.

3.3 Adversarial Attacks

An adversarial attack refers to a deliberate alteration of input data to manipulate classifier predictions while maintaining human interpretability of the content. Deep learning classifiers are vulnerable to such attacks, as demonstrated in previous studies where models have been fooled into incorrectly classifying objects, such as a stop sign as non-existent, a cat as guacamole, or a toy turtle as a rifle. In our scenario, the classifiers are neural network models trained to detect faces in images. The adversarial attack involves modifying the images in a way that human observers can still recognize the faces, but the model fails to detect them and returns a high confidence score indicating the absence of any identity or misclassify them.

Several prominent attacks have been studied in this regard. We mention some we studied over the course of this thesis below.

3.3.1 Fast Gradient Sign Method (FGSM)

The Fast Gradient Sign Method (FGSM) is a technique used in adversarial machine learning. It was first introduced in 2014 by Goodfellow et al[16] as a way to attack neural networks used for image classification. The method involves calculating the gradient of the loss function with respect to the input of a given model, and then modifying the input images in the opposite direction of the gradient to maximize the loss. This technique is one of the earliest examples of such attacks and has been widely studied and applied in the field of adversarial machine learning.

It is an algorithm for L_∞ adversarial attacks metric. Given an image x FGSM calculates the perturbations as follows:

$$\mathbf{x}^* = \mathbf{x} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} J(\theta, \mathbf{x}, y)) \tag{3.2}$$

3.3.2 Projected Gradient Descent

PGD can be seen as an extension for FGSM. PGD extended the one step gradient ascent idea. This method iteratively adjusts the direction that maximizes the loss of the classifier by running multiple steps. In each iteration, the values of the pixels in an image are clipped as follows:

$$x^* = Clip(\epsilon)(x^* + \alpha \text{sign}(\nabla_x L(F_\theta(x^*), y))) \tag{3.3}$$

The equation is used to calculate the adversarial example generated by the Projected Gradient Descent (PGD) attack. The adversarial example is denoted by $x^\star$, and it is calculated by taking the original input image x and adding a small perturbation that is calculated using the gradient of the loss function J with respect to the input x. The perturbation is scaled by a hyperparameter epsilon ϵ, and then clipped to ensure that the resulting adversarial example is within a certain distance from the original input image. The sign function is used to determine the direction of the perturbation. The parameter α controls the step size of the PGD attack. Finally, the output of the neural network F_θ with the adversarial example as input, denoted by $F_\theta(x^\star)$, is compared to the true label y using the loss function L. The overall goal is to minimize the loss function L and generate an adversarial example that is misclassified by the neural network.

3.4 Lowkey Adversarial Attack

In LowKey attack, the aim is to modify potential gallery images in such a way that they no longer match the probe images of the same person. The approach involves generating a perturbed image with a feature vector that is far from the original image, while minimizing a perceptual similarity loss between the original and perturbed images. By maximizing the distance in feature space, the modified image is prevented from matching other images of the same individual, while maintaining the quality of the image through the perceptual similarity loss.

Formally, the optimization problem is solved as:

$$\max_{x'} \frac{1}{2n} \sum_{i=1}^{n} \frac{\overbrace{\|f_i(A(x)) - f_i(A(x'))\|_2^2}^{\text{non-smoothed}} + \overbrace{\|f_i(A(x)) - f_i(A(G(x')))\|_2^2}^{\text{smoothed}}}{\|f_i(A(x))\|_2} - \alpha \underbrace{\text{LPIPS}(x, x')}_{\text{perceptual loss}}, \tag{3.4}$$

Here x is the original image, x' is the perturbed image. f_i denotes the i^th model in the ensemble. .They solve this maximization problem with a Signed Gradient Descent(SGD). They iteratively update x' by adding the sign of the gradient of maximization objective in Eq 3.4 with respect to x'. This moves the x' and $G(x')$ away from the original image x in the feature space of models. Figure3.1 shows the adversarial image that is derived after applying the lowkey attack.

Figure 3.1. Adversarial attack example lowkey

Chapter 4

Methodology and Experiments

In this chapter we shall describe the tools we used, the methodology applied, and the set of controlled experiments that we conducted as part of this thesis.

We follow the methodology to compute adversarial attacks as implemented by [9]. We do a comprehensive study of their research and work to understand and reproduce their work. We also evaluate the model in white-box setting by defending against our attacks using models from [50]. Then, we perform the controlled experiments using adversarial blur transformations from Kornia to evaluate the robustness and transferability of the attack.

In the end, we incorporate the highly impactful transformations as part of our adversarial attack pipeline to generate adversarial perturbations that provide greater protection even when denoising filters are applied during inference by the Face Recognition Systems.

Before we move to our results and experiments, we first provide a general overview of tools, libraries, models, and algorithms we use to implement this thesis.

4.1 Face.evoLVe: Face Recognition library

Face.evoLVe library is a comprehensive framework that contains popular deep learning-based methods for face recognition. It includes key components covering the entire face recognition and face-related analytics such as face alignment, data processing, various training backbones, losses, and some performance enhancement techniques.

The library supports multi-GPU training on PyTorch, making it suitable for both large-scale and low-shot datasets. Additionally, the library provides pre-aligned images for some benchmark datasets, source codes, and pre-trained models reducing the technical barriers for comparison and enabling users to focus on developing

advanced approaches more efficiently. Finally, the library is designed to be easily expandable to accommodate new face recognition approaches.

We use this library to perform face detection, landmark localization, and alignment with affine transformations on all the images in our database. We make use of ResNet-152 and IR-152 backbones that are trained with CosFace and ArcFace Heads using the training routines from Face.evoLVe [50].

The models IR-152 and ResNet-152 are part of our ensemble to generate adversarial attacks and also act as feature extractors in order to evaluate the top 1 and top 5 accuracy of our attack.

4.1.1 Face Alignment and Detection

Face Alignment and detection is a critical step in the Face Recognition pipeline as well as in adversarial attacks that focus on facial landmarks to compute efficient attacks. The faces are detected and extracted followed up with resizing and alignment.

Figure 4.1 shows the results of an image whose face is detected using facial landmarks and then the image is cropped and resized with face only.

Here is how briefly it works in our code. First, the script loads a pre-trained facial landmarks detection model from the models directory. The model is a dlib-based facial landmark detector that can detect 68 facial landmarks. Then, the script reads an input image and converts it to a grayscale image. Next, the get_reference_facial_points() function is called to get the coordinates of the reference facial landmarks. These landmarks are used as a template for aligning the face. The reference points in our case refer to the 112 x 112 size. The facial landmarks detector is then used to detect the facial landmarks in the input image.

Figure 4.1. Face localization and alignment pipeline

One key part of our pipeline is to crop and align the face based on the detected facial landmarks and the reference landmarks. This function first computes the affine transformation matrix that maps the detected landmarks to the reference landmarks. Then, transformations are applied to the input image to align the face. Finally, the faces are saved in the output directory.

We use this as a pre-processing step in our pipeline. We first detect , align and crop faces from images before they are fed to feature extractors for feature extraction and matching. This thus eliminates , the other distractions such as lighting, illumination that could disrupt the face identification.

4.1.2 Feature Extractor Models

We use ResNet-152 and IR-152 that are trained using Cos-face and Arc-Face loss functions through the training routine from the library. The models are named IR-152A, IR-152C, ResNet-152A, ResNet-152C. We use an ensemble of these models to perform the adversarial attack on images and do controlled experiments. These feature extractor models have been pre-trained on the MS-Celeb-1M dataset.

ResNet-152 Model

ResNet-152 is a deeper and more complex neural network architecture than the original ResNet-50 architecture. It has 152 layers in total, as compared to ResNet-50's 50 layers, allowing it to capture more intricate and abstract features from input images.

IR-152 Model

IR-152 is a deep convolutional neural network architecture that is commonly used for face recognition. The network is made up of 152 layers, hence its name, and is based on the Inception-ResNet-v1 architecture.

The IR-152 model is trained using a combination of a softmax loss function and the ArcFace loss function. The softmax loss function helps to minimize the intra-class variance, while the ArcFace loss function helps to maximize the inter-class variance by introducing an angular margin between different classes. This makes the model more discriminative and helps it to better distinguish between different faces.

The IR-152 model also uses residual connections, which help to address the vanishing gradient problem during training. This allows for faster convergence and better performance on large datasets. It is a variant of the Inception-ResNet-v1 model and was proposed by researchers at the Chinese Academy of Sciences in their paper "SphereFace: Deep Hypersphere Embedding for Face Recognition". The IR-152 model uses residual blocks with bottleneck designs, similar to ResNet, and also includes the inception module to capture multiple feature scales.

In Face.evoLVe library, the IR-152 model backbone is used as a feature extractor for face recognition tasks. It is a modified version of the ResNet architecture, where the residual blocks are grouped into several stages with different numbers of blocks and feature map sizes. The "IR" in IR-152 stands for "insightful representation", indicating that the model is designed to learn features that are especially useful for face recognition tasks.

4.1.3 ArcFace Loss Function

ArcFace is a widely used face recognition model that is used to learn discriminative features for face recognition. The ArcFace head is a specific type of neural network architecture that is used to learn these features. It is composed of a fully connected layer that maps the output of the backbone network to an embedding space of a fixed dimension, followed by a normalization layer that normalizes the embedding vectors to have unit length. The final layer is a classification layer that predicts the identity of the input face using a softmax function. The key innovation of ArcFace is that it introduces an additive angular margin to the logits of the classification layer, which makes the embedding vectors more discriminative and enhances the separation between different identities. This results in higher accuracy and better generalization performance of the face recognition model.

4.1.4 CosFace Loss Function

CosFace is another type of loss function that can be used with the backbone models in face recognition systems like face.evoLVe. Like ArcFace, CosFace is also designed to increase the inter-class variation and intra-class compactness of feature embeddings, with the aim of improving face recognition performance.

The main difference between the two is the way they compute the final feature embedding. In CosFace, a scaling factor is added to the cosine similarity between the feature embedding and the weight matrix of the classifier. The added scaling factor ensures that the cosine similarity of the embedding with its true class weight is greater than that with any other class weight by a certain margin, which is a hyperparameter that needs to be set.

In contrast, ArcFace uses an angular margin penalty in addition to a scaling factor. The angular margin penalty pushes the embedding further away from its true class weight, while at the same time increasing the margin between it and the other class weights. This makes the embeddings more discriminative and better suited for face recognition tasks.

In terms of performance, both ArcFace and CosFace have been shown to outperform the traditional Softmax loss function in face recognition tasks. However, studies have suggested that ArcFace tends to perform better in cases where there are a large number of classes, while CosFace may be more suitable for smaller datasets.

4.2 Kornia Transformations

Kornia is an open-source computer vision library in PyTorch, designed to perform various image and video processing tasks, such as geometric image transformations, filtering, color space conversion, feature extraction, and more. It is developed and maintained by a group of researchers and engineers at Facebook AI Research.

Kornia provides a comprehensive set of tools for classical and deep learning-based image processing tasks that are efficient, differentiable, and can be seamlessly integrated with PyTorch-based deep learning models. Its main focus is on the geometric operations that allow for the manipulation of images in a continuous manner, which is essential for developing vision-based deep learning models.

The library includes a wide range of functionalities, such as affine and perspective transformations, homography, image warping, image filtering, convolution, edge detection, feature extraction, color space conversion, gradient computation, optical flow, and more. The core operations of Kornia are implemented in a differentiable manner, meaning that they can be backpropagated through a neural network, making it easier to train models that use these operations.

The setting we use the Kornia library is for applying blur transformations to the images before features are extracted so as to evaluate the robustness of lowkey attack proposed by [9] against the blur transformations described below.

4.2.1 Median-Blur

Median blur is a type of image filtering technique used to reduce noise in digital images. The basic idea of median blur is to replace the color value of each pixel with the median value of neighboring pixels within a defined kernel size. The kernel is a sliding window that moves through the image, pixel by pixel. The median of the neighboring pixels is then computed and assigned to the central pixel. This process is repeated for every pixel in the image, resulting in a new image with reduced noise and smoother edges.

4.2.2 Box Blur

Box blur is a type of image blur that is achieved by averaging the color values of neighboring pixels within a square or rectangular window (also known as a "kernel" or "filter"). The size of the window determines the strength of the blur effect, with larger windows creating more blur. Box blur is commonly used in image processing to reduce noise, smooth out details, and create a more uniform appearance.

The box_blur function applies a box filter of a specified size to an input image tensor. The size of the box filter is specified using the kernel_size parameter, which should be an integer or a tuple of two integers representing the height and width of the filter.

During the box blur operation, the function slides the box filter over each pixel of the input image tensor, computes the average of the pixel values within the filter window, and replaces the center pixel with the computed average value. The process is repeated for all pixels in the image, resulting in a smoothed image.

The function also supports optional parameters such as border_type and normalized, which control how the filter handles image borders and whether the output values should be normalized.

4.2.3 Motion Blur

Motion blur is a common type of blur that occurs in images when the camera or the object being photographed is in motion during the exposure time. It results in streaks or smears of the moving objects in the image.

The appearance of the motion blur depends on various factors such as the speed of the moving object, the direction of motion, the camera's shutter speed, and the distance between the camera and the object.

Motion blur can have artistic value in photography, as it can convey a sense of movement or speed, but it can also degrade image quality and reduce the clarity of details in the image. In computer vision and image processing, motion blur is often considered a type of noise that needs to be reduced or removed to improve the quality of the image.

The motion-blur filter in Kornia applies a motion-blur effect to an input image. This filter is useful for simulating the blur that occurs when capturing images of moving objects with a camera. The amount of blur is controlled by the kernel size and angle parameters.

The motion blur filter works by convolving the input image with a kernel that simulates the motion of a moving object. The kernel has a rectangular shape and a length that is proportional to the desired blur strength. The angle parameter specifies the direction of the motion, which is the angle in degrees from the horizontal axis.

4.2.4 Max Blur

Max blur is a type of image blurring technique that replaces each pixel in an image with the maximum value of a local neighborhood. This technique is also known as "maximum filter" or "dilation" in image processing.

In Max-blur, a kernel or a window of a fixed size is passed over the image, and for each pixel in the image, the maximum value within the kernel is determined. This maximum value then replaces the original pixel value, resulting in a blurred image.

In the context of Kornia, the Max Blur filter is applied to a tensor image, where

the maximum value within a rectangular kernel is computed for each pixel location. The rectangular kernel has a specified width and height, which can be adjusted depending on the desired level of blurring.

4.2.5 Pool Blur

Pool blur, also known as average pooling, is a filtering technique commonly used in computer vision to reduce the size of an image or feature map while retaining its most salient features. It works by dividing an image or feature map into a set of small, non-overlapping rectangular regions (pools) and computing the average value of each pool.

The pool blur filter implemented in Kornia performs average pooling over a specified kernel size, which determines the size of each pool. For example, a kernel size of 3x3 means that each pool is a 3x3 rectangle of pixels, and the output is obtained by computing the average of each pool. This results in a smoothed version of the input image or feature map, where each pixel value represents the average value of a small region around it.

Pool blur can be used for various tasks, such as downsampling an image or feature map to reduce its resolution, or reducing the amount of noise in an image by averaging out small variations in pixel values. It is a simple and effective technique that can help improve the performance of many computer vision algorithms.

4.2.6 Gaussian Blur

Gaussian blur is a widely used image smoothing technique in computer vision and image processing. It is a type of linear filter that is used to reduce noise and details in an image by convolving the image with a Gaussian kernel. The Gaussian kernel is a square matrix of size N x N, where N is an odd number, and the values in the kernel are determined by the Gaussian distribution function.

The convolution operation involves sliding the kernel over each pixel of the image, and at each position, the values of the surrounding pixels are multiplied by the corresponding values in the kernel, and the sum of these values is taken. The resulting value is then assigned to the center pixel.

The values in the Gaussian kernel are arranged in a way that the kernel assigns more weight to the central pixel and less weight to the surrounding pixels, following the bell-shaped curve of the Gaussian distribution. This ensures that the filter smooths out the image while preserving its edges and important features.

The amount of smoothing applied to the image is controlled by the size of the kernel and the standard deviation of the Gaussian distribution. Larger kernel sizes and higher standard deviations result in stronger smoothing effects.

Mathematically, the Gaussian blur operation can be defined as a convolution of an image with a Gaussian kernel. Given an image I and a Gaussian kernel G, the output of the Gaussian blur operation, B, at pixel (i,j) can be computed as:

$$B_{i,j} = \frac{1}{\sum_{m=-k}^{k} \sum_{n=-k}^{k} G_{m,n}} \sum_{m=-k}^{k} \sum_{n=-k}^{k} G_{m,n} \cdot I_{i+m,j+n} \tag{4.1}$$

where k is the size of the Gaussian kernel and G is the Gaussian kernel defined as:

$$G_{m,n} = \frac{1}{2\pi\sigma^2} e^{-\frac{m^2+n^2}{2\sigma^2}} \tag{4.2}$$

Here, σ is the standard deviation of the Gaussian distribution, which controls the degree of smoothing. A larger value of σ results in a more blurred image.

4.3 Adversarial Attack

For generating the adversarial attack, we follow the work of [9] which is described in their paper [9] which was released in 2021. We re-design their attack methodology, perform controlled experiments with Kornia blur transformations and add our modifications to the attack which make it more robust and hard to defend against by commercial FR systems.

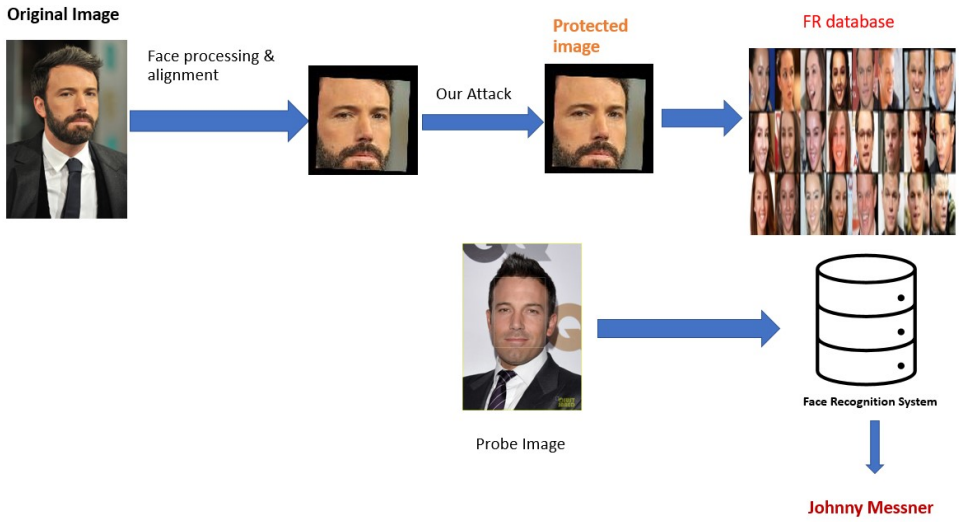


Figure 4.2. Our Attack pipeline

A visual depiction of our attack pipeline is given in Figure 4.2. We design an adversarial attack that attacks the images at inference. We focus on face identification

which is generally used by commercial face recognition systems for surveillance and tracking.

Before extracting the facial features, the faces are first detected and aligned. Then, the neural network models extract features from the probe image. After that, the matching algorithm finds gallery images with the closest feature vectors using a similarity metric such as cosine similarity or euclidean distance. The matched gallery images are then considered as likely subjects corresponding to the person in the probe.

Our attack can be used as a filter for user images before they are put up in an organization's database of gallery images for Face Recognition. The goal is to manipulate and corrupt an FR system so that when a probe image is given to a system, it may not be able to match it with the correct identity and misclassify the probed image.

We create an adversarial image whose feature vector lies away from its original image while also ensuring that the altered image maintains a perceptual similarity to the original image. By maximizing the feature space distance, the image cannot be matched to other images of the same individual, while minimizing the perceptual similarity loss helps maintain the image quality.

We also incorporate the Gaussian Blur and Median Blur into our attack pipeline to ensure that images do not get recognized even if FR systems use denoising filters. Gaussian Blur has been employed to improve the appearance and transferability of the attacked images [9]. However, we find through our experiments in Section 4.5 that the Lowkey does not provide good protection when a high-intensity blur is applied. Thus, we incorporate Gaussian-blur and Median-blur with high intensity to ensure better protection in our attack pipeline.

LPIPS (Learned Perceptual Image Patch Similarity) is a metric used for measuring the perceptual similarity between two images. It calculates the distance between the feature representations of two images obtained from a pre-trained deep neural network(which in our case is Alex-net). Specifically, LPIPS uses the ℓ_2 distance in the feature space of an ImageNet-trained feature extractor to measure the similarity between two images.LPIPS is used for visual perception in adversarial attacks on face recognition by incorporating it as a loss function in the optimization problem.

The goal of our attack is to generate a perturbed image that is visually similar to the original image but is misclassified by the face recognition system. Thus, the objective function of our attack can be given as:

$$\max_{x'} \frac{1}{2n} \sum_{i=1}^{n} \frac{\overbrace{\|f_i(A(x)) - f_i(A(x'))\|_2^2}^{\text{non-smoothed}} + \overbrace{\|f_i(A(x)) - f_i(A(G(x')))\|_2^2}^{\text{smoothed}} + \overbrace{\|f_i(A(x)) - f_i(A(M(x')))\|_2^2}^{\text{transferability}}}{\|f_i(A(x))\|_2} - \alpha \underbrace{\text{LPIPS}(x, x')}_{\text{perceptual loss}} \tag{4.3}$$

where x is the original image, x' is the adversarial image, f_i represents the i^{th} model in our ensemble of models, G is the Gaussian blur function with high intensity, M represents the Median blur smoothing term and A denotes face detection and alignment pipeline followed by resizing based on [50]. The optimization process is performed using signed gradient descent, where the gradient is computed using back-propagation. The optimization is performed several times up to 50 epochs, each time with a different initial perturbation added to the original image, and the best adversarial image x' is returned (i.e., the one that minimizes the objective function).

Gaussian and Median Blur In our adversarial attack pipeline, Gaussian smoothing and Median blur are used to improve the robustness and transferability of the attack. We propose to incorporate Gaussian smoothing and median blur in the objective function by simultaneously maximizing distance to the blurred adversarial image in the feature space. Moreover, this procedure would enforce perturbed images to keep adversarial properties even when blurred and thus improve the robustness of our attack.

The optimization is performed using signed stochastic gradient descent (SGD) with a small step size. At each iteration, the gradient of the loss function with respect to the adversarial image is computed, and the adversarial image is updated in the direction of the gradient. By doing this, we move x', $G(x')$, and $M(x')$ far away from the original image x in the feature spaces of models f_i used in the ensemble. The ensemble contains the ResNet-152 and IR-152 trained with ArcFace and CosFace loss functions.

4.4 Experimental Setting

We use pre-trained feature extractors that are trained on IR and ResNet backbones with ArcFace and CosFace heads. Specifically, we use an ensemble of ResNet-152 and IR-152 with ArcFace and CosFace loss functions. These models are pre-trained on the MS-Celeb-1M dataset which contains over 5.82 million images of over 85k identities. We employ these models in our ensemble to generate attacks and to perform other experiments which will be described in Section 4.5.

We use the Face-Scrub dataset as our benchmark to compute attacks and evaluate the results. The dataset contains 106,863 face images of 530 male and female celebrities

scrapped from the internet. The dataset is provided by the vision and interaction group at NUS Singapore, whom we kindly thank for providing access to the dataset since it is no longer publicly available.

We use a subset of Face-Scrub which contains over 12,000 images that belong to 140 subjects. We discard the duplicate images and also detect and align the faces for all the subjects. Then, we treat 10% of images from each subject as probe images and the remaining 90% as gallery. We identify 24 subjects to protect, apply both the attack generated by [9] and our attack on their corresponding all images and then insert them back into the gallery. This setting simulates that there are some users in an FR system database that have their images protected by our attack among a large population of non-users.

In order to face identification trial, we take each probe image and find its closest match among all subjects in the gallery using cosine similarity. Then we do the evaluation with rank k accuracy. As part of this thesis, we calculate the top 1 and top 5 rank accuracy for evaluating the trail of face identification.

Our work differentiates from work done by [9] in the sense that we perform an exhaustive evaluation of the attack against blur transformations. Blur transformations have been used as a denoising filter against adversarial attacks to remove the perturbations and make the face recognition work [54]. We also base our new attack to include Gaussian Blur and Median Blur with a high intensity of kernel size 11 in our attack pipeline based on our experiments conducted in Section 4.5.

Secondly, lowkey reports its results by only doing probe against the protected identities images and not against the whole gallery. While we probe images against all gallery images and report the top 1 and top 5 accuracies accordingly. This setting shows us the impact our attack has on the accuracy of FR systems and how this impact lead to misclassification.

Finally, we detect and align faces as a pre-processing step before performing adversarial attacks on them. While lowkey does this as part of their attack pipeline. We believe that having faces already aligned and cropped improves the run time of attack rather than doing it as part of the pipeline to generate attacks.

4.5 Evaluating Attacks against Blur Transformations

In this section, we evaluate the robustness of the attacks against blur operations. Initially, we use a subset of the FaceScrub dataset where 5 subjects are protected by attack among a gallery of 24 subjects. We apply blur to all gallery images with various intensity levels by increasing the kernel size to see the impact of blur transformations provided by Kornia against the clean gallery and gallery protected by lowkey to evaluate their robustness.

In order to perform these experiments, we make use of Kornia which is a differentiable computer vision library. We evaluate the impact of Median-blur, Gaussian-blur, Box-blur, Pool-blur, Motion-blur, and Max-blur on the gallery with kernel sizes ranging from 3 to 11. We do not use a kernel size higher than 11 because that would already distort the images further from visual perception and thus make use of blur with high intensity irrelevant for the attack. We also do not use a high kernel size because as pointed out by the work of [9], we need to attack images but also make sure that perceptual similarity is maintained for a human observer so that the user protected can actually use these images online.

4.5.1 Applying Median-blur in inference

Median blur is a type of image-filtering technique used to reduce noise and blur the sharp edges of an image. It works by replacing each pixel in an image with the median value of its neighboring pixels. The median value is calculated by sorting all the pixel values in a given neighborhood (a square or rectangular region) and choosing the middle value.

We apply median-blur on a gallery of clean images and a gallery corrupted with lowkey. We evaluate the top 1 and top 5 rank accuracy. The results of rank 5 accuracy is given in Table 4.1. A visual depiction of the attack on clean and lowkey images is given in the Figure 4.3.

Top-5 Evaluation pipeline Motion_Blur with intensity=3,5,7,9,11 &17				
Attacker	IR_152A	IR_152C	RESNET_152A	RESNET_152C
Clean	99.34	99.34	99.34	100.0
Clean + median_blur=3	99.34	99.34	99.34	100.0
Clean + median_blur=5	99.34	99.34	99.34	99.34
Clean + median_blur=7	99.34	99.34	99.34	99.34
Clean + median_blur=9	96.03	96.03	86.09	83.44
Clean + median_blur=11	99.34	99.34	100.0	99.34
Clean + median_blur=17	77.48	92.05	77.48	75.5
Lowkey	74.17	74.17	74.17	74.83
Lowkey + median_blur=3	74.17	74.17	74.17	74.83
Lowkey + median_blur=5	74.17	74.17	75.5	74.83
Lowkey + median_blur=7	76.16	74.17	77.48	76.82
Lowkey + median_blur=9	76.82	78.81	80.13	78.15
Lowkey + median_blur=11	81.46	82.78	84.11	84.11

Table 4.1. Rank-5 accuracy of Median-blur on the clean and lowkey gallery. Here k indicates the kernel size. Notice how lowkey accuracy tends to go high as we increase the kernel size.

As you see in Figure 4.4 that for clean images, FR (Face Recognition) systems are invariant to the median-blur up to kernel size 11. Surprisingly, when we repeat the same experiment for Lowkey protected images, the accuracy increases as we increase the kernel size. This means that median-blur acts as a remover of the lowkey noise

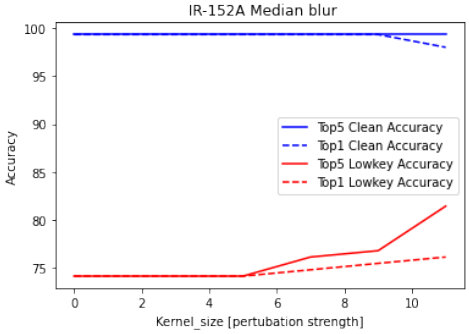
Figure 4.3. The figure highlights the view of transformation visually as we increase the blur size for median-blur on protected images.

and part of the recognition is restored and perturbations are removed. It is possible that increasing the kernel size of the median blur can lead to an increase in the top-k accuracy of the gallery images protected by Lowkey.

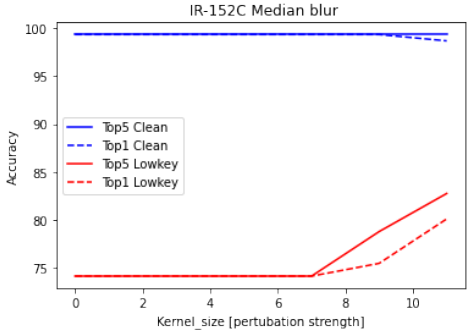
4.5.2 Applying Box-blur in inference

Box blur is a type of image blur that is applied to an image to reduce the level of detail and noise in the image. It is a simple and commonly used technique for image smoothing and is based on averaging pixels within a certain image region. The box blur filter applies an average value of each pixel and its surrounding pixels in a rectangular region, which produces a uniform blur effect across the entire image. The size of the rectangular region is defined by the size of the kernel or the mask that is used for the filter. The larger the kernel, the more blur is applied to the image. Box blur is often used in image processing and computer vision applications to remove noise and smooth the edges of objects in an image.

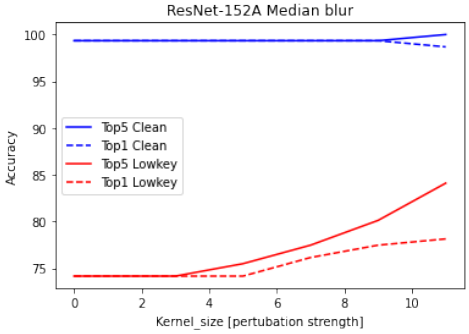
We apply box-blur on the clean gallery with kernel size ranging from k = 3 to 17 and on the lowkey corrupted gallery with kernel size ranging from k = 3 to 11. We compute the rank 5 accuracy which is given in Table 4.2. A visual depiction of an attack on images protected by lowkey is given in Figure 4.5.



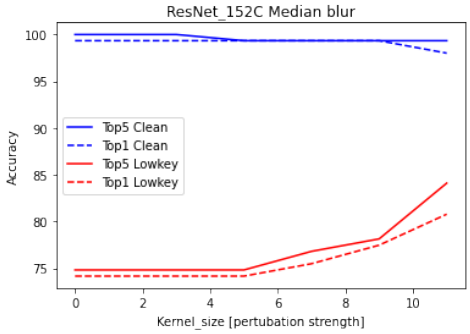
(a) IR-152A median-blur w.r.t k



(b) IR-152C median-blur w.r.t k



(c) ResNet-152A median-blur w.r.t k



(d) ResNet-152C median-blur w.r.t k

Figure 4.4. Median-blur Top 1 and Top 5 Accuracy w.r.t kernel-size

Top-5 Evaluation pipeline Box Blur - 24 probe,24 gallery				
Attacker	IR_152A	IR_152C	RESNET_152A	RESNET_152C
Clean	99.34	99.34	99.34	100.0
Clean + box_blur with k=3	99.34	99.34	99.34	100.0
Clean + box_blur with k=5	99.34	100.0	99.34	100.0
Clean + box_blur with k=7	99.34	99.34	99.34	100.0
Clean + box_blur with k=9	99.34	99.34	100.0	100.0
Clean + box_blur with k=11	100.0	99.34	100.0	100.0
Clean + box_blur with k=17	43.71	70.86	61.59	51.66
Lowkey	74.17	74.17	74.17	74.83
Lowkey + box_blur with k=3	74.17	74.17	75.5	74.83
Lowkey + box_blur with k=5	74.17	74.17	75.5	75.5
Lowkey + box_blur with k=7	76.16	78.15	78.15	78.81
Lowkey + box_blur with k=9	80.79	79.47	86.75	84.11
Lowkey + box_blur with k=11	91.39	84.11	85.43	88.74

Table 4.2. Rank-5 accuracy of Box-blur on the clean and lowkey gallery. Here k indicates the kernel size. Notice how accuracy tends to go increase for the lowkey gallery as we increase the kernel size from 7 to 11.

We can also see the same effect here as Median-blur. As we increase the kernel size up to 11, the clean images remain invariant to the blur transformation. However, when we apply the same to Lowkey protected images, the adversarial noise added by Lowkey gets removed and part of the recognition is restored. This maybe because when we apply box blur to an image, it smooths out the image by averaging pixel values within a kernel. This means that it reduces the noise and perturbations caused by the Lowkey attack, making the image more similar to the original image. As a result, the similarity between the original and perturbed image increases, and the top-k accuracy of the gallery protected with lowkey attack also increases. This relationship can be seen in the figure for top 1 and top 5 rank accuracy in Figure 4.6.

4.5.3 Applying Gaussian Blur in inference

Lowkey uses Gaussian Blur already in its attack pipeline. However, we tended to test it for checking out if it does stand the denoising when a blur with high kernel size is applied to it.

Gaussian blur is a smoothing operation applied to an image to reduce the impact of noise and perturbations caused by the attack. It is a type of image filtering that uses a Gaussian function to blur the image. The Gaussian function is a bell-shaped curve that describes the distribution of values in a dataset, and it is used to compute the weights of neighboring pixels in the blur kernel. The blur kernel is a matrix that specifies how the neighboring pixels are combined to obtain the smoothed value of each pixel in the image.

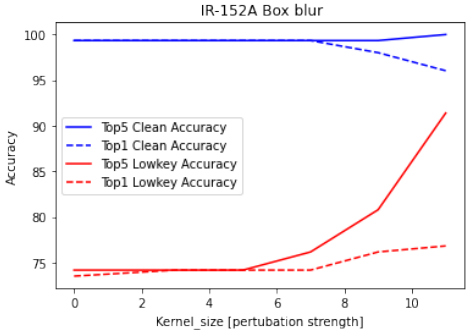
Gaussian blur is used by the lowkey attack as a preprocessing step to generate a

Figure 4.5. The figure highlights the view of transformation visually as we increase the blur kernel size for box-blur on lowkey protected images

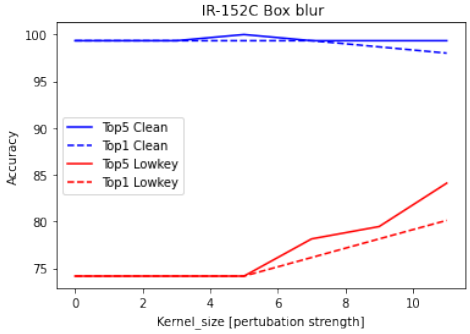
smoothed version of the original image, which is then used as a reference image to optimize the perturbed image. The idea behind using Gaussian blur is that it removes high-frequency noise and details from the image while preserving the edges and structures, making it easier for the optimization algorithm to converge to a perturbed image that is visually similar to the original. Table 4.3 shows the rank 5 accuracy of clean and lowkey corrupted gallery with kernel size ranging from 3 to 11. It is interesting that in Table 4.3 applying Gaussian-blur at inference does not increase the recognition when lowkey is applied. This motivated the usage of Gaussian-blur in the lowkey training of the attack. A visual depiction of this scenario can be seen in Figure 4.7.

4.5.4 Applying Max Blur in inference

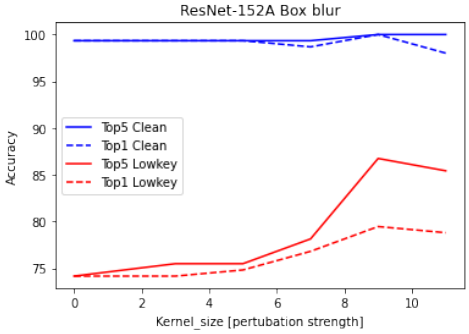
Max blur is a type of image-blurring operation used in computer vision and image processing. In this operation, the pixel value at each location in the image is replaced by the maximum value in a local neighborhood around that location. The size of the local neighborhood is determined by a kernel, which is a small matrix of weights that specifies how much each neighboring pixel contributes to the maximum value calculation.



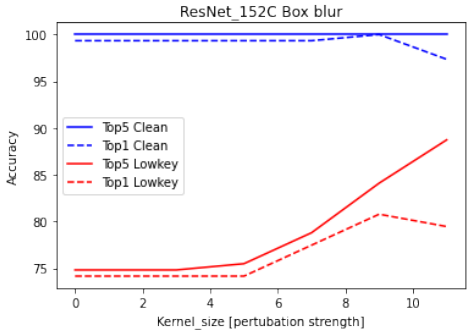
(a) IR-152A Box-blur w.r.t k



(b) IR-152C Box-blur with w.r.t k



(c) ResNet-152A Box-blur w.r.t k



(d) ResNet-152C Box-blur w.r.t k

Figure 4.6. Box-blur Top 1 and Top 5 Accuracy w.r.t kernel size

Top-5 Evaluation pipeline Gaussian Blur - 24 probe,24 gallery				
Attacker	IR_152A	IR_152C	RESNET_152A	RESNET_152C
Clean	99.34	99.34	99.34	100.0
Clean + gaussian_blur with k=3	99.34	99.34	99.34	100.0
Clean + gaussian_blur with k=5	99.34	100.0	99.34	100.0
Clean + gaussian_blur with k=7	99.34	99.34	99.34	99.34
Clean + gaussian_blur with k=9	99.34	99.34	99.34	100.0
Clean + gaussian_blur with k=11	99.34	99.34	100.0	100.0
Clean + gaussian_blur with k=17	99.34	99.34	100.0	100.0
Lowkey	74.17	74.17	74.17	74.83
Lowkey + gaussian_blur with k=3	74.17	74.17	74.83	74.83
Lowkey + gaussian_blur with k=5	73.51	74.17	75.5	74.83
Lowkey + gaussian_blur with k=7	74.17	72.85	77.48	74.83
Lowkey + gaussian_blur with k=9	75.5	74.83	81.46	76.82
Lowkey + gaussian_blur with k=11	75.5	74.83	83.44	78.15

Table 4.3. Rank-5 accuracy of Gaussian-blur on the clean and lowkey gallery. Here k indicates the kernel size.

We test the attacks generated on Max-blur transformation to evaluate the robustness of the attack against this transformation. We observe that lowkey is robust to this transformation and it doesn't impact its accuracy much. The rank 5 accuracy is given in Table 4.4 while a visual depiction of this scenario is given in Figure 4.8.

Top-5 Evaluation pipeline Max Blur - 24 probe,24 gallery				
Attacker	IR_152A	IR_152C	RESNET_152A	RESNET_152C
Clean	99.34	99.34	99.34	100.0
Clean + max_blur with k=3	99.34	99.34	99.34	99.34
Clean + max_blur with k=5	99.34	99.34	99.34	100.0
Clean + max_blur with k=7	99.34	99.34	99.34	100.0
Clean + max_blur with k=9	99.34	99.34	99.34	100.0
Clean + max_blur with k=11	99.34	99.34	99.34	99.34
Clean + max_blur with k=17	99.34	99.34	99.34	100.0
Lowkey	74.17	74.17	74.17	74.83
Lowkey + max_blur with k=3	72.85	74.17	74.17	74.83
Lowkey + max_blur with k=5	74.17	74.17	75.5	74.83
Lowkey + max_blur with k=7	74.83	73.51	76.16	75.5
Lowkey + max_blur with k=9	74.17	70.86	76.16	74.83
Lowkey + max_blur with k=11	74.17	71.52	76.16	72.85

Table 4.4. Rank-5 accuracy of Max-blur on clean and lowkey gallery. Here k indicates the kernel size.

4.5.5 Applying Motion Blur in inference

Motion blur is a type of blur that occurs when there is relative motion between the camera and the subject being photographed. It results in a blurred image with streaks or smears in the direction of the motion. Motion blur can occur when either the camera or the subject is in motion and can be intentional or unintentional.

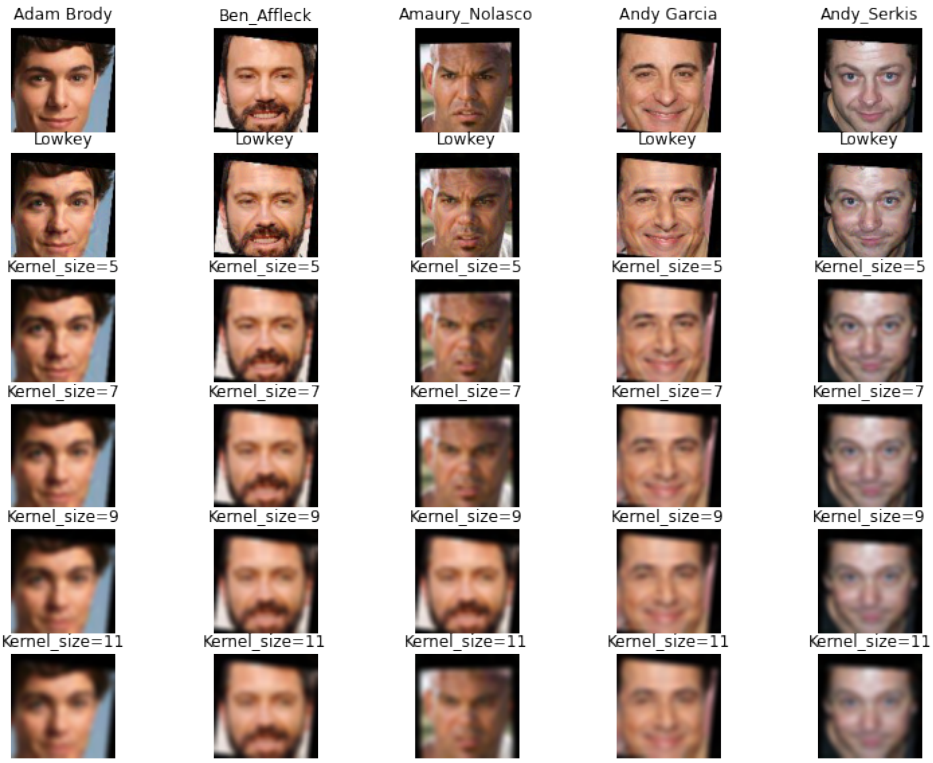


Figure 4.7. The figure highlights the view of transformation visually as we increase the kernel size for Gaussian-blur on lowkey protected images.

Intentional motion blur can be used creatively in photography to give a sense of movement, while unintentional motion blur can be a problem in situations where the camera is not stable or the subject is moving quickly.

A great deal of study is going on to protect users from surveillance. Motion blur simulates the situation where the image of any person is captured for identification from surveillance footage and cameras. Thus, we also try to see the protection of lowkey attacks against Motion-blur to see if the system would allow protection even when the image being probed is footage from a surveillance camera.

We use the motion-blur filter from Kornia. We set the angle to 90° and do the blur in the forward direction. We vary the kernel size from (3,3) to (11,11) for both clean gallery and lowkey corrupted gallery. The results of the top 5 accuracy are given in the Table 4.5. We see that it doesn't impact the accuracy much. Lowkey attacks can be said to be robust to this transformation. A visual depiction of the attack on lowkey protected images is given in Figure 4.9.

4.5.6 Applying Pool blur in inference

Pool blur is a type of blur operation that is used to reduce the resolution of an image. It involves dividing an image into non-overlapping rectangular regions, called

Figure 4.8. The figure highlights the view of transformation visually as we increase the kernel size for Max-blur on lowkey protected images.

Top-5 Evaluation pipeline Motion_Blur with intensity=3,5,7,9,11 &17				
Attacker	IR_152A	IR_152C	RESNET_152A	RESNET_152C
Clean	99.34	99.34	99.34	100.0
Clean + motion_blur=3	99.34	99.34	99.34	100.0
Clean + motion_blur=5	99.34	99.34	99.34	99.34
Clean + motion_blur=7	99.34	99.34	99.34	99.34
Clean + motion_blur=9	99.34	100.0	99.34	100.0
Clean + motion_blur=11	99.34	100.0	99.34	100.0
Clean + motion_blur=17	99.34	99.34	99.34	100.0
Lowkey	74.17	74.17	74.17	74.83
Lowkey + motion_blur=3	74.17	74.17	74.83	74.83
Lowkey + motion_blur=5	74.17	74.17	74.17	74.83
Lowkey + motion_blur=7	74.17	74.17	74.17	74.83
Lowkey + motion_blur=9	74.17	74.17	74.83	75.5
Lowkey + motion_blur=11	74.17	74.83	74.83	78.81

Table 4.5. Rank-5 accuracy for Motion-blur applied on the clean and lowkey gallery. Here k indicates the kernel size.

pools, and then replacing each pool with a single pixel that represents the average color of the pool. This process results in a lower-resolution image, where each pixel represents a larger area of the original image.

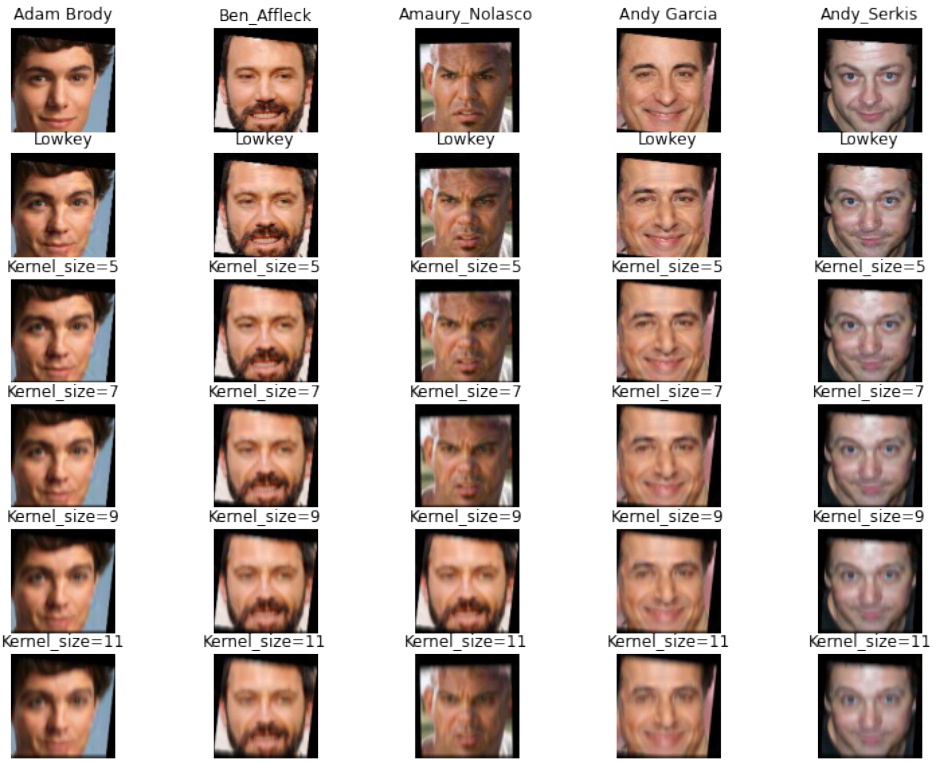


Figure 4.9. A visual depiction of motion blur applied on lowkey protected images with k = 3 to 11.

Its implementation in kornia is based on the work of [59]. Specifically, the paper introduced a new type of pooling operation called "pooling by overlapped pooling (POP)", which divides the input feature map into non-overlapping regions and then pools them using the traditional max pooling or average pooling operations. The resulting feature map is then overlapped and concatenated to form the final output feature map. This method was shown to improve the shift-invariance of CNNs and outperform other methods on several image classification benchmarks.

We test lowkey attack for its robustness against the pool blur with kernel size ranging from 3 to 11. The top accuracy results of this experiment are given in Table 4.6. The visual depiction of how protected images change as we increase the kernel size is can be seen in Figure 4.10. As you can see that this does not impact the lowkey attack. Thus, it can be safe to assume that lowkey is robust to pool-blur up to kernel size 11.

4.6 Generating attack with Median-blur

The experiments regarding the robustness of lowkey to blur transformations in Section 4.5 gave a good overview of how much the lowkey attack is resistant to blur transformations. As we saw in Section 4.5 that by applying Median-blur and

Figure 4.10. A visual depiction of pool-blur applied on lowkey protected images with k = 3 to 11.

Attacker	IR_152A	IR_152C	RESNET_152A	RESNET_152C
Clean	99.34	99.34	99.34	100.0
Clean + pool_blur with k=3	99.34	99.34	99.34	100.0
Clean + pool_blur with k=5	99.34	99.34	99.34	100.0
Clean + pool_blur with k=7	99.34	99.34	99.34	100.0
Clean + pool_blur with k=9	99.34	99.34	99.34	99.34
Clean + pool_blur with k=11	99.34	99.34	99.34	99.34
Clean + pool_blur with k=17	99.34	99.34	99.34	100.0
Lowkey	74.17	74.17	74.17	74.83
Lowkey + pool_blur with k=3	74.17	74.17	74.83	75.5
Lowkey + pool_blur with k=5	74.17	74.17	76.16	74.83
Lowkey + pool_blur with k=7	74.17	74.17	76.16	75.5
Lowkey + pool_blur with k=9	73.51	74.17	75.5	75.5
Lowkey + pool_blur with k=11	73.51	72.85	76.16	74.83

Table 4.6. Rank-5 accuracy for Pool-blur applied on clean and lowkey gallery. Here k indicates the kernel size.

Box-blur with kernel size ranging from 9 to 11 on lowkey protected images, it was observed that the top 5 accuracy increased, indicating that these transformations break the attack of lowkey and enable recognition.

This highlights that any face recognition system could use these blur transformations

in their pre-processing step on gallery images to reduce the effectiveness of the lowkey attack. However, it also implies that the user is not completely protected by lowkey.

Therefore, we design a new attack that includes Median-blur and Gaussian-blur with high intensity of kernel size up to 11 and improves the attack that was proposed by [9]. We apply the attack on the same identity images that were protected by lowkey and compute the top 1 and top 5 rank accuracy for a fair comparison. The equation of our attack is given in Eq 4.3. The top 5 accuracy is given in Table 4.7. We also calculate the top 1 accuracy which is given in Table 4.8. A visual depiction of the comparison between lowkey and our attack is given in Figure 4.11.

Evaluation Results Top 5 Accuracy				
Attacker	IR_152A	IR_152C	RESNET_152A	RESNET_152C
Clean	99.47	99.36	99.57	99.47
Lowkey	87.18	87.07	87.29	87.18
Our Attack	87.18	87.07	87.82	87.29

Table 4.7. Rank-5 accuracy for Clean, low-key, and Our attack

Evaluation Results Top 1 Accuracy				
Attacker	IR_152A	IR_152C	RESNET_152A	RESNET_152C
clean	99.36	99.36	99.57	99.47
Lowkey	86.97	87.07	87.29	87.18
Our Attack	87.07	87.07	87.39	87.29

Table 4.8. Rank-1 accuracy for Clean, lowkey, and Our attack

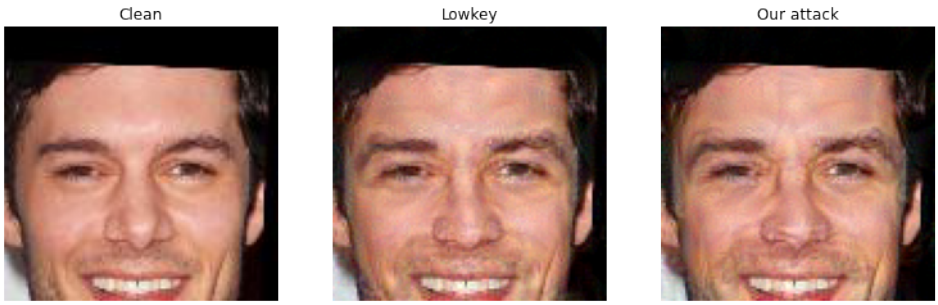


Figure 4.11. Visual Comparison of clean, lowkey and our attack images.

We see that our attack provides protection in the gallery which is equivalent to

lowkey when probed. We also see that our attack removes some visible distortions as in the lowkey attack while also maintaining the same amount of protection against recognition.

4.7 Run Time

Running adversarial attacks on images is a computationally expensive and time-consuming task. We run lowkey attacks and our proposed attack on Telsa T4 GPU provided by Google Colab. Our proposed attack averages between 35-40 seconds to compute an attack on an image while lowkey takes 50-55 seconds on average. On the task of protecting one identity which contains 39 images, our attack takes 24 minutes and 42 seconds while for the same lowkey takes 29 minutes. Thus, if a user needs to protect images at scale, our attack runs faster.

4.8 Discussion

In this work, we propose an adversarial attack that can be used for protection by users against unauthorized face identification. The attack we develop performs adversarial pre-processing on user images prior to their upload on social media. The pre-processed images are rendered unusable for third-party organizations that gather them for facial recognition purposes. We initially redesigned the attack that was proposed by [9] and conducted controlled experiments of their attack against blur transformations using the Kornia library.

We came to the conclusion that Lowkey does not give better protection against Median-blur and Box-blur filters with kernel sizes of 9 to 11. This shows that the face recognition systems can remove the attack if they employ these blur operations in their pre-processing pipeline before adding the images into their database. Thus, we propose an attack that includes the Median-blur with high intensity as part of the attack pipeline to generate attacks. This would lead to a more resistant attack against transformations while giving the same accuracy. We find that our attack performs equally well against the pre-trained FR models and is able to resist the recognition of subjects protected by our attack. Thus, this also leads to the conclusion that attacks that use blur transformations in their attack pipeline are harder to defend against and give better protection from face recognition systems overall.

However, it is pertinent to note that we do not claim to protect users 100% of the time. Face Recognition systems are not fragile and many of the state-of-the-art face recognition systems have been trained on large datasets with millions of images.

On the other hand, adversarial attacks cannot be trained on such a high amount of data to evade face recognition and provide better perturbations because of constraints

of runtime and resources required to run them. An attack like ours and lowkey for one subject would take almost an hour to compute on all their corresponding images.

Chapter 5

Conclusion

In this thesis, we first did a thorough literature review of the related work and progress achieved in Face Recognition and adversarial attacks against Face Recognition systems. We saw that current work is not enough and there has been a trend of ongoing research in this area. The first paper to propose the vulnerability of face recognition systems to adversarial attacks was [2].

We study, recreate and do a comprehensive review of the lowkey attack[9] that was released in 2021. Lowkey is state of the art adversarial attack that can evade even commercial face recognition systems of Amazon Rekognition and Microsoft Azure Face and reduce the top 50 accuracy to less than 1 percent. We evaluated the lowkey against blur transformations with varying kernel sizes ranging from 3 to 11. We came to the conclusion that for Median-blur and Box-blur, lowkey protected images get recognized when a blur with kernel size 9 to 11 is applied.

Thus, we proposed our own attack that provides for equivalent protection but includes the Median-blur as a part of our attack pipeline to generate better protection even if the FR systems use these blur transformations for denoising the protected images in their recognition pipeline. We primarily generate attacks using an ensemble of IR-152 and ResNet-152 that are trained with ArcFace and Cosface Loss functions. To evaluate our attacks, we use the same models as feature extractors to extract features, do matching and compute the rank 1 and rank 5 accuracies.

We conclude that our attack runs slightly faster than lowkey, and provides equivalent protection against Face Recognition. We also find as shown in 4.11 that our attack generates slightly smoother and visually pleasing perturbations that match visual perception better than lowkey.

However, computing adversarial attacks is a time-consuming and computationally expensive task as compared to training a face recognition system in comparison. We also do not claim in this thesis to protect against all face recognition systems and ensure protection 100 percent of the time.

However, I do hope that through this work, we are able to have a discussion regarding privacy and unauthorized face recognition. As technology continues to evolve, I believe that privacy-protecting tools should also evolve at the same pace to prevent misuse of the technology. While we are under no illusion that this proposed system is itself future-proof, we believe it is an important and necessary first step in the development of user-centric privacy tools to resist unauthorized machine learning models. We hope that follow-up work in this space will lead to long-term protection mechanisms that prevent the mining of personal content for user tracking and classification.

In the future, we aim to include the Box-blur and Motion-blur as part of our attack pipeline to generate attacks. Also, we aim to test our attack methodology against state-of-the-art Face Recognition models such as VGG-face, Open-face, FaceNet, and GoogleNet. We also aim to test our attacks in the black-box setting against Amazon Rekognition and Microsoft Azure Face in the future.

Bibliography

[1] Dhaval Adjodah, Gretchen Greene, Joshua Joseph, Thomas Miano, Francisco O., and Daniel Pedraza. equalais - empowering people & thwarting machines, April 2018.

[2] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1563–1572, 2016.

[3] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR, 2018.

[4] Qiong Cao, Li Shen, Weidi Xie, Omkar M. Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. *CoRR*, abs/1710.08092, 2017.

[5] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2016.

[6] Varun Chandrasekaran, Chuhan Gao, Brian Tang, Kassem Fawaz, Somesh Jha, and Suman Banerjee. Face-off: Adversarial face obfuscation. *arXiv preprint arXiv:2003.08861*, 2020.

[7] Charu A Chandrasekhar. Flying while brown: Federal civil rights remedies to post-9/11 airline racial profiling of south asians. *Asian Lj*, 10:215, 2003.

[8] Jun-Cheng Chen, Rajeev Ranjan, Amit Kumar, Ching-Hui Chen, Vishal M Patel, and Rama Chellappa. An end-to-end system for unconstrained face verification with deep convolutional neural networks. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 118–126, 2015.

[9] Valeriia Cherepanova, Micah Goldblum, Harrison Foley, Shiyuan Duan, John Dickerson, Gavin Taylor, and Tom Goldstein. Lowkey: Leveraging adversarial attacks to protect social media users from facial recognition. *arXiv preprint arXiv:2101.07922*, 2021.

[10] Umur A Ciftci, Gokturk Yuksek, and Ilke Demir. My face my choice: Privacy enhancing deepfakes for social media anonymization. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1369–1379, 2023.

[11] K Crockford. How is face recognition surveillance technology racist? aclu, 2020.

[12] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4685–4694, 2019.

[13] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019.

[14] Camilla Dul. Facial recognition technology vs privacy: The case of clearview ai. *QMLJ*, page 1, 2022.

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

[16] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[17] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 87–102. Springer, 2016.

[18] Robert Hart. Clearview ai fined 9.4 million in u.k. for illegal facial recognition database. *Forbes*, May 2022.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

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

[21] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *Similarity-Based Pattern Recognition: Third International Workshop, SIMBAD 2015, Copenhagen, Denmark, October 12-14, 2015. Proceedings 3*, pages 84–92. Springer, 2015.

[22] Gary B. Huang, Marwan A. Mattar, Tamara L. Berg, and Eric Learned-Miller. In *Labeled Faces in the Wild: A Database forStudying Face Recognition in Unconstrained Environments*, 2008.

[23] Jim Isaak and Mina J Hanna. User data privacy: Facebook, cambridge analytica, and privacy protection. *Computer*, 51(8):56–59, 2018.

[24] Shaharyar Khan, Ilya Kabanov, Yunke Hua, and Stuart Madnick. A systematic analysis of the capital one data breach: Critical lessons learned. *ACM Transactions on Privacy and Security*, 26(1):1–29, 2022.

[25] Brendan F Klare, Ben Klein, Emma Taborsky, Austin Blanton, Jordan Cheney, Kristen Allen, Patrick Grother, Alan Mah, and Anil K Jain. Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1931–1939, 2015.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[27] Lixiang Li, Xiaohui Mu, Siying Li, and Haipeng Peng. A review of face recognition technology. *IEEE Access*, 8:139110–139120, 2020.

[28] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017.

[29] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. *arXiv preprint arXiv:1612.02295*, 2016.

[30] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083, 2017.

[31] K. Mahantesh and H J Jambukesh. A transform domain approach to solve pie problem in face recognition. *2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT)*, pages 270–274, 2017.

[32] Iacopo Masi, Yue Wu, Tal Hassner, and Prem Natarajan. Deep face recognition: A survey. In *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pages 471–478. IEEE, 2018.

[33] Tom M Mitchell and Tom M Mitchell. *Machine learning*, volume 1. McGraw-hill New York, 1997.

[34] Alex Najibi. Racial discrimination in face recognition technology. *Science in the News*, 26, 2020.

[35] Hongwei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets. *2014 IEEE International Conference on Image Processing (ICIP)*, pages 343–347, 2014.

[36] Hieu V Nguyen and Li Bai. Cosine similarity metric learning for face verification. In *Computer Vision–ACCV 2010: 10th Asian Conference on Computer Vision, Queenstown, New Zealand, November 8-12, 2010, Revised Selected Papers, Part II 10*, pages 709–720. Springer, 2011.

[37] Kate O'Flaherty. China facial recognition database leak sparks fears over mass data collection, Feb 2019.

[38] Swami Sankaranarayanan, Azadeh Alavi, and Rama Chellappa. Triplet similarity embedding for face verification. *arXiv preprint arXiv:1602.03418*, 2016.

[39] Sara Shahsavarani, Morteza Analoui, and Reza Shoja Ghiass. Deep-id: A novel model for multi-view face identification using convolutional deep neural networks. *ArXiv*, abs/2001.07871, 2020.

[40] Shawn Shan, Emily Wenger, Jiayun Zhang, Huiying Li, Haitao Zheng, and Ben Y Zhao. Fawkes: Protecting privacy against unauthorized deep learning models. In *29th USENIX security symposium (USENIX Security 20)*, pages 1589–1604, 2020.

[41] Maya Shwayder. Clearview ai facial-recognition app is a nightmare for stalking victims, Jan 2020.

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

[43] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[44] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, D. Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.

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

[46] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings*

of the IEEE conference on computer vision and pattern recognition, pages 1701–1708, 2014.

[47] Anh Tuan Tran, Tal Hassner, Iacopo Masi, and Gérard Medioni. Regressing robust and discriminative 3d morphable models with a very deep neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5163–5172, 2017.

[48] Fatemeh Vakhshiteh, Ahmad Nickabadi, and Raghavendra Ramachandra. Adversarial attacks against face recognition: A comprehensive study. *IEEE Access*, 9:92735–92756, 2021.

[49] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018.

[50] Qingzhong Wang, Pengfei Zhang, Haoyi Xiong, and Jian Zhao. Face. evolve: A high-performance face recognition library. *arXiv preprint arXiv:2107.08621*, 2021.

[51] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*, pages 499–515. Springer, 2016.

[52] Mika Westerlund. The emergence of deepfake technology: A review. *Technology Innovation Management Review*, 9:40–53, 11/2019 2019.

[53] Cameron Whitelam, Emma Taborsky, Austin Blanton, Brianna Maze, Jocelyn Adams, Tim Miller, Nathan Kalka, Anil K Jain, James A Duncan, Kristen Allen, et al. Iarpa janus benchmark-b face dataset. In *proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 90–98, 2017.

[54] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 501–509, 2019.

[55] Ying Xu, Kiran Raja, Raghavendra Ramachandra, and Christoph Busch. Adversarial attacks on face recognition systems. In *Handbook of Digital Face Manipulation and Detection*, pages 139–161. Springer, Cham, 2022.

[56] Ying Xu, Kiran Raja, Raghavendra Ramachandra, and Christoph Busch. *Adversarial Attacks on Face Recognition Systems*, pages 139–161. Springer International Publishing, Cham, 2022.

[57] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[58] Yi Zeng, Enmeng Lu, Yinqian Sun, and Ruochen Tian. Responsible facial recognition and beyond. *arXiv preprint arXiv:1909.12935*, 2019.

[59] Richard Zhang. Making convolutional networks shift-invariant again. In *International conference on machine learning*, pages 7324–7334. PMLR, 2019.

[60] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[61] Xiangyu Zhu, Zhen Lei, Xiaoming Liu, Hailin Shi, and Stan Z Li. Face alignment across large poses: A 3d solution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 146–155, 2016.

[62] Weiwei Zhuang, Liang Chen, Chaoqun Hong, Yuxin Liang, and Keshou Wu. Ft-gan: face transformation with key points alignment for pose-invariant face recognition. *Electronics*, 8(7):807, 2019.